The Reuse of Knowledge in Ripple Down Rule Knowledge Based Systems

Deborah Christina Richards

THE UNIVERSITY OF
NEW SOUTH WALES

A Thesis submitted in the fulfilment
of the requirements for the degree of
Doctor of Philosophy

Department of Artificial Intelligence
School of Computer Science and Engineering
The University of New South Waltes
Sydney, NSW, 2052
Australia

December 1998

Abstract

The work reported in this thesis is motivated by the belief that knowledge-based systems (KBS) research needs to focus more on users' needs and cater for the various decision situations in which users will find themselves. To build individual systems that cater for all the activities that may be needed is not feasible or desirable. The problems associated with capturing knowledge are well known and the ability to capture knowledge once and access and manipulate the knowledge in multiple ways is highly desirable. It adds value to the original knowledge and offers all the benefits associated with the reuse of resources. Thus, the problem becomes one of knowledge reuse. The research question pursued in this thesis is "can knowledge captured for one purpose, such as consultation, be reused to support a wide range of alternative purposes, such as critquing or tutoring, allowing the user to answer different types of questions according to their current circumstances"? Further, this question was to be answered in a situated cognition, dynamic knowledge framework.

The system developed in this thesis is based on the Multiple Classification Ripple Down Rule (MCRDR) knowledge acquisition and representation technique. MCRDR is a form of cased-based reasoning which uses rules to index cases. The cases and the exception structure provide a situated cognition based approach. It was found that MCRDR already supported the activities of knowledge acquisition (KA), maintenance, inferencing and validation within the one system. However, a range of activities, which are referred to as reflective activities, were not well supported by the performance knowledge being captured using MCRDR. These activities include explanation, tutoring, causal modelling, 'what-if' analysis, hypothesis testing and student modelling. These reflective activities required uncovering the concepts, both primitive and more abstract, in the knowledge base. In addition these concepts needed to be ordered and structured. It was found that Formal Concept Analysis (FCA) provided a way of retrospectively uncovering an abstraction hierarchy which could be used to support the reflective activities.

A tool was developed which allowed FCA to be used with MCRDR. The approach was evaluated by consulting experts about the discovered concepts and carrying out a student survey comparing a number of different rule representations. Although these evaluations are not definitive, they clearly suggest that an integrated MCRDR-FCA system provides the types of KA and reuse required.

A number of related issues were addressed in this thesis. One is the removal of repetition from RDR KBS and experiments performed using Induct and Rough Set Theory are reported and discussed in this thesis. Another issue is how to support the KA activity when multiple sources of expertise are involved. A requirments engineering (RE) framework has been developed as a solution to this problem and is an extension of the FCA modelling work.

Perhaps the most significant contribution of this thesis is the demonstration that we can start with assertional knowledge that is simply acquired and automatically derive terminological knowledge. Such an approach may be an answer to the KA bottleneck problem and the diffulties associated with building models such as unreliability and variation between users and the articulation, maintenance and validation of such models.

Acknowledgements

There are many people I must thank. Without their assistance the pages that follow would be blank. Firstly, I thank my supervisor Paul Compton for his various contributions which include: encouragement, knowledge, advice, understanding, flexibility, RDR and even babysitting. I thank Tim Menzies for his enthusiasm and support and for pushing me to look further at the area of RE and conflict resolution. I also thank Tim for his valuable comments on an earlier draft of this thesis. I thank the academic and administration staff in the School of Computer Science and Engineering. I thank the Australian Federal Government and the School of Computer Science and Engineering at UNSW for their financial support which made pursing my dream possible. I thank my friends in the AI lab at UNSW just for being there. Phil Preston deserves a special mention as I could always rely on him to help me out whether the problem concerned software or arranging a piece of equipment for a seminar. I apologise to those who have had a quite weekend in the lab disturbed by my three noisy children. I wish you all success.

Beyond Kensington campus, I have many others to thank. Bob Colomb has answered many an email message and provided me with a range of software and files each time I have asked. And his response time is truly remarkable. I thank the four experts that gave me their time to provide evaluations: Glenn Edwards, Zvi Hochman, Geoff Taylor and Mary Mulholland. Others who have kindly assisted me include Jacek Sienkiewicz, Klaus Biedermann, Frank Vogt and Peter Burmeister. I have taken much inspiration, and encouragment, from Bob Colomb, Brian Gaines, Zdzislaw Pawlak, Mildred Shaw and Rudolf Wille.

On a more personal level, I thank my mother for giving me the confidence to begin and complete such a journey. I thank my children Sally, William and James for your unquestioning love and hope that you will follow your dreams as I have done. You have been my reality. I always have and always will put you first. A very special thanks goes to my girlfriends: Deidre Vaill, Helen Terry, Susan Campbell, Pauline Holmwood, Anne Green and Brigit ten Bruggencate. I thank Deidre for driving my kids to and from school for the last three years. You gave me time, which is such a precious gift. Helen had the privilege, or misfortune, of being my only friend who lived near UNSW and often minded my kids while I was at Uni during the school holidays. I am indebted to all of you for your support and very real assistance in caring for my children when I had paper deadlines or was away at conferences. This thesis is a tribute to the sisterhood of women.

Table of Contents

List of Figures

List of Tables

Chapter 1

1 The Need to Reuse Knowledge

The motivation for the work reported in this thesis is the belief that not only is it beneficial to reuse knowledge but it is essential if we wish to build knowledge based systems (KBS) that meet the needs of users. Reuse can range over a number of dimensions (Menzies, personal communication) including:

- Reuse by the same/different person.
- Reuse over the same/different task.
- Reuse for the same/different purposes or activities.
- Reuse at the same/different time.
- Reuse over the same/different software application.

The focus of this research is on the reuse of knowledge for different activities. A more appropriate term for the type of reuse of interest is adapting or *repurposing*, a term Clancey has taken from the hypermedia field. Based on a situated view of expert knowledge and action and an emphasis on user-requirements, the goal of this thesis was to see if a knowledge based system could be built that allowed the user to select from a wide range of activities using and reusing knowledge according to their current purpose and decision situation. Activities are seen as related to a purpose with each activity designed to answer different types of questions relating to the same knowledge. An activity is seen as distinct from a task because a task in knowledge engineering (KE) is typically equated with a problem type such as diagnosis, planning or configuration. To differentiate a task and activity further we can view a task as related to making a decision in the world. An activity is not concerned so much with the world but with the individual (or group) and how they arrive at a decision. A task such as room allocation could be achieved through different activities such as consultation, learning about the domain, proposing a solution to have critiqued or exploring different scenarios. To avoid confusion with more conventional meanings of reuse, the type of reuse being pursued is referred to in this thesis as *activity-reuse.* While some have looked at building systems that perform one or a small number of activities, which could include knowledge acquisition (KA), consultation, maintenance, explanation, critiquing or ‘what-if’ analysis, this study is concerned with providing all of these activities and more within the one session because user requirements will differ between users and in different situations for the same user.

In addition to a different focus on the type of knowledge reuse, another distinguishing feature of this thesis is a breakaway from mainstream approaches to KBS development, which emphasize the need to build terminological KBS (T-boxes[1]) at the knowledge level. Such approaches require complex analysis and modelling of the ontologicial, problem solving and/or domain knowledge as prerequisites to system development. Developing complex models is time-consuming and the emphasis on capturing a "good" and comprehensive model before knowledge acquisition can begin contributes to the knowledge acquisition bottleneck. The difficulties associated with capturing ontological or problem-solving models is due to the well recognised fact that much expert action is subconscious and not easily described (Ignizio 1991) and the inherently unreliable nature of models. A descriptive model can be seen as: "merely an abstraction, a description and generator of *behaviour patterns over time*, not a mechanism equivalent to human capacity" (Clancey 1993, p.89). It has been shown that models are imperfect representations that vary not only between experts but also over time with the same expert (Gaines and Shaw 1989).

The work described in this thesis is based on a paradigm that captures a simple model in terms of attribute-value pairs and the related conclusions without the need for modelling of the data into an abstraction hierarchy or identification of the nature of the problem to be solved. The simple model is being captured into an assertional[2] (A-box) KBS which is a performance system that can be executed and validated. In keeping with the situated cognition view, this thesis describes a KBS that supports human-computer interaction in modes that seem to be more in keeping with the way a human may perform that activity. Just as most human action is reflexive (Winograd and Flores 1986) the system described in this thesis performs knowledge acquisition, maintenance and inferencing in a reflexive[3] mode without the need for reflective action on the part of the expert. However, reflective activities such as explanation, tutoring, 'what-if' analysis and causal modelling also need to be supported.

The knowledge acquisition and representation technique used as the foundation of the work reported in this thesis is known as Ripple-Down Rules (RDR) (Compton and Jansen 1990). RDR have been built from a situated standpoint which acknowledges that

---

[1] Terminological KB consist of terms structured into inheritance networks (Brachman 1979). Their main building blocks are concepts and roles and they reason by determination of subsumption between concepts (Nebel 1991).

[2] Assertional KBS are made up of executable assertions (such as rules) that assert the relationships between terms (such as conditions and conclusions).

[3] The terms reflexive and reflective are used in this thesis to distinguish between acts that are automatic and often subconsciously performed and acts that require deliberative thought often involving the consideration of abstract models, respectively.

the capture of a performance model may be easier and more reliable than reliance on acquiring a complete descriptive model. The use of RDR as the vehicle for the *activity-reuse* research reported in this thesis has resulted in a refinement of the research goals. It was the goal of this research to see if knowledge captured for one purpose or activity such as KA could be reused for another activity such as critiquing with only changes to the user interface. Prior to this study, RDR could already successfully (Edwards et al 1993) handle a number of activities such as KA, inferencing, maintenance and validation all within the same system. These activities are performed in a reflexive manner by the expert with minimal consideration of the model being built. As described in Chapter Three, RDR develops rules by going straight from data to conclusions without the intermediate step of building an abstract model of the data (see the inference structure of heuristic classification given in Clancey (1985)). All rules in an RDR KBS consist of a final conclusion and a set of attribute-value pairs which are the rule conditions. The close link between the use of RDR and the goals of this research meant that the activities already supported required minimal further research effort and led to a focus on the limitations of RDR that were relevant to the *activity-reuse* of knowledge. The first limitation was the existence of repeated knowledge in an RDR KBS. While repetition is not excessive and does not greatly affect system performance, the removal of repetition was seen as beneficial for the activities involving explanation. This work is described in Chapter Four. The second limitation was the lack of higher level models which were necessary for the reflective modes such as critiquing, ‘what-if’ analysis and explanation. It became apparent that the key to supporting these activities was an understanding of the concepts in the KBS and the relationships between them. Others have found that models of abstraction are beneficial for teaching (Schon 1987) and for explanation (Clancey 1993). Of particular importance was the ability to find the higher concepts which were implicitly represented by the low-level concepts in the form of primitive rules. As described in Chapter Five, formal concept analysis (FCA) (Wille 1982) has been employed not only to uncover the higher-level abstractions in the RDR KBS but also the structure between all concepts in the KBS to support the reflective modes. To support determination of the closeness of concepts, a nearest neighbour algorithm was developed and is reported in Section 3.3.2.1.

Another activity that was not supported by RDR was the ability to compare the knowledge contained in multiple KBS acquired from multiple sources of expertise. The FCA work was extended to allow the development of a single KBS from multiple KBS. The ability to detect and manage conflicts between viewpoints can be seen as a requirements engineering (RE) activity. An RE framework which is an extension to the FCA work is described in Chapter Six. Thus it was the goal of this research to address these limitations of RDR in order to provide a KBS technique that supports a wide range of *activity-reuses*. Chapters Seven, Eight and Nine describe and evaluate the multipurpose system developed.

In the next section the benefits of reuse at a general level beyond *activity-reuse* are considered. Firstly, we look at the material or quantitative benefits of reuse followed by consideration of the more soft benefits of building systems that people want and are able to reuse.

1.1 The Benefits of Reuse

The reuse of software components has become widely accepted due to the following factors:

1. Savings in cost
2. Savings in time
3. An increase in reliability (Hemmann and Voss 1993a).

There are two main methods of software reuse: using what you already have as building blocks or using an existing pattern to create new code. Hemmann and Voss (1993a) term these approaches composition and generation respectively. The former is the more widely accepted practice and appears to be more natural for software engineers. In the knowledge engineering community there is a similar situation with much research into developing building blocks (e.g. Chandrasekaran 1986, Steels 1993) that can be reused for different applications. However, in both software and knowledge engineering reuse research the focus is on the technical person, the software or knowledge engineer (KE). The focus in this study is on the end-user or the expert who is not directly assisted by the building block approach.

There is much similarity between the efforts towards reuse in software engineering and knowledge engineering. The focus in software engineering on the reuse of programs and procedures is similar to the focus on the reuse of problem solving methods in knowledge engineering. Also the benefits of reusing the representation language and the method of inference for different applications are obvious and much research focuses on these aspects (Clancey 1992). However a valuable lesson from software engineering is that different languages will affect whether and how that component can be reused. Analogously in knowledge engineering, different KA and knowledge representation techniques may affect the reusability of that knowledge (Chandrasekaran and Johnson 1993).

Software components are not the only resources that can benefit from reuse. Data are considered to be a major resource and the explosion of the information age confirms its value. Database management systems (DBMS) are an important and necessary development to allow organisations to manage their data. Key benefits of DBMS are a reduction in redundancy and repetition. Data can also be captured once and reused for a whole range of purposes. Knowledge is value-added data and includes such things as experience, insight and skill. The use of data dictionaries to support the reuse of data is similar to the use of common ontologies for describing knowledge. The work done by Jansen and Compton (1988) on the knowledge dictionary is an even closer development of the data dictionary idea.

The benefits of reusing knowledge, which includes both domain and problem solving knowledge, should be even greater than the benefits of reusing data because of its increased value and the problems associated with acquiring knowledge (Ignizio 1991). By capturing expert knowledge once and reusing it a number of times, better use can be made of this scarce resource and the knowledge acquisition bottleneck can effectively be reduced.

As mentioned, this work is interested in reusing knowledge within the same domain to support multiple activities and sees the benefits of reuse for humans as substantially greater than the material benefits outlined above. To justify this claim, the need to support a range of activities must be shown. Let us take a brief look at the some of the limitations of first and second generation expert systems (ES)[4], Human Computer Interaction (HCI) issues for KBS, the importance of control, the types of systems needed and the implications of situated cognition on the development of KBS.

1.1.1 The Limitations of First Generation Expert Systems

It has been shown that even where the knowledge is accurate and reliable, the expertise captured becomes an underutilised and wasted resource if the knowledge can not be accessed by the user in an acceptable way (Langlotz and Shortliffe 1983). An indepth study by Salle and Hunter (1990) considers the lack of attention to computer and user cooperation issues to be the main reason for the poor acceptance of ES technology by end-users. Cooperation includes the user interface but is much more. The user interface is concerned with usability, whereas the mode of interaction is concerned with usefulness (Rector 1989). The term used by Salle and Hunter is modality and is defined as: "referring to different forms the cooperation may have regardless of agents being

[4] The term expert system is an older term that has been mostly superceded in the literature by the term knowledge based system and is often used to avoid the stigma associated with the limitations of first generation ES. Sometimes the distinction is made that KBS encompass more than ES because the term may also refer to other computer systems that use knowledge components. The term knowledge based system and expert system will be used interchangeably in this thesis.

human or electronic" (De Greef, Breuker and De Jong 1988).[5] Collaboration is important because humans and computers have opposite abilities that complement each other. Cooperation aims at man-machine synergy.

Salle and Hunter (1990) consider that most KBS act as a prosthesis rather than support. Due to the subjective and changing nature of many domains such as medicine, they consider the prosthesis approach to be unsuitable. Users need more than prescriptive systems, they need to assess the answer through better explanation and query facilities. Expert Systems have been designed from the machine's viewpoint and lack the scope and robustness needed by users with a wide range of needs. They state "the KBS should be able to respond to a large variety of interactions, not only the commonly supported question" (Salle and Hunter 1990, p.6).

Kidd and Sharpe (1987) go so far as to say that many ES are not useful once in a commercial situation. More understanding of tasks and a theory of cooperative problem solving between man and machine is needed. They argue that current systems do not really solve the users' problem. Users don't only want to know 'what is the fault' or 'what is the remedy' they want to ask questions and negotiate a remedy. The system described in this thesis allows the user to ask a variety of questions by accessing the knowledge in different ways.

Having looked at first generation ES and their various limitations, we take a more up-to-date look at second generation ES and the issues that are relevant in delivering systems that are more user-centred.

1.1.2 The Limitations of Second Generation Expert Systems

To avoid or overcome the problems associated with first generation ES, there has been a general focus on building KBS at the knowledge level (Newell 1982) and interest in generic solutions to KBS development. These approaches to reuse are reviewed in the next chapter. In the next few subsections we consider the issues that are important to the usage and acceptance of KBS from a user perspective. The issues covered concern human-computer interaction, the importance of user control and a discussion of the types of systems users need. Then we look particularly at how a situated cognition view of knowledge affects the system requirements.

---

[5] In this thesis the term activity is used in preference to the word mode due to the possible negative connotations associated with modal systems (Tesler 1981) where for example, the user may become confused whether they are in browse or update mode.

1.1.3 Human Computer Interaction Issues for KBS

Stelzner and Williams (1988) emphasize the importance of the user interface for the acceptance of expert systems. One common interface in KBS is the consultation system, where the user is asked to answer certain questions and is given a recommendation at the end. Clancey (1992) considers the typical consultation mode of ES to be 'superficial' because the line of questioning approach does not allow the user to build a model that they can manipulate. Stelzner and Williams (1988) see a change from the traditional consultation style ES to what they term 'Expert Advisory Systems'. In these systems the human is involved in the loop and therefore it is necessary to support cognitive tasks.

Baroff et al (1988, p.100) consider question-answer style dialogues to be inappropriate for user control, visibility and user initiative. In fields like medicine and electrical engineering, graphical displays, such as visual causal diagrams (VCDs), are used for both the input of data and the display of results. This could be classed as a direct manipulation interface, which Baroff et al (1988) define as interaction via the mouse only, without the use of the keyboard, menus or commands. Direct manipulation reduces the knowledge requirements necessary to use the computer, letting the user concentrate on the task. Baroff et al (1988) argue that such a system provides a better fit with the user’s perception of the domain because they are usually easier to learn, less computer obtrusive and may allow 'what-if' exploration. Such a dynamic "world-of-action" interface provides instant response and update. They also believe that users should be given syntactic and semantic knowledge of computers and semantic knowledge of the domain.

Baroff et al (1988) see graphical user interfaces as the most appropriate for the capture of knowledge because they claim people think in pictures. They also recommend development of the user interface before entering domain rules so they can be validated on entry and the use of procedural control may be necessary to structure the knowledge acquisition process. This idea is exemplified in the work of Zacklad and Fontaine (1993) in their C-Kat system which assists users who are entering knowledge to choose the best conclusion from a range of numerous possibilities. C-Kat is discussed later in more detail in section 3.4.1. A final point made by Baroff et al (1988) is the importance of user involvement in the design of the interface since comprehension of the domain knowledge is dependent on the quality of the user interface.

The aspect of HCI focused on in this thesis is the usefulness of the system rather than its usability. Therefore no particular user interface style is recommended. In fact, this thesis emphasizes the situated nature of ES and that systems must be built to fit the cognitive models of the user while providing the sort of flexibility of interaction with the knowledge base that the user would have if the user was performing that activity without a computer. Due to the relevance of HCI issues for the acceptance of ES, further discussion can be found in section 1.2.1.

1.1.4 The Importance of Control

Since the advent of the PC, users have come of age and want to have control of the application (Pew and Rollins 1975, Schneiderman 1980). Users of experts systems do not want to relinquish decision making to a machine (Ignizio 1991, Langlotz and Shortliffe 1983). In the case where the user is an expert this concept is easily accepted. However various research efforts have shown that novices also wish to be in control of the decision situation. Kidd (1985) studied discussions between experts and novices where it was shown that the novice wanted to be able to propose solutions, evaluate remedies and have the remedy explained. The process was a form of negotiation. Kidd argues that the user approaches the activity with intentions, expectations and constraints. The ES needs to assist the user with exploring the 'remedy space' by a process of evaluation and comparison.

An interesting study by Gunnar (1978) revealed how fear in 12- and 13-month old infants could be alleviated if control of the situation was given and may be one of the reasons why control is so important in gaining acceptance of computer systems. Fear of change or the unknown are identified as causes of non-acceptance of computer systems (Flaaten et al 1989).

Hender and Lewis (1988) consider control to be a key issue in the design of user interfaces for expert systems. Experts want control and want to be able to validate and alter the knowledge. This involves understanding the reasoning process used by the system and the justification for the recommendation. They believe novices are more likely to accept the expertise offered without as much scrutiny but want a better interface to improve access to the knowledge.

1.1.5 The Type of Systems Needed

From the previous sections it can be seen that ES often do not match user requirements. A particular limitation mentioned was the consultation system style. Hender and Lewis (1988) argue that the user interface needs to be tailored to the environment in which it will be used because it will need to fit in with other software interfaces currently being used and may have different data sources. Possible interface alternatives to consultation include "second opinion" system styles which could be something like a critiquing or ‘what-if’ analysis activity. Hender and Lewis see a need for multiple interfaces to be provided with ES.

Stelzner and William (1988) argue that due to the interest in reuse there are new minimal requirements that must be supported by ES. For example, first generation ES were limited to narrow domains, but today with the development of corporate databases the scope of ES is increasing requiring the same knowledge to be used to cover more domains. Stelzner and Williams (1988) echo the thought expressed by many others (Chandrasekaran and Johnson 1993, Silverman 1992a and Swartout and Moore 1993) that deeper models are needed. They see the answer to be model-based reasoning which emphasizes organisation of the knowledge into a well-structured model separated from the inference engine. They see the user interface as the component that must handle the complexity of multiple uses and recommend the possible use of multiple interfaces one for each usage. Due to their emphasis on the importance of the user interface in enabling reuse they suggest that user interface tools need to be developed that may be reused to minimise the effort. This thesis has taken up the challenge to see if knowledge could be reused for different activities with only changes to the user interface required. The complete system developed is described in Chapter Seven.

Stelzner and Williams (1988) see that the difficulty is not in building commercial tools but in matching the way the system behaves related to the user’s cognitive tasks. They also argue that different user's will have different needs because their focus is different. A KE is interested in how the knowledge will be represented. The user is interested in the knowledge itself. While Stelzner and Williams have suggested an approach to HCI they admit their approach is still emerging since ES technology is still developing. Below are six features that they consider will be mandatory features available in a KBS:

1. access must be via a natural idiom such as diagrams, charts, networks, text, etc depending on the domain and the user;
2. a format that allows manipulation and experimentation such as spreadsheets;
3. immediate feedback;
4. recoverability which allows the user to play with scenarios without penalty such as loss of current state and actual data;
5. different levels of abstraction and granularity and
6. multiple interfaces for different users and situations such as an intelligent and interactive interface for the user and another interface for the programmer.

The author has previously explored another approach to improving the static nature of ES which was to integrate ES with another technology, Decision Support Systems (DSS), that do provide the features of flexibility and user control. DSS, such as spreadsheets and database packages, were launched in the market place in the same time period as ES, in the 1980s. In contrast to ES, DSS gained widespread acceptance even though these systems are often unreliable due to user bias (Phillipakis 1988). The acceptance of DSS is attributed to the high degree of user control, system flexibility and “user-friendliness” of these systems. “User-friendliness” is seen as a good match between the things the user sees and does with what they feel to be appropriate. In other words interaction is intuitive. This intuitiveness is better described as “transparency of interaction” by Winograd and Flores (1986, p.164).

Many have looked at the development of integrated ES/DSS models and systems (Jones 1986, Turban and Watkins 1986, Kuo 1988, Teng, Mirani and Sinha 1988, Lal 1989, Bannis 1990, Klein and Methlie 1990 and Marvin 1990). These approaches require creating new systems from scratch. Others consider adding a reasoning component to an existing DSS (Leigh and Doherty 1986, Turban and Watkins 1986, Turban 1993). The study by Richards (MAppSc 1994) took a new approach by adding DSS capability to an existing ES knowledge base (KB). The adapted system differed from a traditional ES, or DSS with an ES component, in the user interface, the variables that could be manipulated and the way the knowledge base was processed. Of the six points listed above by Stelzner and Williams (1988) all except for point 5 are supported by the adapted system. The changes made to the system are described in detail in Richards (1994) where a generic approach to adaptation was offered.

The research reported in this thesis continues the work of Richards (1994) and describes a system that offers all of these features including a means of uncovering and displaying different levels of abstraction and a wider range of interfaces. The work in this thesis also builds on the work by Lee and Compton (1995) who speculate that the application of knowledge to a new problem situation will be "one of the most productive approaches to knowledge sharing and reuse" and is a form of "value added transformation".

Many researchers have offered solutions to the problems described above, some of which have already been noted. In a similar vein to many of the above suggestions, Kidd and Sharpe (1987) see proper analysis of the problem as the key to providing KBS that address the users needs. They propose beginning with an initial description of task requirements involving cooperative problem formulation, cooperative generation of alternative solutions and cooperative explanation. The inputs include domain knowledge models, mapping between the model and the machine representation, task specifications, logic descriptions and a framework for performing task and domain analysis. They argue, in addition, if the domain is weak or has no underlying theory then AI will not be able to solve the problem. It can be seen that the solution they outline is exceedingly complex and of limited applicability.

1.2 The Situated Nature of Expert Knowledge and Action[6]

In contrast to the complex approach offered above by Kidd and Sharpe (1987), the solution to the limitations of KBS offered in this thesis is a simple approach which is founded on a completely different premise. RDR were created in the belief that knowledge is not an artifact which only needs to be properly defined in order to be used. Knowledge is not some 'stuff' that can be mined from the heads of experts as was implied by the physical symbols hypothesis (Newell and Simon 1976). Instead the recommendation given by an expert depends on the context in which it is given and does not consist of a description of the expert's thought processes but is a justification of why that recommendation was made (Compton and Jansen 1990).

The emphasis of most ES approaches on building terminological KBS as a prerequisite for the capturing of expertise is not consistent with the way that experts behave: "If we take Heidegger and Maturana seriously, we see that experts do not need to have formalised representations in order to act" (Winograd and Flores 1986, p. 99) and "reflection and abstraction are important phenomena, but are not the basis for everyday action" (Winograd and Flores 1986, p. 97). The act of assigning a conclusion to a pathology report and picking the salient features in the case is an example of what Heidegger (1962) terms *throwness,* which refers to the common human experience of acting in response to being thrown into a situation rather than reflecting first and then acting. Winograd and Flores argue it is the ability to act spontaneously that makes an expert an expert - "The essense of our intelligence is our *throwness* not our reflection" (Winograd and Flores 1986, p.99). From this discussion it appears that most expert action is reflexive. Popper also (1963) implies the impossibility of deduction in an ultimate sense. Since conceptual models are unreliable and difficult to capture an approach which avoids the expert having to articulate their thought processes may be preferrable. Knowledge acquisition, maintenance and inferencing using RDR are simple activities which are provided in reflexive modes that do not require introspection.

Using the first and third person interpretation analogy used by Clancey (1993) the aim is to build a system which performs KA in the first person. Clancey describes the first person as primary learning that is "inventive", requiring reperceiving, and reshaping previous experiences. Primary learning involves automatic recoordination and is a reflexive type of action. An expert viewing a pathology report, assigning a conclusion and picking the salient features in the case to form the rules are viewed as reflexive actions. On the other hand the explanation of why a conclusion and features were

[6] Much of this section is taken from a paper accepted by the International Journal on Human Computer Studies to appear in a Special Issue on "The Challenge of Situated Cognition for Symbolic KBS"

chosen is a reflective action that requires third person interpretation. This is analogous to other situations, such as in social interaction, where models may not be necessary prerequisites but use of them in retrospect is often beneficial for understanding what has occurred (Clancey 1993).

One of the problems with capturing and using expertise is that little is known about how experts arrive at a conclusion. Progress in understanding human cognition has been slow due to: dense information, complexity of the world and the impossibility of observing all the relevant aspects of human cognition (Norman 1993). Situated cognition necessitates interaction with the real world which Vera and Simon (1993) characterise as having real-time involvement; immediate responses to external stimuli and complexity. But the situated view is more than just taking into account interaction between the individual's inner state and the external environment and trying to record all the influencing factors. The problem is that thinking and acting are not two separate activities but thinking is itself an act that modifies further action (Clancey, personal communication). Thus, context in a situated sense is more than just the environment but occurs at a conceptual level that exists within a social setting involving such things as activities, participation, roles, contribution and norms (Clancey, personal communication). Given the complexity of human cognition it may seem inconsistent to advocate simpler techniques. However, by capturing a simple model of the knowledge based on expert behaviour is one way of overcoming or at least avoiding the difficulties associated with the development of high-level models. This does not mean breaking the task down into smaller components in the way that many other approaches have done, such as Generic Task Framework (Chandrasekaran 1986), KADS and CommonKADS (Schreiber, Weilinga and Breuker 1993), Role-Limiting Methods (McDermott 1988) and Components of Expertise and the Componential Methodology (Steels 1993). In the next chapter these approaches and the problems associated with them are considered. First, let us look at the implications of situated cognition for HCI, KBS development, maintenance, verification and validation.

1.2.1 The Implications of Situated Cognition for HCI

The need for complex modeling of the domain before KA can take place has resulted in a preoccupation with knowledge elicitation rather than user requirements (Salle and Hunter 1990). This has resulted in systems designed for knowledge engineers rather than experts or end-users. By simplifying the KA technique and making the user responsible for KA and maintenance we are able to focus more on user requirements elicitation rather than knowledge elicitation.

The ability to capture knowledge for one purpose, such as consultation, and reuse it for other purposes, such as teaching, critiquing or 'what-if' analysis, is more akin to the situated view that knowledge is not a fixed structure that is retrieved and applied from fixed storage positions in memory as many AI programs presume but that knowledge will need to be reperceived and reconstructed for each situation. If we take the view that "perceiving, behaving and learning are one process (Rosenfield 1988) then it makes sense to build a system that does not separate these processes but allows the user to perform KA, find concepts, perform inferencing, learn about and explore the knowledge, and so on, all within the one session. Figure 1.1 shows how the user may choose which activity to perform based on the situation in which they find themselves.

Situation	**Activity**
Develop, Fix KBS	KA, Maintenance
Get advice	Consultation
Evaluate own solution	Critiquing
Consider options	What-if Analysis
Understand advice	Explanation
Learn about domain	Tutoring
Find causes	Causal Modelling
Compare KBS	Requirements Engineering

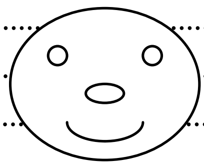
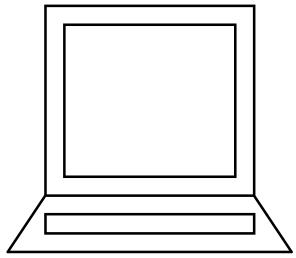



Figure 1.1 The various situations in which a user may find themselves are shown on the left. To solve their current problem they use the features of the system that support the necessary activity on the right. The choice of activity is situation and user-driven. The user can switch between activities as their needs change.

In agreement with Winograd and Flores (1986) and Suchman (1987), human computer interface methods should be reevaluated and changed to support the lessons of situated cognition. This requires understanding the context in which the system is situated (Winograd and Flores 1986). Vera and Simon describe what they term "the soft form of investigation of SA [that] builds AI systems that incorporate principles of representing objects functionally and interacting in a direct and unmediated way" (1993, p. 11). The system described later has taken this form by using an object oriented interface environment and has removed the need for *a priori* analysis of the domain or the use of a knowledge engineer in the KA process.

Not only do we need to study how humans use computers but also their uses and what problems, or "breakdowns" occur (Winograd and Flores 1986). The underutilisation of ES described by Langlotz and Shortliffe (1983) is viewed as a breakdown that requires attention. The following quote describes the features of and alternative to a breakdown:

> *"A system that provides a limited imitation of human facilities will intrude with apparently irregular and incomprehensible breakdowns. On the other hand, we can create tools that are designed to make the maximal use of human perception and understanding without projecting human capabilities onto the computer" (Winograd and Flores 1987, p.137).*

This last point is note worthy. By creating systems that are more usable by humans we are not saying that we are making systems that are more like humans (Winograd and Flores 1987). This is not necessary.

Further to detection, it is necessary to fix or adapt the breakdown. "The ability to resolve breakdowns adaptively is a basic human cognitive capacity" (Vera and Simon, 1993, p.15). The ability of humans to adapt to different situations is considered "perceptive and intelligent" (Clancey 1991b, p.256). Particularly when seeking to reuse knowledge to answer different types of questions, it may be necessary to manipulate the knowledge so that new knowledge can be derived from it. Stroulia and Goel (1994) suggest two approaches to building systems which adapt to the ever changing needs of users and the environment. In the first approach the developer needs to predetermine what the changes may be and try to allow for them in system development. The second approach is to enable "the system to reflect on its own problem solving process" (Stroulia and Goel 1994, p. 395). They have used an adapted version of structure-behaviour-function and Chandraskaran and Johnson's (1993) task-structure framework to design what they term *reflective, self-adaptive problem solvers*. Structure-behaviour-function is used to analyse and describe the problem, which assists in defining the appropriate knowledge and how it is used. Failure driven learning is used to detect when a conflict exists and how to resolve it. Reflection is used to analyse why the problem could not be solved with the plan chosen and to reason about how the plan needs to be modified. The generation of new knowledge from existing knowledge, in the form of new plans from old plans, can be seen as a reasoning task where new knowledge is being derived from existing knowledge.

There is a considerable body of research into adaptive systems. This covers problem-solvers and learning systems, such as Soar (Laird, Newell and Rosenbloom 1987) and Prodigy (Carbonell Knoblock and Minton 1989) and reflective learning systems such as Teiresias (Davis 1977), MetaAqua (Ram and Cox 1994) and Autognostic (Stroulia and Goel 1994). Some of these systems are able to detect a success/failure, assign credit/blame and revise beliefs of/repair the system. While not achieved by all of these systems they all share a common goal, perhaps with the exception of TEIRESIAS, to perform learning independent of human intervention. Since changes in circumstance can not be fully anticipated or described it is difficult to believe that systems can be built which automatically adapt to changing environments. This is why this study has taken the approach of building user-adaptable systems rather than self-adapting systems. The system reported in this thesis learns by helping the user to learn about their domain, reflect on the knowledge and incorporate any new insights into the system. The system offers a wide range of activities and leaves the user to select the tools appropriate for their purpose. With Lave (1988) this thesis would like to move from a "learning transfer" attitude and argues that many approaches to explanation or tutoring do not take the situated nature of learning into account. Allowing the user to decide how to use the system is seen as a better alternative to creating systems that anticipate what the human wants to do, such as systems that include user models (Cawsey 1993, Swartout and Moore 1993).

The simplicity of KA in the system described aims at minimising system breakdowns by providing an intuitive environment. As described later in Chapter Three, the expert is required to look at a case, decide whether the conclusion is correct and if not assign a new conclusion and pick which features in the case warrant the new conclusion. The user is further assisted in their decision making by being shown a list of differences between the current case and the case associated with the rule that incorrectly fired. The system supports direct and intuitive interaction in a wide range of user-selected activities which matches the users desire for system flexibility, system usability and user control.

1.2.2 The Implications of Situated Cognition for System Development, Maintenance, Verification and Validation

Situated cognition offers some exciting as well as some potentially discouraging implications for the development of KBS. Much knowledge is non-verbal and can not be realistically "inventoried" or shared (Clancey 1993, p.109). The immediacy of much human action, the ever-changing nature of contexts, the uniqueness of each act and the complexity of factors that affect an action may lead us to abandon the endeavour to build useful KBS or at least to ignore the need for system verification and validation. This perhaps accounts for the research emphasis on building KBS with little thought for how to maintain these systems once they are in operation (Menzies and Compton 1995, Soloway, Bechant and Jensen 1987). Most approaches to verification and validation support a system development life cycle that assumes the system will be built, verified and validated before it is put into routine use (Kang, Gambetta and Compton 1996). Part of the reason why verification and validation (V&V) approaches focus on pre-implementation is due to the problem that in a typical rule-based system each time a modification is made there may be unwanted side-effects that occur (Grossner et al

1993). It is common in the literature to read of systems that were literally abandoned once the rules became too hard to maintain (Mittal, Bobrow and de Kleer 1988).

A situated view of knowledge places great emphasis on incremental techniques that allow change. RDR have been developed with maintenance viewed as an essential activity and offers online and incremental validation performed by the user.

1.2.3 Summary of the Impact of Situated Cognition

From the discussion above, situated cognition imposes some further requirements on the type of KBS that users need. As noted in 1.1.4, control was an important concern of users which is reinforced from a situated stance since the user will need to be able to adapt their usage of the system to suit their varying circumstances. Taking into account the frame of reference problem (Clancey 1991a) it is also necessary that different users are able to control and adapt the same system according to their different requirements, viewpoints and personal preferences. The need for users to have primary control of the system also extends to the activities of KA, maintenance and validation. In most KBS research, these three activities are the responsibility of the knowledge engineer and the user is a third party. In the RDR approach used in this thesis these activities are performed by the user with no or minimal intervention of a knowledge engineer. The use of contextual knowledge that is grounded in real world examples is another feature that a situated view and RDR support.

Having described the underlying motivation for this thesis and determined the types of KBS that users need we go on to look at the aims of this thesis towards the development of such a system.

1.3 The Aims of this Thesis

It was the goal of this thesis to develop an approach to building KBS that supported the user in a wide range of activities. In keeping with a situated view of human action, the user could determine what activity was appropriate to their current needs. This meant that knowledge that may have been captured for such purposes as inferencing or consultation would need to be reused to support other activities such critiquing or tutoring.

To achieve this a KA technique needed to be found or developed that facilitated capture of knowledge that could be reused in many ways. As is discussed in section 2.3 this involves acquiring the right kind of knowledge, at appropriate levels of abstraction and depth, the context within which the knowledge applies and enough knowledge to cover these different uses. To avoid the limitations of first generation ES and to cope with the evolving nature of expertise, a simple KA method and a reliable maintenance strategy were also highly desirable. To give users control and ownership of the system it was preferable to find a KA technique that could be performed by the user.

It was found that RDR satisfied all of the above criteria except for capturing knowledge at different levels of abstraction. The ability of RDR to support the user in performing KA, maintenance and inferencing, which are referred to as reflexive modes, had already been demonstrated in other RDR research. However, two problems with RDR were noted: the repetition problem and the lack of higher-level modelling. The repetition problem was of concern to reuse because it was expected that a cleaner representation would result in better explanations. Of greater concern was the development of a method for finding the higher-level concepts from the primitive concepts in an RDR rule-base that could be used to support the explanation or reflective modes of interaction. It was the aim of this thesis to address these two limitations.

In addition to these initial goals, it became apparent that the framework that allowed hierarchical conceptual models to be developed from an RDR KBS could be useful for the group activity of comparing KBS and detecting and resolving inevitable conflicts. This added a further goal to be pursued which was to extend the knowledge engineering framework to support requirements engineering (RE). Each of these goals are described further in this thesis together with the work that was conducted in the pursuit of them. In the next section an overview of the thesis structure is given which indicates in what part of this thesis the discussion relating to the various goals can be found.

1.4 The Structure of this Thesis

In this chapter, the need for reuse in general and *activity-reuse* in particular has been considered. In **Chapter Two** the various approaches to KBS reuse are discussed, followed in **Chapter Three** with a description of RDR and related theory. To allow RDR KBS to support *activity-reuse* it was necessary to address two shortcomings of RDR: the repetition of knowledge and the lack of terminological models. This led to the investigation of the use of rough set theory (Pawlak 1991) for the removal of repetition, which is described in **Chapter Four** and the use of formal concept analysis (Wille 1982) for uncovering a T-box from an RDR-initialised A-Box, which is presented in **Chapter Five**. As a further, and significant, outgrowth of the formal concept analysis work is a framework to support the group activity of requirements engineering that is discussed in **Chapter Six**. In that chapter it is shown that RE can be viewed as a maintenance or validation activity over multiple KBS. In **Chapter Seven** we review the different activities and how they are performed in the MCRDR/FCA system developed.

Chapter Eight offers an evaluation of the MCRDR/FCA tool and its general applicability to rule-based systems. The thesis concludes in **Chapter Nine** with a discussion of the contribution of this thesis. We turn now to a review of the knowledge reuse literature.

Chapter 2

2 Approaches to Knowledge Reuse

In Chapter One the ability to support users in a wide range of activities was seen as an important feature of KBS. This required knowledge to be reused and was called *activity-reuse*. In the previous chapter we also looked at the benefits of reuse in general and in this chapter we review current approaches to knowledge reuse which covers a number of types of reuse. These approaches not only differ in the solution they offer but also in the types of reuse they address. The main issues identified from the current knowledge reuse literature are given in Section 2.2. The focus on *activity-reuse* in this thesis is unusual, so a summary of what issues are considered relevant to this study is given in Section 2.3 at the end of this chapter.

2.1 Knowledge Reuse Research

Due to the many perceived benefits that can result from reuse of a resource there is currently much interest in the area of knowledge sharing and reuse. Weilinga et al (1993) consider reuse to be a common research goal of many current projects. Van de Velde (1993) terms ES with reusable components, second generation ES. Sharing and reuse was the theme at the 1995 Banff Conference on Knowledge Acquisition. Many believe that the building of large KBS will only be possible if efforts are combined (Neches et al 1991). Most of the reuse research discussed in this chapter is concerned with reuse of problem solving methods or ontologies. Section 2.1.2 introduces these research efforts. However, there is a body of research, known as the MYCIN experiments, which forms a notable exception and has been concerned with reusing knowledge for different purposes such as teaching or KA. This research is well known in the KBS community and has led the field in many regards including in the area of knowledge reuse.

2.1.1 The Mycin Experiments

The most well reported reuse of knowledge concerns the MYCIN family of ES (Buchanan and Shorltiffe 1984). MYCIN was originally designed to assist physicians in the diagnoisis and treatment of meningitus and bacterial infections. Its purpose was to provide a research vehicle to learn about and demonstrate the building of expert systems. The separation of the diagnostic process from the knowledge of the disease into EMYCIN (Empty MYCIN) was a demonstration of how problem solving knowledge could be reused with different domain knowledge. PUFF (Harmon and King 1985) is an ES which handles pulmonary disorders and is an example of a system using the EMYCIN shell that went into routine use. Numerous other adaptations of the original MYCIN system have been developed, these include:

- the reuse of MYCIN in NEOMYCIN to test that a knowledge representation (KR) could be used for explanation as well as problem solving.
- the abstraction of NEOMYCIN into HERACLES (Clancey and Bock 1985) for use as a heuristic classification shell. Clancey distinguishes HERACLES-DX as the first task-specific ES design (Clancey 1992, p.16).
- the development of TEIRESIAS (Davis 1977) to provide a debugging dialogue to assist the knowledge acquisition process for MYCIN. It did this by presenting a rule trace of its reasoning to the expert when a wrong conclusion was given. The expert could then determine if the rule was incorrect, the application of the rule was inappropriate or whether another rule could be specified to negate the incorrect conclusion. The addition of modelling tools to RDR to asisst with KA, described in Section 7.3.1.1 has a similar goal to that sought by TERESIAS. The RDR approach and TEIRESIAS both pass the responsibility for the system's learning onto the user, as opposed to many learning systems that seek to be self-adaptive. In the case of the RDR system described in this thesis, the emphasis on the role of the user in the learning process is intentional. In keeping with the system criteria discussed in sections 1.1.5 - 1.2.3 user control and system flexibility are seen as essential ingredients in gaining acceptance of computer systems.
- GUIDON-MANAGE is a tutoring program that uses subtasks to assist students to learn about diagnostic strategies using the knowledge in NEOMYCIN. In effect it acts as a simulation of using NEOMYCIN (Clancey, 1992). Clancey notes that when NEOMYCIN was reused in this way the subtask translations that were too abstract for consultation purposes became useful as hints for describing problem solving strategy.
- GUIDON-WATCH is another reworking of NEOMYCIN in a graphic form for instruction and debugging purposes. The system presents a causal model to the user and is a useful method of representation. This led Clancey to the concept of the situation specific model (SSM) which was also based on work in ABEL (Patil, Szolovits and Schwartz 1984). The SSM is the case model that results when a general model of problem solving is applied in a particular context. The SSM is manipulated by subtasks which are groups of metarules that perform part of a task. They can be considered as model construction operators (Clancey 1992), responsible for constructing nodes and links in the SSM. If they recur they should be abstracted for reuse.
- GUIDON-DEBUG was constructed to assist the detection of holes in the SSM.

The GUIDON systems mentioned above were developed as part of Clancey's PhD thesis to test the feasibility of the reuse of the MYCIN KB for tutorial purposes. Since then Clancey has had much to say on the topic of knowledge reuse. In his landmark paper on model construction operators (Clancey 1992) he claims "that describing knowledge acquisition tools in terms of model construction operators facilitates collecting and sharing knowledge bases and representation languages" (Clancey 1992, p.7).

From an analysis of numerous systems, including the MYCIN-related ones mentioned, Clancey lays to rest the fallacy that an ES consists of a knowledge base and inference engine alone. The third component is the inference procedure which provides the control or strategic knowledge. He says: "it is the ability to store facts about inference procedure constructs (subtasks, metarules, premise relations[7]) that enables it to be used for multiple purposes" (Clancey 1992, p.32). These procedures offer reusability of the abstract control knowledge. By employing different procedures the knowledge can be used in multiple ways:

> *Operating on a knowledge base in different ways (e.g. for explanation, tutoring, or compilation) requires different procedures with different relations, and hence a reclassification of expressions in the knowledge base" (Clancey 1992, p.10).*

By classifying process constructs into different groups the software can be reused by different interpreting procedures by offering different views of the model and reasoning processes. This involves establishing a new structure and new relations, which act as filters on what is included in the SSM. It is necessary to focus on the relationships between types of systems, process model structures, SSM structures and inference procedures.

It was also found in this thesis that abstraction, different views and understanding of the relationships between the primitive concepts in the KB (the rule premises and conclusions) were important to enable reuse. FCA is the means used in this thesis to automatically generate the relationships between concepts, to provide the structure of the knowledge and to provide a number of views of the knowledge. A major difference between the FCA work and Clancey's work is the absence in the former of metarules or procedural/control knowledge in supporting the different uses. The procedures described by Clancey in terms of subtasks and metarules are such things as Forward-Reason Generate-Question, Process-Finding Apply-Evidence-Rules, Explore-and-Refine, Process-Hypothesis. Instead of using these type of procedures to control manipulation of the domain knowledge the one inferencing mechanism is used on the low-level assertions and formal concept analysis is used to find the higher-level abstractions and relationships that exist.

---

[7] Subtasks are collections of metarules, a metarule is a rule that controls other rules and premise relations are the relationships between rule conditions.

Clancey sees other research into generalising as being relevant to reuse. Clancey equates Chandrasekaran's Generic Tasks to his definition of subtasks. What Clancey terms a complete inference procedure he equates to McDermott's role-limiting methods. Another point Clancey (1992) makes is that through the use of modelling tools and a system-as-a-model viewpoint (Newell 1982), an ES should be able to be reused for a range of different purposes, rather than needing to build different architectures for each problem situation. This thesis is interested in building a multipurpose system but, unlike most KBS approaches, the starting point is the development of an assertional KBS based on cases and observed expert behaviour rather than a terminological KBS which requires building high level models of the knowledge first. Before commenting further on the two contrasting approaches, let us look at the modelling perspective and generic approaches to reuse.

2.1.2 The Knowledge Level - Reuse of Problem Solving Methods and Ontologies

Like Clancey, many researchers stress the importance of the knowledge level (KL) perspective for sharing and reuse. The knowledge level was first introduced by Newell (1982), although, as Newell points out other research at that time implicitly acknowledged the KL viewpoint. An example is Marr (1982) who proposed a view using four levels that emphasized separation of implementation and computational decisions and analysis of expertise according to the tasks performed. Newell (1982) says knowledge has a competence-like structure and gives the following definition of knowledge as: “whatever can be ascribed to an agent, such that its behaviour can be computed according to the principle of rationality” (Newell 1982, p. 105). He argues that knowledge is described functionally not structurally. We need to know how knowledge works, why we need it and why it is effective to be able to use it. This is not an easy task. He points out, as AI researchers we are interested in generating knowledge intelligently. Simple procedures only generate uninteresting knowledge, a thought also expressed by Schank and Leake (1989), and will not be adequate for the second generation of ES. In reply to the debate over the best knowledge representation to use, he argues that logic is useful for modelling the knowledge level but it is not necessarily the choice for the symbolic level.

Prior to the introduction of the knowledge level, research had focused on the representation and manipulation of knowledge using symbols, also known as the physical symbols hypothesis (Newell and Simons 1976). The belief was that knowledge could be transferred or *mined* from an expert because both humans and machines required physical symbol systems to represent knowledge. Newells’ main idea is that in addition to the various layers that make up a computer system there is the knowledge level which sits above the symbolic top layer and has resulted in the abandonment of the knowledge-transfer attitude by most AI researchers. Van de Velde (1993) sees that separation of the knowledge (KL-Model) and the knowledge representation (symbolic level) is essential for reuse as it allows each aspect to be dealt with individually.

Van de Velde (1993) describes the following approaches which have been built on Newell's knowledge level model:

- Generic Task Framework (Chandrasekaran 1986)
- KADS and CommonKADS (Breuker 1994, Weilinga et al 1993)
- Role-Limiting Methods (McDermott 1988)
- Components of Expertise and the Componential Methodology (Steels 1993)
- Protege and Protege II (Puerta et al 1992)
- KIF and Ontolingua (Patil et al 1992)

Each one of these approaches are discussed below. Van de Velde describes these approaches as KL-models since they have more structure and are therefore less flexible than Newell's model (Van de Velde 1993). “A KL-model is a structure that is imposed on knowledge when it is being put to use in a class of problem situation” (Van De Velde 1993, p. 215).

While each of the approaches mentioned above are different, Van de Velde (1993, p.1218) considers three concepts to be generally included as part of a KL-model. These are the domain model, the task model and the problem solving method (PSM). Van de Velde argues that the task features and the domain model will affect the type of KL-model chosen and therefore it is important for more work to be done on how to select a KL-model. Selection of the right KL-model will affect the robustness of the system. The brittleness problem frequently referred to occurs when we build an ES to solve a particular set of cases and then try to solve an unseen case not previously covered by the knowledge base. To reduce some of the brittleness of ES, Van de Velde recommends matching the problem characteristics with the most suitable problem solving method.

Weilinga *et al* (1993) discuss the unification of the various knowledge modelling approaches. He particularly discusses CommonKADS, which combines the components of expertise and KADS approach. The suite of models in the CommonKADS methodology includes the expertise model, task model, communication, design, agent and organisation model. Weilinga sees human expertise as a separate model which may be useful input for the KBS expertise model but it is not the same, which matches Clancey’s (1993) criticism of descriptive models and is a view shared by Rademakers and VanWenkelhuysen (1993). In the same vein as Clancey’s argument for abstraction:

> *"The KL is the appropriate level for modelling the competence of KBS. It calls for the description of problem solving behaviour at a conceptual level that is independent from representation and implementation decisions" Weilinga 1993, p. 303.*

Weilinga adopts the role limiting principle introduced by McDermott (1988). Role limiting refers to the way an agent imposes structure on the knowledge when faced with a particular task. Role limiting enables us to describe the 'how' of an agent in knowledge level terms. The role-limiting method requires the knowledge level to be broken into three categories - domain knowledge, task knowledge and inference knowledge. The latter may be equated to Van de Velde's problem solving method category. There is intra and inter-category structure. Domain knowledge includes the domain ontology and domain model. The domain model has its own schema and a more generic and abstract ontology which is different to the domain ontology. The model schema describes the structure of the entities in the domain model. Within these categories further generic types can be found. Problem solving knowledge is not a new category but a metalevel. It includes the PSM and strategic knowledge. Weilinga gives the following definition: a problem solving method "specifies a number of possible actions and some way of determining how these actions are to be ordered in time" (Weilinga et al 1993, p. 315). For a detailed description of each level see Weilinga et al (1993). The relevance of role-limiting for this discussion on reuse is that the idea of a task limiting and constraining the KA process means that the tasks handled by one role-limiting method will not be able to be reused by another role-limiting method so that reuse is only possible over the same task but not between tasks. As mentioned in the next chapter, recent RDR research questions whether it is possible to use the same problem solving method for different tasks such as classification and configuration and thus suggests that the role limiting method may indeed be unnecessarily limiting. Alternatively, it may be that classification is a specialisation of configuration and in this way they both share a similar role.

Weilinga et al (1993) distinguishes between what Newell described as the principle of differentiated rationality where we assume an agent will behave rationally to satisfy a goal and Van de Velde's two-step rationality where the knowledge configuration and application are also considered. Weilinga finds the latter to be a more useful description of how an agent will behave.

Weilinga et al (1993) describe a CommonKADS library of generic components which is being built to facilitate reuse. Generic components may be PSMs, instantiated PSMs, generic tasks, inference structures and generic domain models. Weilinga et al remark that some components may be too general (that is, the grain size is too coarse) and need adaptation. They are also developing a formal modelling language to assist reuse.

Other work building on the KADS structure uses the MoMo methodology (Hemmann 1993a and 1993b). Hemmann has a stronger commitment to reuse and sees that there are two separate types of knowledge that can be reused: domain knowledge and the problem solving knowledge. Unlike KADS, MoMo uses abstractions of PSMs, known as Frameworks of Problem Solving (FPS), which are said to be useful for knowledge acquisition, explanation generation, teaching, combining different implementations and the creation of repositories of different model types. He claims MoMo outperforms KADS in the area of explanation because of binding mechanisms which allow the user to be more involved in the inference process. Hemmann finds that KADS is "well suited to build reusable models of knowledge" (Hemmann 1992, p 1) due to the four layer framework and the widespread acceptance of KADS. While he argues that KADS builds models that are reusable, MoMo better supports sharing. MoMo is an operationalisation of the KADS concepts. In contrast to KADS, Hemmann (1992) has a greater emphasis on the separation of the domain layer from the rest, where a MoMo environment is made up of an interpretation model and a domain model. He equates the interpretation model to a problem solving method. Domain models can be used by different PSMs and a PSM could be used for different domain models. There is an emphasis on clear and clean links between the domain-specific and generic PSM knowledge. The joining of domain-specific and PSM knowledge is called a view (Hemmann 1993b).

Another approach is generic tasks (Chandrasekaran 1986). A generic task may be described in terms of the I/O description of the task, the method/strategy and knowledge. Generic task shells have been developed for classification (CSRL) and for design (DSPL). Generic tasks assist the process of knowledge acquisition by providing a vocabulary for seeking knowledge and guided organisation and use of the knowledge acquired. When domain knowledge is added the generic task shell works as a problem solver for the task in that domain.

Due to various problems such as chunkiness and inflexibility, Chandrasekaran has moved away from generic tasks to the task structure perspective. This structure is a tree of tasks, methods, operators, subtasks and inferences. Search-control knowledge and sub-task operators are kept separate to allow dynamic selection of operator sequence which offers wider applicability of the method to a range of problems. Methods are ways of organising knowledge to achieve a task. Methods are generic components that combine to form an organised collection of inferences for generic types of goals. Knowledge sharing continues to be a research concern for Chandrasekaran and Johnson (1993). They have come to the conclusion that knowledge can not be separated from its use and that different usages require different representations. This thesis questions this conclusion by demonstrating that knowledge can be used in different ways for different activities using the same knowledge representation. It is acknowledged, however, that the restructuring of the knowledge into the FCA subsumption hierarchy is another form of knowledge representation but since it is generated automatically we could view the multiple classification RDR and FCA representations as a general KR pair that together support a wide range of activities. The ability of the same RDR structure to support configuration and classification tasks also suggests, but is not substantiated in this thesis, that the same knowledge representation can be used to support multiple types of tasks.

In considering the use of generic tools, Van de Velde (1993) identifies the need to determine the type of problem to be solved. Breuker (1994) defines a problem as a discrepancy between the current and norm state (as opposed to the traditional view of a goal state). A solution is not a simple answer to the problem but a probably-abstract complex object comprised of a case model and an argument structure. The problem space can be described as <P,S,solution> where (P) is the set of problems, (S) is the set of solutions and (solution) is the set of relationships between P and S. Breuker points out that a problem is not the same as a task. In a task the problem solving methods are known and tasks can be repeated. Tasks are more structured, well-defined and general. Problems are deviations or complications. Tasks include three steps:- problem identification; problem definition; and task specification.

Breuker identifies eight problem types:- Modelling, Design, Planning, Assignment (scheduling and configuration), Prediction, Monitoring, Assessment and Diagnosis. These eight fall into two broad categories which he calls synthesis and analysis, more commonly known as construction and classification, respectively.

Breuker (1994) advocates using minimal solutions with generic conclusions. Rather than using a taxonomy to analyse tasks he prefers the idea of dependencies. In real life, task output (which is often a conclusion) from one problem is input to solve the next problem. For example a planning task may involve a chain of problems such as:

planning/assignment → prediction → monitor → diagnosis → assignment

Dependencies can be identified by distinguishing the problem from the task. Breuker suggests various common dependencies that are found. Breuker’s main argument is that within one application there is a suite of problem types and dependencies. He suggests that the set of problem types will be smaller and more similar than the set of tasks. He sees the development of a different problem solving method for each problem type as an ideal future goal.

Protege II (Puerta et al 1992) offers a similar approach by allowing knowledge engineers to select problem-solving methods from a library of domain-independent reusable resources. Protege-I was limited to a single method. The suite of tools assist in the definition of ontologies using MAITRE a frame-based editor for building MODEL structures, the building of KA interfaces using DASH and the development of run-time systems. The system uses the CLIPS production language and runs on the NeXT platform (Rothenfluh et al 1994). A domain ontology is used to map the library method to the domain. Ontologies and methods may be indexed and searched so that they can be edited and combined in numerous ways. Control knowledge is contained in the problem solving method and joined to the domain via the domain ontology that controls the reasoning process. This process of conceptualisation of the domain at a higher level and mapping to the problem solving method is one example of how the KL-model can be implemented.

Puerta et al (1992) describe Mecano, a component of PROTEGE-II, which reuses domain ontologies to build different knowledge acquisition interfaces. They argue that research has concentrated on the reuse of expert system problem solving methods and ignored the problem of having to build new interfaces for each application for knowledge acquisition. DASH uses a nodes-and-links graph depicting the dialog structure and the relationships defined in the domain ontology to infer the knowledge requirements, dialogue and presentation and layout. The benefit of this is that the relationships used in the KA tool is the same as those used in the problem solving method used for reasoning and results in reuse of the domain ontology.

The word ontology has arisen in a number of places and before going further it is worthwhile to define what is meant by an ontology as opposed to a knowledge base. The word ontology comes from the Greek words *ontos* and *logos* which mean, being and word, respectively, and therefore means that is it concerned with the study and description of being or the basic categories of existence (Sowa 1991). Although there are a number of variations, a common understanding of the word is a set of terms by which the domain or task can be described which includes definitions of the terms and the relationships between them. The definition of an ontology given by Gomez-Perez depended on whether the ontology was for reuse or sharing. If an ontology is designed for reuse it reduces the effort that would be required if it had to be built from scratch. If the purpose is sharing then an ontology acts as a common vocabulary and means of sharing one's viewpoint. Gomez-Perez (1994) offers the following differences and similarities between an ontology and a KB. The language used for an ontology needs to be "expressive, declarative, portable, domain-independent, semantically well-defined (and) machine-readable." (Gomez-Perez 1994, p.2). This language is not necessarily the target language of the application. The strategic and object knowledge should be separate in an ontology, although it is difficult to represent strategic knowledge declaratively. An ontology tends to be domain-independent and contains more abstract definitions than a KB. To enable reuse an ontology needs to be better structured and defined than a KB. "A KB is the knowledge module of a KBS. It contains abstract and specific knowledge of a particular subject in a machine-readable format" (Gomez-Perez 1994, p.2). Typically ontologies do not include reasoning methods although Guarino and Giaretta's (1995) definition includes predicates and functions that support operation on a logical language. Increasingly researchers such as van Heijst, Schreiber and Weilinga (1997) and Martinez-Bejar, Benjamins and Martin-Rubio (1997) use ontologies for the specification of knowledge-level conceptualisations. Thus the ontology provides a constrained structure on which the domain knowledge is based. Since ontologies are removed from the actual application in which they will be used it is very difficult to evaluate them. An exception is the work described near the end of this section (Ikeda et al 1996) where the conceptual model can be executed at the symbol level which allows ontological commitment to be validated. This question of evaluation is discussed later under section 2.1.3.1 which covers the problems with generic solutions.

While there is little consensus on a single definition of an ontology the motivation for the development of an ontology and the type of ontology provided by the FCA line diagrams in this thesis is in keeping with the general "aim of ontology research [which] is to explicitly represent the meaning of concepts and the relation among them" and the notion that an "ontology, in general, is an agreement between users and systems" (Ikeda et al 1996).

Returning to a review of KL-model approaches, Steels (1993) advocates breaking the KB into components, however his criteria for grouping is not knowledge content but tasks, domain models and methods. This he calls the *componential framework*. The task view looks at what needs to be achieved, the model views looks at what type of models are being built and utilised. The method view considers how to use the knowledge to accomplish the task. A brief description of models and methods and how they relate to tasks are given below. Refer to (Steels 1993, pp.276-281) for more detail.

A model that is fixed is known as a domain model. A dynamic model is a case. A domain model is used to solve a specific case model. He refers to ontological models as special cases of domain models that "constrain the vocabularies that can be used in other models" (Steels 1993, p.276). It is due to this inflexibility that Schmalhofer, Aitken and Bourne (1994) criticise the ontological approach to reuse and their solution is offered in section 2.1.3. A model is related to a task either as a target model, which is affected by a task, or as a source model which is consulted by a task. Model dependency diagrams are used to show inputs, outputs and interfaces. He breaks tasks into domain acquisition, application, solution and decomposition tasks.

A method is an algorithm that organises a series of activities. Task decomposition methods break tasks into subtasks and solution methods define how to perform the subtask. The decomposition grainsize will affect the size of the reusable chunks. These methods can be further subdivided into acquisition, inference and presentation methods. All of these are of interest to reuse. The acquisition and presentation methods affect the way the user is able to interact with the system. Inference methods include learning methods and problem solving methods. Problem solving methods are concerned with case models and the question of 'what'. Knowledge is more concerned with 'why' (Clancey 1992, p.24). Learning methods enable new domain models to be inferred.

Steels comments that it would be impossible to provide a workbench which includes all of the methods because of the volume of methods. Even if possible, selection of the appropriate method would be untenable. He envisages a workbench with a range of methods that suit a particular application domain.

Steels(1993) describes COMMET as such a workbench which has been created to allow experimentation with reusability. The application is built at three levels: the knowledge level, execution and code level. The application is first designed to produce a knowledge level model in terms of tasks, models and methods. Next the system is built at the execution (the implementation of the knowledge level on computer), followed by the code level (the actual files). The complete application with the three levels is called a project and can be used to acquire knowledge. To specifically assist the developer with reuse of components there is an application kit manager which can query and guide the developer to the appropriate chunk. This chunk can then be adapted and fused at either the knowledge or execution level. COMMET has provided more structure for the design of a KBS and the acquisition of knowledge. Its ability to facilitate reuse of KBS components is of interest to this study but not directly related since the focus of COMMET is not on *activity-reuse*.

There is another area of research which combines the ontological and problem solving method approaches by developing an environment for building ontologies of problem solving methods or otherwise known as task ontologies (Mizoguchi, Vanwenkelhuysen and Ikeda 1995, Ikeda et al 1996). The implemented system is known as the Conceptual LEvel Programming Environment (CLEPE). Ikeda et al (1996) acknowledge that mainstream ontology research focuses on general ontologies and their reuse on the assumption that the human does most of the work involved in the task. They take a different view based on the premise that "where a machine plays a central role in a task .. we may say that task ontology should be investigated first". They elaborate that the task ontology should be a first principle of knowledge engineering where the "machine is intelligent enough to cooperatively work with a human and computationally strong enough to play a central role in problem solving" (Ikeda et al 1996, p. 211). CLEPE provides a Task Ontology representation Language (TOL) together with a tool for editing and browsing the ontology. Generic Process Networks (GPN) are used to offer a lexical level ontology which represents the user's problem solving process. GPN are made up of a natural language component and a task ontology translation component of the process. Using GPN users are able to use natural language to describe the task which is then translated by CLEPE. The conceptual model is executed and shown to the user where differences between what was intended by the user and what has been developed by the system can be reconciled. The purpose of this step is to ensure ontological commitment. This research differs from other research mentioned in that it is a hybrid ontological and PSM approach which focuses on the development of task rather than general ontologies. The role of the user in the specification of these models and the ability to execute the model for validation by the user are further features which distinguish this work from the mainstream. While the mapping between the conceptual (knowledge) and the symbolic level is well defined and avoids one of Schmalholfer, Aitken and Bourne (1994) criticisms of the Knowledge Level mentioned below, this work still fits with mainstream KA research in so far as modelling is seen as a prerequisite to KA.

This section has described the mainstream approaches to KBS development and reuse. Despite the widespread acceptance of the knowledge level as the basis for KBS development and the trend towards generic solutions there are a number of limitations associated with these approaches. The next section includes an analysis of the why the knowledge level has such limitations and how the work in this thesis is aimed at avoiding such shortcomings.

2.1.3 Limitations of Knowledge Level Approaches and Generic Solutions

The approaches described in the previous section have been based on modelling at the knowledge level. In keeping with the arguments offered in Chapter One of this thesis, the ability to reuse knowledge has been seen by many as essential to support the development of large KBS and many research efforts have developed generic approaches to support reuse. Unfortunately, there also appear to be a number of shortcomings of the generic and knowledge level approaches that various researchers have identified. In this section we look at these limitations, including a subsection which considers how the knowledge-level approaches are being evaluated and validated. In the final subsection, we consider how the approach used in this thesis may overcome some of these weaknesses.

The knowledge level is based on the assumption that certain actions will be performed by rational agents according to the given knowledge and goals. This is not necessarily so and does not differentiate adequately between different possible situations and when a PSM should be applied. Breuker (1994) pointed out that most problems will require a suite of problem solving methods. Selection and combination of this suite is difficult (Zdrahal and Motta 1995) and made more difficult by the situated nature of problem solving. What one person perceives as the solution to a problem will differ in each situation with what is perceived by another person. Even what constitutes a problem in the first place will vary (Lave 1988, Schon 1979).

Schmalholfer, Aitken and Bourne (1994) describe four misconceptions upon which the Knowledge level is based:

"*1. Knowledge and goals are in themselves inadequate to fully characterise intelligent systems.*
2. *Knowledge level descriptions are developed as if intelligent systems were causal systems.*
3. *For this level of abstract description, the distinction between an agent and its environment is artificial.*
4. *The knowledge level does not lie directly above the symbol level and there is no tight connection between them. Therefore knowledge level descriptions can not be reduced to the symbol level (Clancey 1991)* " (Schmalhofer, Aitken and Bourne 1994, p.87).

They further cite Clancey's criticisms concerning the frame of reference problem (Clancey 1991a) which basically argues that a description of a model is not the same as the model and that definition and interpretation of that model will require entering inside the agent so that the situation is framed in the same way. Another problem is that as systems are embedded in larger systems their appropriateness changes. The knowledge level does not accommodate change and results in knowledge level ontologies that are static and difficult to reuse (Schmalholfer, Aitken and Bourne 1994). To address the lack of flexibility of many generic approaches, some research groups are looking at smaller-grained reusable components (Tu et al 1994). Schmalholfer, Aitken and Bourne (1994) propose an extension to the Knowledge Level model using behaviour level descriptions that take into account goals, knowledge, skills and performance and the relationships between them. Context is important in allowing sharing and reuse of behaviour descriptions as they can be adjusted to support new uses.

As mentioned in Chapter One, the focus on the knowledge level has resulted in a focus on knowledge elicitation rather than user requirements elicitation. Salle and Hunter (1990) point out that the two differ and ideas and opinions form part of user requirements. So while they acknowledge the usefulness of the knowledge level, Salle and Hunter recognise that its usefulness is limited by how good a model can be found. A major problem with specifying user requirements is that users often do not know what they want or need. So the elicitation of knowledge and user requirements both suffer from the situated and uncertain nature of human knowledge and action.

Gennari et al (1994) argue that the original goal of reuse to reduce the KA bottleneck may introduce new bottlenecks if the domain and methods cannot be easily and appropriately connected. There are two different entities that may be reused: a domain-independent problem-solving method or a domain description that can use multiple methods. If the aim is to reuse problem-solving methods and ontologies then there needs to be a mechanism to join the two. Gennari et al (1994) have explored the use of bridging elements called "mapping relations", which become the third entity in the reuse kit. As they point out, the development of reusable components is a challenge since methods often become intertwined with the domain.

While there is widespread support for the development and use of generic KBS solutions. Rademakers and Vanwelkenhuysen (1993) describe some of the problems that can occur with the use of generic models. They define generic models as “predefined partial or abstract models about tasks, problem solving methods or domain ontologies” (Rademakers and Vanwelkenhuysen 1993, p. 354). As they point out generic models can assist as templates for top-down knowledge acquisition and as skeletons for abstracting raw data. The problems occur because the knowledge level is a model and is therefore subjective and imprecise. In fact we are dealing with two models that need to be kept separate, the expert's problem solving behaviour and the intended system behaviour.

Some of the problems they identified are:

1. Multiple generic models apply, but with different degrees of appropriateness.
2. Generic models are often too general to give useful support and guidance.
3. Generic models can seldom be used 'as is' and may need to be adapted or combined.
4. The terminology of generic models may be hard to tailor to the audience to which it is communicated. Matching between audiences and model terminology will be necessary.
5. The consequences of selecting a wrong model can be very costly if it is found that it does not support the needed functionality and a change to another model is required.
6. The intended goal will affect the appropriateness of the model.

7. Viewing knowledge acquisition as a knowledge transfer process misleads the knowledge engineer into focusing solely on the human expert. Context, expectations, KE experience and an experts tendency to justify rather than explain need to be considered.
8. Important details may be overlooked when trying to fit the generic model.

Despite these shortcomings, Rademakers and Vanwelkenhuysen describe some types of generic models and when they may apply. The main distinctions they make between approaches are based on the kind of generic model and the model's granularity. They suggest the following guidelines to reduce the problems identified above:

1. provide support for mixed modelling,
2. distinguish between observable expert behaviour versus desired system behaviour (current generic models concentrate on the system),
3. modeling expert problem solving behaviour alone is insufficient, it is necessary to consider communication and cooperation between actors,
4. a model of expert behaviour positions or orients the consultant to understand the behaviour - the environment should be accounted for and biases of experts avoided,
5. a model of expert behaviour is a source of information for system specification and design,
6. domain characterisation should precede the model construction phase,
7. models should be simple and transparent to its users,
8. modelling vocabularies should be extended to cover cooperation, negotiation and communication,
9. modelling and validation should be tightly coupled.

The last point is taken up in the next subsection.

2.1.3.1 Validation and Evaluation Concerns

The approach to verification and validation (V&V) of Knowledge Sharing Technology (KST) is not clear and there is not a great deal of energy being expended in this area. Gomez-Perez (1994) finds this situation has occurred due to a vicious circle of no standards so no evaluation, no evaluation so no need for standards. Hemmann (1992) argues widely accepted standards are necessary if reusability is to be possible. Gomez-Perez states that if KST is to become useful it is necessary to have some guidelines of when to use them, how to validate and verify them, where and how to find resources and how to use them. Gomez-Perez suggests the use of competency questions or counting the number of applications that successfully reuse the ontology. He provides an extensive table of approaches to evaluation by other researchers. Gomez-Perez offers detailed definitions of V & V specifically related to ontologies and adds the category of assessment which is how useful and usable a system is. Verification is checking that the system is behaving in the way it was designed. Validation is checking that the system is appropriate for the problem.

Rothenfluh et al (1994) see the use of projects such as SISYPHUS (Shadbolt 1996) as a way of evaluating general architectures. They argue that any architecture must be tested with real-world tasks. This thesis has taken advantage of the material supplied as part of the SISYPHUS III experiments in the geology domain and the usage and results are described in Sections 5.3.3, 6.4 and 8.2.2.3. Such collaboration with other research groups tests the robustness of approaches and allow comparison. A project specifically aimed at KST could uncover and address some of the issues in V&V of such technology. The commonsense KB developed by DARPA is being tested in this way.

Biggerstaff's "Rules of Three" (Tracz 1988 in Hemmann 1992) may be a useful guide when deciding when reuse of a component is useful:

1. *Before one can develop reusable software one needs to have used it three times.*
2. *Before one can reap the benefits of reuse, one needs to reuse it at least three times.*

In an evaluation study of PSM and their usefulness for reuse, Menzies (1997b) has concluded from the PSM reuse literature that PSM change is more frequent than PSM reuse. He also argues that the general lack of consensus between PSM research groups on the number and nature of PSMs and a set of PSM primitives, such as select or classify, and even on the definition of how to perform the same task, such as diagnosis, are all indicators that the use of PSMs has not stabilised and therefore the reuse of them is unlikely.

Menzies (1997b) cites two studies that have provided some statistics on PSM-based reuse: SPARK/BURN/FIREFIGHTER (SBF) (Marques et al 1992) and Mechanisms for KA (MeKA) (Runkel 1995). In SBF, the user supplies business information which is used by SPARK to create a domain-specific KA tool. A structured interview is conducted by BURN which maps the business information into a library of sub-routines (known as mechanisms). Meta-knowledge in the PSM is used to guide the mapping process and to determine what questions should be asked from the user. When the process is complete a rule-base is generated which is used by FIREFIGHTER to assist the user to execute and debug the performance system. In MeKA, each mechanism corresponded to a PSM. The PSM was broken into data structure and control knowledge and contained four modules to support acquisition, verification, generalisation and a dialogue module. The purpose of the study by Runkel was to see how frequently each

PSM developed using MeKA was reused in the development of subsequent systems. Table 2.1 shows the results of eight applications that were developed. The first application in Table 2.1 shows no reuse since all MeKAs were new but the other seven applications range from 50 to 88 percent reuse.

App. No.	Application Name	MeKAs used/total MeKAs
1	Room assignment	0/7 = 0 percent
2	Elevator configuration	5/8 = 63 percent
3	Elevator design validation	7/8 = 88 percent
4	Configuration validation	6/7 = 86 percent
5	Truck design 1	5/9 = 56 percent
6	Truck pricing	10/12 = 83 percent
7	Truck design 2	8/16 = 50 percent
8	Truck manufacturing	15/17 = 88 percent

Table 2.1 An evaluation of Mechanisms for KA Reuse
(In Menzies 1997b from Runkel 1995)

The 13 mechanisms included in SBF have been used to build nine applications that showed an decrease in development times from 1 to 17 days (using SBF) to 63 to 250 days (without SBF). Menzies (1997b) sees this result as significant but points out that from the published reports it was unclear whether the applications built with and without SBF were of equal quality and whether the personnel using SBF were the SBF developers or others biased to SBF. Another limitation is that experiments on such a scale are difficult if not impossible to repeat because they require a substantial commitment of resources which result in throw-away systems.

The MeKA study has a greater likelihood of being repeatable but once again it is unclear whether the favourable results are due to the skill of the developer or the usefulness of the tool. It is difficult to compare the two studies because the SBF study has used a reduction in development time as its benchmark and MeKA has used the amount that mechanisms were reused as the evaluation criteria. A higher percentage of reuse may be an indicator of an increase in productivity and reduction in development time but this is not made clear.

What can be argued from these studies, and is in fact what Menzies (1997a) argues, is that support for the reuse of PSMs is not clear and the benefits described in these studies can be seen as support for standard reuse of code and data structures that we see in software engineering. Menzies cites Stark (1993) who reports 70-80 percent code reuse using FORTRAN and object-oriented (O-O) design principles. Frakes and Fox (1995) consider more than code and found maximum median values for reuse in requirements, design, and code reuse at 15, 70, and 40 percent, respectively (Frakes and Fox 1995). It is interesting that:

> *"Frakes and Fox found no significant correlation between reuse and technology options such as the use of CASE tools; the presence of code repositories; or language level (assembler has a lower language level than O-O languages such as Smalltalk). Instead, the factors that were positively correlated to reuse were all organisational factors such as reuse training, unified software process; or industry type (for example reuse was high in telecommunications industries, which was possibly facilitated by the use of standard hardware configurations in that field).Menzies 1998, EVAL-5 p. 13.*

Menzies makes the point that the reports of Stark (1993) and Frakes and Fox (1995) suffer from inexact measures of reuse and that PSM researchers should be seeking to test whether the use of PSM results in benefits over standard reuse of software. We now return to an overall analysis of knowledge level approaches.

2.1.3.2 An Analysis of Knowledge Level Limitations

It is not the intention of the author to imply that there is no merit in general problem solving and ontological approaches to reuse. The point being made is that they have sufficient limitations to warrant consideration of alternative approaches. This thesis argues, and demonstrates, that complex descriptive models are not necessary prerequisites for building KBS.

Many of the problems identified with knowledge-level PSMs, ontologies and generic models are related to the inherent limits of descriptive models and the effort needed to interpret those models when they are applied. This situation is summed up by Chapman when he states:

> *"Representation is generally thought to make problems easy; if a problem seems hard, you probably need more representation. In concrete activity, however, representation mostly just gets in the way" (Chapman 1989, p. 48).*

A useful description of the types of models that are involved in capturing conceptual models is provided by Shaw and Woodward (1989). It is possible that experts have their own internalised model known as the mental model. When the expert articulates that model it becomes the conceptual model and the model that is developed from that is a model of the conceptual model, not the internalised model. To add to the difficulty in achieving a model that is truly representative of the expertise, the mental model may not correspond to how the problem is really being solved, the expert may be unable to articulate their mental model into a conceptual one and the knowledge engineer may not interpret or represent accurately the conceptual model into the final developed model.

Norman (1986) explains that when the developers conceptual model of the users mental models do not match, the design model will conflict with how the user expects to use the system. The key is to make the mental model explicit so that it is clear what the design model should be. This is an argument for building KBS using the RDR approach, that is capture knowledge in the way the expert exercises their knowledge and then look at the models that underlie the knowledge. This is another way in which knowledge in an RDR KBS becomes a mediating representation taking the user from what they do to how they think.

Menzies summarises what he believes to be the major potential and actual benefits of reuse in a very interesting paper which makes comparison between the lessons learnt from ES software development and the use (and reuse) of patterns[8] in the object-oriented community. Menzies (1997a) describes the three potential benefits of reuse of patterns: the reuse benefit which lets a designer build systems more efficiently using proven old systems; the guidance benefit that provides insights discovered in other systems and the communication benefit where patterns are useful in explaining existing systems. Menzies (1997a) sees that adoption of the knowledge level has brought with it the benefits of a more structured approach that has resulted in better organised projects that have seen some industrial success. These are guidance and communication benefits. Menzies (1997a) doubts that the reuse benefit of efficiency has yet been achieved. The KL approach approach is superior to the transfer-of-knowledge approach and acknowledges that knowledge is a model and not an artifact. However, the amount of effort being put into getting the model *right*, in such forms as PSMs or ontologies, appears to treat the model as an artifact and does not adequately acknowledge the deficiencies of models. A major concern of this thesis is that the preoccupation with complex models as a prerequisite to KA produces a bottleneck of its own with KA itself becoming a side-issue or afterthought. A situated view of knowledge does not support such an emphasis on developing good models but demands that systems are grounded in the real-world and support incremental change.

It may be that the focus on the development of complex models and analysis has occurred because much research has been based on laboratory experiments which have shown the regularities between expert behaviour and the use of reflection and abstraction. However, such settings are missing the broader context and many of the features and constraints that typically affect expert behaviour in natural settings (Lave 1988). Laboratory settings usually involve an individual in a predefined task that has

---

[8] "A fragment of a high-level conceptual model which may be useful in many applications (Menzies 1997j) oo-patterns

already been analysed and the problem-space has already been structured. The results from these studies characterise expert behaviour as reflective making use of “abstract, accepted methods that constrain the activities of practitioners and render the activities of practitioners predictable” (Shalin et al 1997, p. 195).

Given the limitations of laboratory experiments and the situated nature of knowledge, we may question the feasibility of building KBS to be used in the real-world. However, Shalin et al (1997) go on to report studies of behaviour of real-world tasks which show use of accepted methods but the decision of when to employ which method is based on “perceptual experience and temporal awareness obtained through engagement with the physical world” (Shalin et al 1997, p.196). Shalin et al (1997) have studied some contextually rich environments that cover what Woods (1988) calls “complex, dynamic and physical domains”. Such domains often involve large systems, multiple experts, quick turn-around from problem to solution and critical decision making where a wrong decision may have direct or indirect catastrophic results. The good news is that although at the low level the domains may be “complex, ill-structured and unpredictable” at some abstract level human behaviour can be seen as predictable. These predictable actions can be seen as stable behaviours (Clancey 1991b). Shalin et al (1997) state: “because accepted methods with well-defined conditions of applicability appear to dominate performance in these domains, we are optimistic about the viability of artificial aids for certain functions ordinarily thought to be intelligent” (p.213). Collins (1987) also comes to the conclusion that it is viable to use symbolic systems but we must acknowledge the role of the human in filtering the input and output accord to their social context. Similarly, while Suchman (1987) showed that plans are adapted to fit the situation and the adaptations are executed reflexively, she does not argue “don’t make any plans”. Plans are useful but we must be able to change them.

The RDR paradigm has been used in domains that qualify as “complex, dynamic, physical” and have been built not only to support but to encourage continual adaptation or refinement. Instead of the complex analysis route, RDR captures a simple model focused on the observed stable behaviours of the expert. The role of the user is paramount in building, maintaining and using the system and the expert acts as the filter that controls the inputs and outputs supporting customisation of the knowledge to suit the particular environment (Edwards 1996).

This section has described a number of knowledge level approaches and generic solutions to knowledge reuse. These approaches have been concerned more with the development of general reusable problem solving methods and ontologies than with reuse of domain knowledge which is viewed as more specific to an application. There is an implication that once the model is right the domain knowledge can be acquired fairly straightforwardly. This, however, is often not the case. In the next section we look at an alternate group of researchers that are focused on the capture of commonsense knowledge.

2.1.4 Reuse of Commonsense Knowledge

Many of the research efforts described above have been primarily concerned with the sharing of knowledge base components, problem solving methods and tools rather than sharing of the domain and lower level knowledge. Two notable large-scale exceptions are the DARPA Knowledge Sharing Effort (KSE) (Patil et al 1992) and MCC Corporation's CYC Project (Guha and Lenat 1990).

The KSE has a vision of providing a library of KB resources, which will reduce effort, cost and time and allow the building of large KB systems. The KSE is looking at four main areas:

- The Interlingua Group are creating a language (Knowledge Interchange Format KIF) that allows heterogenous knowledge representation languages to interchange knowledge. KIF enables systems using different KRs to communicate. To do this KIF accepts the sending KR format, converts it into KIF and then translates from KIF into the receiving KR. They see this solution as a better alternative to writing translation programs for every possible combination of KRs.
- The Knowledge Representation System Specification (KRSS) Working Group is concerned with variations or dialects within knowledge representation families. They aim to specify a common version of each main KR family providing a practical and comprehensive version which includes the best features.
- The External Interfaces Working Group are looking at establishing conventions in the area of communication between KBs and other computer technologies such as database, hypertext, accounting packages, etc. They have developed a Knowledge Query and Manipulation Language (KQML) which includes a protocol and common high-level language for information and knowledge exchange.
- The Shared Reusable Knowledge Bases (SRKB) Working Group is concerned with creating libraries of reusable and shareable knowledge modules. Ontolingua has been developed to provide a portable representational vocabulary which allows sharing and reuse of ontologies (a domain of discourse).

The last group is of greatest interest to this study. The focus of the SRKB is on the content of the knowledge captured rather than the aspects of how to represent the knowledge or share the knowledge, which are the concerns of Interlingua and the KRSS groups, respectively. As noted in section 2.1.2 there are a number of definitions of the word ontology used within the AI community and the SRKB have their own. The SRKB see a common ontology as:

> *"a set of definitions of representational terms used to construct expressions in a knowledge base, such as classes, relations, slots and object constants" (Patil et al 1992).*

This becomes a shared vocabulary with shared definitions. The research has been broken into the following three models. The library model is concerned with making a range of knowledge bases available and reusable. The software engineering model includes the sharing and reuse of software components. The reference model involves providing ontologies for describing domains or problem areas.

The other significant effort is MCC Corporation's CYC Project (Guha and Lenat 1990). The approach of this group is to build a large commonsense KB that can be used for building various applications. This avoids the problems of heterogeneous KR systems, family dialects, and different communication protocols. Over one million assertions have been established through various collaborations with other researchers (Soloway, Bachant, and Jenson 1987, Cohen and Loiselle 1988 and others). The KSE group see the CYC project as complementary and are providing compatibility with the CYC-L KR language through KIF. In this way the CYC outputs become components of the KSE library of resources. Over time, as sharing becomes widespread the CYC and KSE visions may lead to the same result.

Research on a smaller scale is being conducted by Pirlein and Studer (1994) who have developed KARO (Knowledge Acquisition environment with Reusable Ontologies). Their definition of an ontology is broader than the one presented by the KSE definition. They refer to that definition as a representation ontology, one that provides structure but no guidance. They are more concerned with what they term commonsense ontologies where an ontology is "the topmost level of a large hierarchy of commonsense knowledge" (Brachman 1990, p.1089). They have built a commonsense KB using the LILOG ontology with 700 concept definitions clustered into groups covering objects, time and events, qualities, quantities, measuring scales, energy and motion and assemblies and matter. LILOG is a colloboration between four German universities and IBM Germany. They use a formalised engineering framework which supports the KADS domain, task and inference layers called Model-based and Incremental Knowledge Engineering (MIKE).

Pirlein and Studer claim superiority to the KSE Ontolingua approach because KARO assists in the creation of new conceptual models of a domain by providing tools and methods rather than simply providing a communication medium between domains.

These projects, particularly the KSE and CYC, are more concerned with the concept of sharing between groups and across applications than reuse. In contrast, the focus of this study is on reuse rather than sharing. The distinction may seem spurious because the same benefits of reuse are being gained. In either case knowledge only needs to be captured once and can then be made available to a wide audience. This reduces the amount of effort required and also brings the added benefit of consistency. However reuse is more concerned with the aspect of different uses rather than the same uses in different applications. Commonsense knowledge can be seen as very low level knowledge not specific to a particular domain or application and involved sharing of the concepts across applications. The knowledge needed to support *activity-reuse* is more concerned with reuse of the actual domain knowledge within the same application possibly by the same people to support different decision situations and styles. Work by Puerta et al (1992b) does look at offering multiple interaction styles using Mecano which is a component of PROTEGE-II. PROTÉGÉ-II supports reuse of ontologies and problem solving methods which can be combined and presented to the user in different ways. However, the focus of Puerta et al (1992) is on developing knowledge acquisition tools from domain ontologies to suit a range of needs. The focus of this thesis is broader than simply knowledge acquisition.

The research by Steels (1993) and other work by Marques et al (1991) is more concerned with reuse of knowledge base components rather that the knowledge itself. If we think in terms of software engineering, much software reuse research is interested in the reuse of programs, procedures and data definitions (which can be equated to the PSM and ontologies) rather than the data (the domain knowledge). The sharing of knowledge and the reuse of components are concerned with reducing the effort involved in building KB and thereby enabling the development of large KBS. The focus of this study is not on reducing the effort in building a KB, but on how to build a knowledge resource that will encourage the use of the resource and allow the gain to be maximised. There are two reasons for this emphasis on reuse.

Firstly, it is debatable how willing organisations and individuals are to share their knowledge with others. The advent of the web has not significantly changed this and industrial use of the web tends to be for marketing purposes with possible free samples to download as an enticement to purchase a full system. With the exception of companies whose business it is to offer public databases, we generally do not see organisations sharing their databases with other organisations. On the contrary, corporate databases are strictly guarded from unauthorised and external access. How much more important is a company's knowledge resource, which contains not only data but valuable expertise as well ?

This is not to say that the KSE and CYC efforts are a waste of time. The development of a commonsense KB that acts as the foundation for building other domain specific KBS is highly desirable. Without such an effort no individual organisation would be able to build and maintain such a resource. The standards set by the KSE will be of benefit to the AI community. However, as with all standards, particularly true of the diverse computer industry, the difficulty, is in having them adopted. Typical of this problem, KSE and CYC have developed their own ontologies. Chandrasekaran and Johnson (1993) ask:

> *"is there any sense in which we can determine and agree on a set of terms that are not idiosyncratic to each research paradigm, but in fact represent some universally sharable set of analytic primitives ?..... We need a principled theory of inference operators and how they connect to world models that we construct and the tasks that we perform..... theories of generic tasks and methods will eventually have to be grounded in such a categorical content theory of our thought" (pp:267-268)*

On the other hand the 'holy grail' sought by many of a unified approach may not be terribly practical. As a study by Gambetta (1995) points out, experts prefer to use their own terminology. In the area of knowledge representation the KSE has taken the approach of offering compatibility to a range of KRS in contrast to the CYC effort that have committed to one representation.

Having looked at a number of approaches to knowledge reuse we consider what issues appeared to be the most relevant to the reuse of knowledge in general. Following this we look at the issues specific to *activity-reuse*.

2.2 The Issues Relevant to Reuse

Some recurring themes are apparent from the discussion of the reuse literature above. These themes include the importance of contextual, *deeper* and more knowledge and the use of levels of abstraction. These issues are discussed further in sections 2.2.1 to 2.2.4 which follow. Section 2.3 provides a discussion of the issues that appeared most relevant to the work in this thesis.

2.2.1 The Importance of Context

The issue of context is becoming an important factor in the reuse and sharing community (Chandrasekaran and Johnson 1993 and McCarthy 1991). Research is being done by the knowledge sharing effort (Patil et al 1992) into adding contexts into Knowledge Interchange Format (KIF) to facilitate the translation of facts from one context to another. In the area of natural language, context plays a major role as the context of a word will often determine its meaning. In answer to the growing problem of generality in AI, Guha under the supervision of McCarthy began looking at the need to provide context. Large KBS such as CYC require context to be captured, as well as knowledge, so that the knowledge could be applied to the different domains covered by the system. If knowledge is to be shared the context in which it was developed needs to be understood and described, an argument made earlier by Clancey (1992), so that the knowledge can be adapted to fit the new situation. Without consideration of context, attempts to collect commonsense knowledge "like so many butterflies" would be advocating a return to the 1960s where the AI community viewed "memory as storage" and believed "knowledge is power" (Clancey 1991b, p.245).

McCarthy (1991) and Buvac and Mason (1993) have developed a formalisation of the context concept. They argue that context allows a specific system to be developed and later generalised to fit the new problem since the context acts as a lifting axiom which transcends the original context. To support this a new modality has been introduced:

ist(c,p) where ist = is true , c = context and p = proposition.
Meaning that the proposition p is true in the context c.

The situated view places even greater emphasis on the role of context. If we believe Rosenfield who states that categories are not "stored things" but that "categorisation occurs at runtime" (Clancey 1991b, p.243) it is hard to understand how ES have found any success, particularly in the area of classification. The ability of experts to state some rules that can accurately predict to which class an object belongs is because classification can be considered a stable behaviour (Clancey 1991b, p. 249). The issue of context is highly relevant to stable behaviours. If we consider the phenomena of only being able to remember a person's phone number whilst in the process of dialing that number (Norman 1988), stable behaviours may be context dependent since they tend to become subconscious or reflexive actions.

When considering context, we must consider the environment, not just the general model and SSM (Clancey 1992). When knowledge is to be reused in different environments it is necessary to consider what is general to all situations. Consideration must be given to what is the case population, what are the world data assumptions (biases), model usage, interactional data and communication constraints. The assumptions about populations, model content and environmental boundary conditions need to be captured in the knowledge base so that they can be changed or tested against when a knowledge is reused for different purposes/situations. Clancey (1992) suggests the use of a matrix of systems, modelling methods and communication environments for boundary testing.

Contextual knowledge extends beyond environmental factors and also includes emotional, social and historical factors. Environmental and historical factors seem to be the aspects of context that most reuse approaches are able to support. Social factors have a high impact on the relevancy of knowledge, what can be considered expertise and affects not only groupware but also single user systems. The ability to customise medical systems to individual environments and cultures is a feature that Edwards (1996) comments is essential for actual systems and is offered by RDR since system development and maintenance is easy and performed by the user/s. This allows tailoring of the knowledge to the social and environmental context.

Emotions appear to be the most difficult aspect of context. On the one hand, emotions colour our perception and affect our recollections. On the other hand, the systems we build should give consistent recommendations. This does not mean that we should ignore emotions but that emotions pose too much of a problem and the alternatives are not clear. The typical approach to emotions is to treat them as a variable that affects the other inputs or outputs depending on the value assigned such as happy, sad, bored, etc. However, this type of approach does not capture the unexpected aspect of emotions which could make the same person act one way one day and a completely different way another day in what appears to be the same circumstances. While Karunananda (1993) still essentially treats emotions as an attribute, Karunananda has taken into account the cumulative effect of emotions which is often the cause for different responses to similar stimuli. Her approach is based on Buddhist philosophy of the human mind.

Context is an important aspect of RDR which is captured in the cornerstone cases and the exception structure. In section 3.2.2 context is discussed in some detail together with a description of how it is handled by RDR. We move on to the next issue which is the abstraction of knowledge.

2.2.2 The Abstraction of Knowledge

Many have advanced the view that reuse is facilitated by structuring the knowledge into different levels of abstraction (Chandrasekaran and Johnson 1993, Clancey 1992 and Swartout and Moore 1993). Hemmann (1992) points out that this is a lesson that should have been learnt from the experience of reusing software.

Chandrasekaran and Johnson (1993) offer an interesting notion of what abstraction is all about. They argue that knowledge needs to be normalised in order to be reused. When abstraction is viewed as a way of normalising knowledge then abstraction seems a logical requirement of reuse.

Clancey sees the key to enabling reuse is the use of abstract terms for the description of the domain and control knowledge. To generalise a domain specific rule it is necessary to replace specific terms with a more abstract term and then to add in the relationships. New relations and the reclassification of expressions may be necessary to enable reuse. He advocates the inclusion of metarules to allow the knowledge to be processed more flexibly by reordering clauses, not just reordering rules. Metarules need to be general enough to be reusable and specific enough to enable reuse, that is they need to be comprehensible.

Clancey (1992) notes that a different type of reuse occurs when abstraction of control or domain knowledge is used more than once in a program for different circumstances. He states "the knowledge base is easier to construct because the expert need not specify every situation in which a given fact or relation should be used" (Clancey, 1992, p.17) The knowledge base is therefore less brittle and can use the facts and relations for different purposes. If a new kind of knowledge is to be added then new metarules and relations may be needed.

Research by Richards (1994) also found that through the use of metarules, in the form of WHEN-CHANGED *methods,* it was possible to adapt a consultation style ES to behave like a decision support system (DSS) by supporting 'what-if' analysis. In that study no changes were made to the domain knowledge only to the way the domain knowledge was used.

Research on knowledge of the visual process has found that there is "a strong correspondence between knowledge organisation and knowledge use" (Tsotsos 1982). The aspects of context and the abstraction of form and motion knowledge into an hierarchy are considered important to adequately represent visual knowledge. Many describe the need for abstraction without giving many clues to what levels may exist. The following list by Tsotsos (1982) of types of concepts that may need to be represented could be used to identify categories:

- prototypical concepts and instances
- discrete and structured concepts
- ordinality and measure
- properties, qualities and attributes
- causality

- spatial knowledge
- temporal knowledge
- states, events, actions and change
- procedural knowledge
- situations and contexts
- description by comparison and differentiation
- conjunction, disjunction and negation
- inheritance, instantiation and reasoning with defaults
- certainty, strength of belief
- expectations

These concepts could then be organised by:

- plurality (sets, sequences, membership, partial orders)
- projection
- abstraction
- multiple viewpoints
- or meta-knowledge.

The use of levels of abstraction has been particularly advocated for the purposes of explanation. Chandrasekaran, Tanner and Josesphson (1988) argue that most knowledge representation methods are at too low levels of abstraction to provide adequate explanation. Swartout and Moore (1993) categorise explanation approaches into four main groups, three of which use abstraction for explanation. These first two approaches have been considered already and are: the use of metarules and the use of generic problem solving methods such as generic tasks. The third group is Swartout and Johnson's work on Explainable ES (EES) that use an automatic programmer to provide varying levels of knowledge and avoids the loss of what he terms *compiled out* knowledge. Each group uses different types and numbers of levels but they all tend to have levels that range from concrete to purely abstract, or from specific to general. The absence of different levels of abstracted knowledge has been a shortcoming of RDR which this thesis has addressed. This work is described in Chapter Five where FCA has been used to find the intermediate and higher levels. A more detailed discussion is given in Section 7.3.3 where the explanation activity is considered. An issue similar to abstraction is the concept of deeper knowledge. We look at this next.

2.2.3 The Need for Deeper Knowledge

A number of different terms, such as deep versus shallow, derived versus direct and deep versus compiled, are used by different researchers to express the notion of levels of knowledge. Chandrasekaran and Johnson (1993) use the terms direct and derived knowledge. As the terms imply, direct knowledge is readily available but derived knowledge requires computation, perhaps via a subtask, from existing knowledge or it must be acquired from the external environment. Derived knowledge can be seen as a *deeper* level because it does not exist on the surface but requires moving deeper into the knowledge performing various procedures until it can be generated. Chandrasekaran and Johnson (1993) explain that depth is a relative notion and the depth of one knowledge element to another can be given in terms of the amount of processing needed to get to each of the elements.

Further understanding is gained by looking at some other related distinctions. Swartout and Moore (1993) speak of deep versus compiled knowledge. Chandrasekaran and Johnson (1993) equate direct knowledge with compiled knowledge because it can be used without the need for further processing. The deep knowledge is the knowledge that is given as a prerequisite for system development. For example, it may be knowledge about the domain, organisation, work and data-flows, various computer files and documents, recorded discussions and so on. Different parts of the available knowledge are used to develop the appropriate system. Often only what is needed to solve the predefined task is kept. Much of the other knowledge is conceptually lost or *compiled-out* and can not be utilised by the system. This is seen as a major cause for the brittleness of ES and their inability to provide adequate explanations.

Another analogy is the comparison between model-based and rule-based reasoning (Chandrasekaran and Johnson 1993). They see the model as being an example of deeper knowledge where more general knowledge is represented than in the case of a specific rule. The model contains “the principles of the domain while rules are supposed to refer to relatively *ad hoc* associations between evidence and hypotheses” (Chandrasekaran and Johnson 1993, p.250).

Finin and Klein (1989) describe the pros and cons of *deep* and *shallow* models. Shallow models are generally easier to build because the knowledge is encoded in the heuristics where the conclusion is drawn directly from the facts. This avoids much of the complexity of trying to get experts to define how they arrived at a conclusion. This also means that inferencing is straightforward and efficient. However, shallow models contribute to the "brittleness" problem of ES. Since the knowledge is not explicit, they are inflexible, can be difficult to maintain or modify and are inadequate for explanation. Deeper models contain explicit knowledge in a manner that can be equated to reasoning from first principles. Finin and Klein (1989) see them as more robust and more verifiable for completeness and correctness. Deeper models tend to handle new problems better and facilitate better explanation. As a trade off, the reasoning process is more complex and slower.

As Finin and Klein point out, there is much discussion on the development of deeper models but there is little guidance on how to build these deeper models. They propose:

1) using an informal definition of "knowledge depth" using the *deeper-than* relation that is applied to sub-tasks,
2) abstracting of high-level reasoning tasks such as diagnosis or simulation and
3) analysis of the relative depths of reasoning task models found in various domains.

Kidd (1985) offers an alternative structuring system based on functional requirements of remedies, which may be broken into three levels: symptom, disorder and cause. The approach is aimed at man-machine symbiosis, but is limited to diagnostic systems only.

The type of knowledge contained in an RDR KBS is hard to define using these various and sometime conflicting terms of deep, shallow, direct, derived and compiled. The assertional RDR KBS can be seen as the direct (Chandrasekaran and Johnson 1993) knowledge. And the terminological KBS that FCA produces can be seen as the deeper derived knowledge. If we use Swartout and Moore's (1993) terms, the RDR KBS is more like deep knowledge which includes the cases and the behavioural knowledge. Neither the RDR assertional KBS or the terminological KBS that FCA develops fit with notion of compiled knowledge since in both cases no knowledge is lost but with FCA more knowledge is generated in the form of higher concepts. Even though the line diagram does not make use of the cases, it is shown in Chapter Six how the line diagram can be used to pop-up the case associated with a particular concept. If we use Finin and Klein's (1989) definitions, an RDR KBS is made up of shallow knowledge that is contained in the heuristics where conclusions are drawn directly from facts. The RDR knowledge does offer the benefit of ease of capture but does not suffer from the maintenance problems associated with shallow models. The extent to which RDR suffer from the brittleness problem is unclear however the ability to support explanation has been seen as an area that needed improvement in RDR. This study acknowledges that knowledge of different types and depths is needed to support *activity-reuse* and is addressed in the incorporation of FCA into RDR described in Chapter Five. We look now at the last issue which is the need for more knowledge.

2.2.4 The Need for More Knowledge

Another finding of the various reuse studies is that extra knowledge may be needed when the usage of the knowledge changes such as using knowledge intended for a consultation system for causal modelling (Lee and Compton 1995) or for explanation (Buchanan and Shortliffe 1984).

Clancey (1992) also concludes that although the SSM perspective is useful for a range of problem types, knowledge that is needed to solve one type of problem may not be adequate to solve another type of problem. It is possible to achieve something without referring to what is conceptually being done. This means that it becomes difficult when we take knowledge out of context and try to apply it to a different situation and that contextual knowledge is one type of knowledge that may be need to be added.

The study by Richards (1994) found that extra knowledge in the form of rules was not necessary when adapting to a 'what-if' system style, although additional models to handle the aspect of time had been added. It seems reasonable that different situations may impose different knowledge requirements and the question of whether more knowledge is needed can be best summed up by Hemmann:

> *"Of course, at least the domain model must either contain redundancy to be combinable, or it must be extended by the knowledge engineer according to the knowledge needs of the individual interpretation models" (Hemman 1992, p. 6).*

In this thesis the capture of additional knowledge has not been necessary but the derivation of additional knowledge in the form of higher level concepts and the knowledge structure has been found to be necessary to support *activity-reuse*. In the next section we summarise the findings of this chapter and the main issues of relevance to *activity-reuse*.

2.3 The Issues Relevant to Activity-Reuse

This section only briefly mentions the main issues identified above concerning contextual, abstracted, deeper and more knowledge as they relate to *activity-reuse* since they are covered in other sections of this thesis. Capturing knowledge in context is a key benefit offered by RDR and is discussed in Section 3.2.2. The need for abstracted knowledge became apparent when looking at various reflective modes to enable the higher level concepts in the domain to be found. The work is reported in Chapter Five which looks at building a subsumption hierarchy from the primitive concepts in the RDR rule-base. This abstracted knowledge fits Chandrasekaran and Johnsons (1993) definition of deep knowledge as being model-based knowledge. The decision-list structure of an MCRDR KBS can also be used to provide layers in the explanation path because the higher level rules offer more general principles and the refinements describe the exceptions to the general rule (Catlett 1992). Swartout and Moore’s definition of deep knowledge corresponds more to background or supporting knowledge and to some extent is satisfied by the use and storage of cases in RDR which are used both for KA and for explanation purposes. With respect to the need for more knowledge this study has not found it necessary to capture more knowledge but the derivation of higher level knowledge using formal concept analysis can be seen as a form of learning more about the domain. Some activities, such as causal modelling, may require additional knowledge. Lee and Compton (1985) have looked at causal modelling using RDR KBS and have hypothesized that much of the additional information needed can be acquired as a rule is added with minimal increase in difficulty or work load on the user's part.

Many of the reuse approaches presented attempt to develop generic approaches which could be applied to other situations. As enumerated by Rademakers and Vanwelkenhuysen (1993) in Section 2.1.3. there are serious problems with the use of generic models, the key problems being matching the best PSM with the problem and then finding the best way to adapt the method. There is also then the problem of how to handle linking of the PSM and the domain model (Hemmann and Voss 1993a). To avoid these issues what is really needed is a representation that can handle the problem regardless of the problem categories into which it may fall. RDR research is heading in this direction as shown in the recent work which required a small modification to the MCRDR inference engine to allow the one problem solving method to be used for classification and configuration tasks (Ramadan et al 1998) and a different alteration to support a control problem (Shiraz and Sammut 1998).

As explained, this study is looking at reuse from a different angle and is primarily concerned with the reuse of knowledge within the domain for which it was captured to suit a range of user needs. The focus on PSMs, ontologies and KBS components is often concerned with reuse across domains. User issues are of lesser importance to these approaches than seeing if something that has worked in one domain is transportable to another.

Activity-reuse will involve investigating questions such as “What are the constraints on the usability of knowledge recorded for different purposes ?” (Neches et al 1991, p.49). This question was asked by the KSE group but does not appear to have been answered by their endeavours. The research reported in this thesis is concerned with this question. In Chapter Seven a description of each activity that was explored and their particular requirements are given.

There are two aspects that appear relevant in pursing a solution to reusing knowledge for different activities. One aspect is the reuse of knowledge already captured and the other is building KBS designed for *activity-reuse*. The first approach looks at taking an existing KBS and reusing the knowledge therein. This will involve what Hemmann (1993a, p.2) loosely defines as "careful analysis" and he cites Clancey's work on

MYCIN in developing the idea of Heuristic Classification as an example of such a technique. It is interesting that despite the strong emphasis Hemmann gives to the use of formal methods such as KADS and the even more formal methodology he has built called MoMo, he offers very little concrete guidance on how to analyse what already exists. Descriptions such as "concentrated on the code" (Hemmann 1993a, p.3) are not very helpful. This study is interested in a stronger solution to reusing existing knowledge than "careful analysis" and the use of rough set theory (Pawlak 1982) and formal concept analysis (Wille 1982) are two techniques that were investigated for this purpose. Chapter Four and Five describe these theories and the work done with them.

The second approach is really the most important. An outcome of this thesis is a general solution to building KBS that are reusable for multiple activities. It seems logical that something that is built from the start with reuse in mind will be better than a system that needs to modified later to cater for reusability. The interesting thing with RDR is that this has not been the experience of this study. The simplicity and the flexibility of the approach meant that using an existing system did not constrain the development of a reusable system. The approach provided offers a way of capturing assertional knowledge and deriving terminological knowledge that is then considered adequate to support a range of activities with the burden on the user interface to support these different uses.

The discussion in Section 2.1.3 covering the limitations of the knowledge level and generic solutions is also important because the approach adopted in this study does not follow these approaches but rather emphasizes solutions that do not rely on the user to state their own conceptual models. The prerequisite that knowledge level modelling must be performed before KA can commence is a major weakness of these approaches since they add to the KA bottleneck and do not acknowledge that although a knowledge level may exist we do not really understand it or how the human mind works. It also appears that if humans use a knowledge level they tend to use it after the event as a means of reflection. An alternative approach is offered by RDR which simply and easily captures behavioural knowledge that can be validated in conjunction with the cases used in the KA process. If has been found during the course of this study that behavioural knowledge is not enough and that some understanding of the conceptual or mental model is necessary. FCA has allowed the derivation of these higher level models from the performance knowledge and it is argued that a post-hoc approach to the development of such models is a feasible and possibly more reliable or verifiable approach.

Up till now a number of claims, but little detail, regarding RDR has been given. This omission will be remedied in the next chapter which describes RDR, its variations, implementations, strengths and weaknesses. In addition other related theory is discussed, namely personal construct psychology and case-based reasoning. We also look at a nearest neighbour algorithm that was developed to determine the closeness of concepts in the KBS and what limitations of RDR this thesis has addressed in the pursuit of a multi-purpose system.

Chapter 3

3 Reusing Knowledge in Ripple-Down Rules KBS

The focus on reuse in this thesis has been termed *activity-reuse* and was described in Chapters One and Two. This focus is further narrowed by particularly considering the *activity-reuse* of knowledge in Ripple-Down Rules KBS. In this chapter RDR will be described together with the reasons why it has been chosen as the foundation for this research into *activity-reuse* and its relationship to existing theory. The chapter closes with a discussion of the limitations of RDR which this thesis has sought to address to allow it to support the reuse of knowledge for a wide range of activities.

3.1 What are Ripple Down Rules

RDR were developed in answer to the problems associated with maintaining a large medical KBS. It was evident that the knowledge provided by experts was situated and dependent on the context in which it was given. It was observed that experts did not give explanations of why they had chosen a particular recommendation but rather they tended to justify their conclusion (Compton and Jansen 1990). Justifications tend to be more directed to the audience and may leave out many of the aspects really used in making the decision. For example the parent of a sick child will be given a different justification than another doctor for the same treatment. Not only would the terminology used differ but the actual reasons given may be different. Justifications also often included reference to certain aspects of the case. Thus the applicability of the knowledge depended on the context which consisted of the case and the person receiving the knowledge. This observation resulted in an emphasis on capturing knowledge in context by storing of the case that prompted a new rule to be added. The unwanted side effects associated with typical rule-based systems (Solloway, Bachant, and Jensen, 1987) were also avoided by the development of the exception structure which is designed to localise the effect of additional rules. The emphasis on maintenance has resulted in a KA technique that is failure-driven (rules are only added in response to a wrong conclusion) and which not only supports but expects incremental system development and on-line validation of knowledge. In Sections 3.1.1 and 3.1.2 the two main implementations of RDR, single classification and multiple classification RDR are described. A number of variations of these are described in Section 3.1.3. Despite the variations, the main ideas embodied in each implementation are: KBS that are grounded in cases, an exception structure and a simple KA technique designed to be performed by the expert and which encourages incremental development, maintenance and validation of the knowledge base.

3.1.1 Single Classification RDR

RDR were first developed to handle single classification tasks. An exception structure in the form of a binary tree is used to provide rule pathways. When the expert determines that a conclusion is incorrect a new rule is added to the rule that incorrectly fired. If the new rule results in the incorrect rule remaining true then it is attached to the true path of the incorrect rule, otherwise the new rule is attached to the false branch. The case that caused the misclassification to be identified is stored in association with the new rule and is referred to as the cornerstone case. The purpose of storing the case is to assist the user with KA and to provide validation of the new rule. The new rule must distinguish between the new case and the case associated with the rule that gave the wrong conclusion (Compton & Jansen 1990).

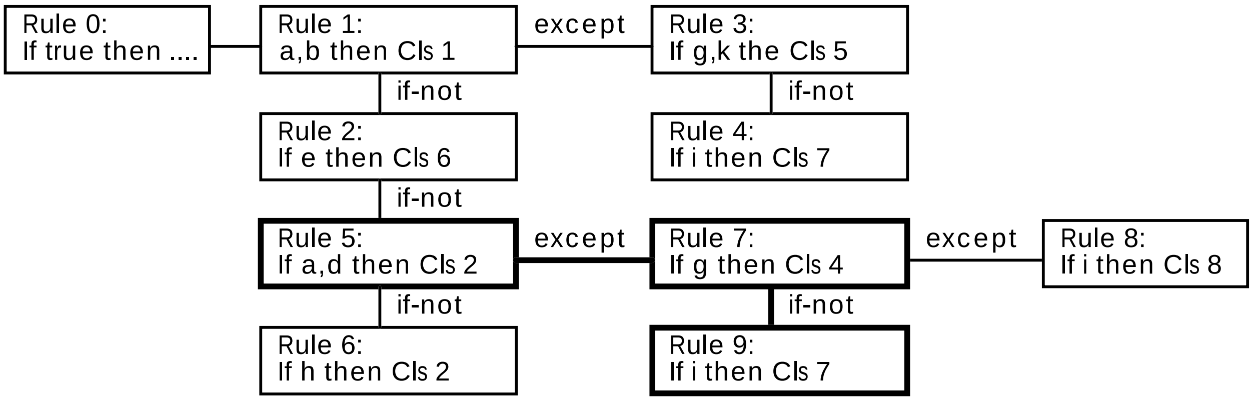


Figure 3.1. A single classification RDR KBS.
Each rule can be seen as a pathway that leads from itself back to the top node which is rule 0. The highlighted boxes represent rules that are satisfied for the case {a,d,i}

Using the grammar developed by Scheffer (1996), a single-classification RDR can be defined as a triple <rule,X,N>, where X are the exception rules and N are the if-not rules, see Figure 3.1. When a rule is satisfied the exception rules are evaluated and none of the lower rules are tested. The major success for this approach has been the Pathology Expert Interpretative Reporting System (PEIRS) (Edwards et al 1993), a large medical expert system for pathology laboratory report interpretation built by experts with minimal intervention of a knowledge engineer. PEIRS is one of the few medical ES that have actually gone into routine use. It went into operation with 198 rules and expanded over four years into over 2000 rules, covering 12 different tests. A total of approximately 100 hours was spent on KA. The development of 10 rules per hour for RDR is outstanding compared to industry standards of only 2 or 3 rules per day (Edwards 1996). Despite the random order in which knowledge is added, simulation studies have shown that the RDR approach of correcting errors as they occur produces KBS that are at least as compact and accurate as those produced by induction (Compton, Preston and Kang 1994 & 1995) .

3.1.2 Multiple Classification RDR

Multiple classification RDR (MCRDR) have more recently been developed to handle classification tasks where multiple independent classifications are required (Kang, Compton and Preston 1995, Kang 1996). This method builds n-ary trees and consists only of exception branches. A better description may be sets of decision lists joined by exceptions. In contrast to single classification RDR all rules attached to true parents are evaluated against the data. Figure 3.2 shows an example MCRDR showing two levels of decision lists. An MCRDR is defined as the quadruple <rule,P,C,S>, where P is the parent rule, C are the children/exception rules and S are the sibling rules within the same level of decision list. Every rule in the first list is evaluated. If a rule evaluates to false then no further lists attached to that rule are examined. If a rule evaluates to true all rules in the next list are tested. The list of every true rule is processed in this way. The last true rule on each path constitutes the conclusions given.

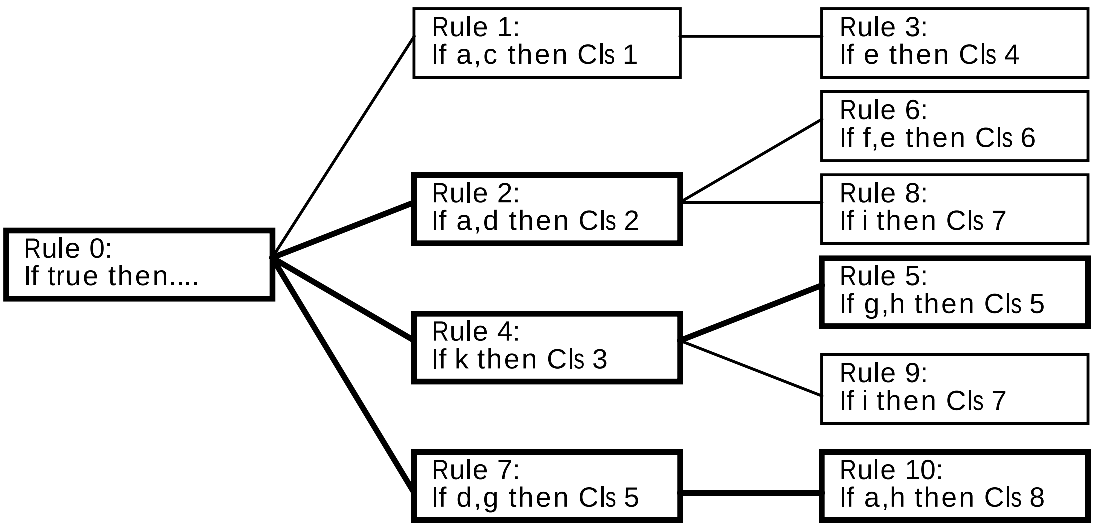


Figure 3.2. An MCRDR KBS.
The highlighted boxes represent rules that are satisfied for the case {a,d,g,h,k}. We can see that there are three conclusions, Class 2 (Rule 2), Class 5 (Rule 5) and Class 8 (Rule 10).

In single classification RDR only one case is associated with each rule. In MCRDR there may be multiple cases that must be distinguished from the current case. In the KA approach developed, the expert is presented with one cornerstone case at a time. The expert constructs a rule to distinguish the new case from the first case presented and then each case in the cornerstone list is evaluated to see if it is also distinguished by the new rule. If a case is satisfied by the rule the expert must add extra conditions to the rule to distinguish this case. This continues until all related cornerstone cases are distinguished. Remarkably the expert provides a sufficiently precise rule after two or three cases have been seen (Kang, Compton and Preston 1995). The decision of where to add a new rule to the KB is affected by the design of the user interface, user preferences and the situation. If rules tend to be added to the top level the domain will be covered more rapidly but there may be greater errors. If rules are added to the end of pathways less cases will be seen so there will be less errors but slower domain coverage (Kang 1996). New cases may be misclassified in one of three ways: one or more of the conclusions are incorrect, one or more conclusions are missing or a combination of incorrect and missing conclusions. Table 3.1 shows the three ways in which a new can be added to correct a knowledge base depending on the nature of the misclassification. As described in Table 3.1 and shown in Figure 6.5, the user may decide to stop an incorrect conclusion instead of replacing it with a new conclusion. This is achieved by adding a stopping rule which has a null conclusion in the same way as adding other types of rules.

Wrong Classification	**To correct the KB**
Wrong classification to be stopped	Add a rule (stopping rule) at the end of path to prevent the classification
Wrong classification replace by new classification	Add a rule at the end of path to give the new classification
A new independent classification	Add a rule at a higher level to give the new classification

Table 3.1: The three ways in which new rules correct a knowledge base (Kang 1996)

Simulation studies (Kang, Compton and Preston 1995) have shown MCRDR to be a superior representation to the original RDR structure by producing knowledge bases that mature more quickly and are more compact even for single classification domains. It is conjectured that this occurs because more use is made of expertise rather than depending on the KB structure (Kang 1996).

With the development of MCRDR a whole new range of possibilities has been opened up. Work is being done into using a modification of MCRDR to support construction tasks, specifically configuration for ion chromatography. From prelimary results (Ramadan et al 1997a, Compton et al 1998) it appears that MCRDR is a sufficiently rich and robust representation to blurr the distinction between problem types which will allow the expert/user to simply differentiate one case from another without concern for how it is being handled by the system. To enable MCRDR to be used for configuration, the inference engine loops around filling in missing values until a complete solution is found. If only one value for a particular attribute is recommended it is added to working memory or if multiple values are suggested by the system the user is asked to choose the correct value, and thus add a new rule. The ability to use MCRDR for classification and configuration tasks is significant and suggests that RDR can be seen as a family of PSM, where the PSM is weak but is able to find a solution for both types of tasks when more data are added.

MCRDR was chosen for this study since the ability to provide multiple conclusions for a given case is more appropriate for many domains and, more importantly, because the problem of how to handle the false "if-not" branches (Richards, Chellen and Compton 1996) does not exist. Since all branches are exceptions and form true branches MCRDR knowledge bases may be easily converted to a flat file structure, with each record representing a rule pathway through the knowledge. It is this ability that made MCRDR KBS readily convertible to a formal context as described in Chapter 5.

3.1.3 Previous Implementations and Variations of RDR.

Over the past 10 years there have been numerous implementations of RDR. Each one has essentially acquired knowledge in the same manner, by presenting cases to an expert who accepts a conclusion or creates a new rule with the correct conclusion which is attached to the incorrect rule. A number of systems based on single classification RDR have been developed. XRDR is a general purpose system which handles single classification problems. Other noteworthy variations are Time Course (TCRDR) which was used by PEIRS and Recursive RRDR (Mulholland et al 1993). TCRDR used various functions and features to handle time course data. For example, if there were data covering five time periods the user could define a function CURR to determine which A-V pair would be used in evaluations of the current value of that attribute. This type of preprocessing of the input data is typical of many of the implementations of RDR and does not affect the way KA or inferencing is performed. On the other hand, Recursive RDR (RRDR) did require a modification to the inferencing process which involved repeated inference cycles using the single classification RDR structure. On each cycle the last true conclusions at the end of each path were returned. A heuristic is then used to choose which conclusions to accept and these are used as inputs and inferencing commences from the beginning again. Another implementation, which was an intitial and only partial solution to the ion chromatography configuration problem, combined RRDR with Possible RDR (PRDR) (Mulholland et al 1993) and reported all last true conditions after taking any branches that might possibly be true. PRDR considers a subset of the branches used in RRDR. Both of these techniques can be difficult to manage as often too many alternatives are provided and it is hard to determine the useful ones.

Interactive RDR (IRDR) is a technique which allows the RDR system to prompt the user for more information when required. The user may enter a value or indicate that the value is “unavailable”. The response is added to the case which is then reevaluated. When changes are made to the case, no matter how small, it is possible that another path will be found through the tree. Some simulation studies have been done using IRDR that yielded poor results. It is conjectured (Paul Compton, personal communication) that it is very difficult for the user to anticipate how the path will change and therefore determining the appropriate change to an A-V pair to test out predictions is very complex. A more recent development to provide a consultation-style system is the use of rules which are added by the expert to state what A-V pair should be requested next when the value of a certain attribute is known. This is a fairly primitive approach and there are plans to interactively compute the information gain to determine the key attributes that should be queried. This is similar to the KADS solution to SISYPHUS III which is described in section 7.2.1. An alternative to using information gain alone would be to use the rules as the input cases and the statistics collected by *credentials* (described later in this section) can be used to weight each rule according to the number of times the rule has fired. Use of these statistics is important because, unlike the use of algorithms like C4.5 with examples, there is only one of each rule and the importance of one rule over another can not be determined without some measure of usage. Rules whose conclusion is to ask a question can also be incorporated to override the suggestions made by machine learning.

Another implementation (Kang 1995 and Edwards 1996), known as WISE, first determined the conclusion of a case using standard RDR and then found all other branches that gave the same conclusion. The goal of the study was to determine the usefulness of storing the history of corrections, provided in an RDR KBS structure. The pathways below each satisfied rule node were compared (normally only one pathway is traversed and kept in single classification RDR). These multiple pathways postulated four possible patterns of knowledge correction and could be used to suggest that a conclusion may be wrong. This would add even further to the explanatory power of the knowledge and allow comparison between alternative classifications of the data.

As a follow-on from WISE, Edwards has looked at building RDR KBS that use reflective learning through what is termed *prudence* and *credentials* (Edwards 1996). The idea behind *prudence* was to reduce some of the brittleness problem that ES suffer from when they reach the limits of their knowledge. This was achieved by altering the behaviour of the ES rather than the knowledge contained therein. WISE developed into Feature Recognition Prudence (FRP), which was implemented to see whether a different conclusion could have been reached by an alternative path by checking the children of other nodes with the same conclusion. Another approach was termed Feature Exception Prudence (FEP) which allowed the system to know when it was reaching its boundaries.

This was achieved by keeping a context profile which included the permissible features for each interpretation. When an inference was performed all A-V combinations (features) in the case, not just the rule conditions, can be evaluated against the permissible range of values for each attribute in the context profile. If a feature is not found in the context profile the pathologist is warned that the conclusion may be incorrect. In the *exceptions* study it was found that knowledge in the form of *filters* was needed so that in certain circumstances data would not be checked, such as checking the value of the attribute *ovulatory* when the patient is a male. One alternative to the use of filters is to provide a secondary KBS which contains rules with valid combinations and other knowledge required to support reflection which could be used in conjunction with the primary KBS containing the main RDR rules.

The other method of reflection was the use of *credentials* much in the same way as we may seek the qualifications of an expert before deciding to accept their advice. *Credentials* kept track of various statistics such as how many times a rule had been fired and how many of these had been accepted by the pathologist. With these statistics the user can determine how the accuracy of knowledge in this rule compared to the accuracy of the total system and decide whether to accept the system's recommendation.

RDR have also been successfully extended to acquire complex control knowledge needed for a flight simulator (Shiraz and Sammut 1998). Much of the knowledge required for this task is subconcious and can not be acquired through techniques such as interviews. By reasoning about cases the operator is able to evaluate and modify the KBS. Dynamic RDR (DRDR) has been developed based on single classification RDR. However, unlike single classification RDR, DRDR handles four KBS concurrently and produces multiple conclusions one for each of the four conclusions regarding Elevator, Flaps, Roll and Throttle. To further assist the dynamic system with learning, machine learning is used to modify and complement the rules created by the flight operator using the flight log. The algorithm is known as LDRDR and unlike most machine learning algorithms it learns incrementally by determining when an action is performed and using the action and attributes that caused the action to form a new rule. This work had begun while MCRDR was still under development and it would be interesting to compare the benefits of DRDR compared to MCRDR for the same task.

MCRDR was first implemented on the Macintosh platform but the development of MCRDR for Windows for the PC was begun in 1995. Standards tools included in implementations of RDR are the ability to browse and visualise the tree, browse individual traces and examine rules and their status based on the current case. Various statistics are available such as a count of the number of times a conclusion is found in rules, the complexity of individual rules (the number of conditions in the antecedent), the frequency that whole conditions occur, the full complexity of a path, the number of rules in spine subtrees and a list of conclusions and which rules use them. Numerous other statistics can possibly be calculated.

A new approach to KA using RDR has recently been developed which seeks to use an ontological approach to KA using RDR. The system is known as Ripple down rule-Oriented Conceptual Hierarchies (ROCH) (Martinez-Bejar, Benjamins and Martin-Rubio 1997 and Martinez-Bejar et al 1998). Using ROCHS the expert defines a conceptual hierarchy of the domain from which cases are first developed followed by rules to cover those cases. The hierarchy includes IS-A and PART-OF relationships and is verified by the system and validated by the user as it is developed. ROCH is currently being extended to cover fuzzy domains with the development of fuzzy ROCH (FROCH) (Martinez-Bejar, Shiraz and Compton 1998). The ROCH work is related to the approach espoused in this thesis in that it results in a conceptual hierachy and set of rules for a domain. However, as with most mainstream approaches to KBS the focus is on the manual development of a model which it uses to guide KA. Unlike mainstream approaches ROCH is an incremental technique allowing rules and/or the hierarchy to be developed on a case by case basis. As mentioned earlier and described in more detail in Chapter Five, the work reported in this thesis differs in that KA only involves changes to the A-box set of assertions in an RDR performance system and the T-Box conceptual hierarchy is derived automatically from the A-box using FCA.

Another approach that has incorporated a conceptual hierarchy into the RDR KA process is Nested Ripple Down Rules (NRDR) (Beydoun and Hoffman 1997 and 1998). This work is concerned with capturing search knowledge and allows concepts and rules at various levels of abstraction to be defined and reused. The chess domain is being used as the testbed for this work but the work is concerned with human search knowledge in general. For every concept to be defined a simple (single classification) RDR tree is used. The conclusion of rules within a concept definition are boolean and can be used as conditions in other concept definitions to form a hierarchy of concepts. Realising the importance of providing the user with a visusal representation of the rules the system developed, SMS1.2 allows the user to view the rules as a tree structure similar to those provided in other implementations of RDR. NRDR is an interesting development and potentially useful for the reuse of knowledge due to its structuring of the knowledge into various levels. Beydoun (personal communication) sees NRDR to be a human-like approach to search involving a depth-first search and backtracking unlike the MCRDR approach which is a breadth-first search. It is felt (Beydoun, personal communication) that the major contribution of this work is the ability of the user to use their own terms unlike other approaches to acquiring search knowledge which require the expert to use the KE imposed terminology. NRDR differs from most RDR implementions or variations in that while it makes use of cases it does not use a difference list since a large number of relational operators are used and a difference list would be computationally expensive and unwieldy for the user. It is up to the expert to decide what the differences are and how they should be used to form the new rule. As with ROCH the NRDR approach relies on the expert to develop the conceptual hierarchy. It is noted that "this hierarchical structure of the knowledge base causes problems for keeping the entire knowledge base consistent when a single concept needs to be altered" and "a more profound update problem is dealing with inconsistencies due to localised updates in the hierarchical knowledge base" (Beydoun and Hoffmann 1997, p. 7). While they provide some solutions to these problems it appears that the imposition of more structure and levels have impacted on the ease of maintenance in standard RDR. To retain the maintenance and KA strengths of RDR, the approach developed in this thesis has used the MCRDR structure without alteration.

The various implementations described above indicate the versatility of the RDR structure. Generally, each variation still used:

- cases to provide context, assist in forming rules and for validation;
- the exception structure;
- the use of some form of preprocessor of the raw data;
- incremental KA and maintenance; and
- the development of a simple model in terms of A-V pairs and conclusions.

The differences tended to concern how the knowledge was being presented and manipulated and the inferencing strategy.

The only research into the reuse of RDR knowledge for other activities is the work by Lee and Compton (1995) where the knowledge in an existing RDR KBS was used to find causal links to support causal modelling. In this approach it was found that additional knowledge from the expert, in the form of causal links, was often required. A description of this work is given in Section 7.3.2. To take the work on causal modelling further, it was the goal of this research to take knowledge in an RDR KBS and see whether it can be reused for a wide range of activities. As suggested by Stelzner and Williams (1988) the challenge was to see if the RDR structure could support any type of activity usage with only changes to the user interface required. However, during this study it was found that for the reflective modes, such as critiquing, it was necessary to learn more about the higher level concepts and the dependencies within the knowledge base. The search for a solution to this problem resulted in the work reported in this chapter (the nearest neighbour algorithm in Section 3.3.2.1), Chapter Four (the use of rough set theory for finding dependencies between rule conditions and conclusions which led to a V&V technique), Chapter Five (the incorporation of FCA into MCRDR) and Chapter Six (the use of MCRDR/FCA for RE). An analsysis of how the resultant system MCRDR/FCA supports a wide range of activities is given in Chapter Seven.

Another aspect of the versatility of RDR is the range of problems which it can handle. This is different to *activity-reuse* and is more concerned with *task-reuse*. RDR has most widely been used for classification tasks but its ability to be used to control a flight simulator, capture search knowledge for chess or to configure an ion chromatographer using basically the same structure indicates that RDR constitute a family of PSMs that can be used to handle multiple types of problems. The major focus of current knowledge acquisition research is on problem solving methods (PSMs). This emphasis tends to view domain knowledge and the PSM as two quite separate issues and little attention seems to be paid to actual acquisition of the domain knowledge. In contrast, the major focus of RDR is on facilitating acquisition of domain knowledge, so that a domain expert is able to add the bulk of the knowledge without knowledge engineering support or skills. RDR have been used successfully for classification, configuration, control and heuristic search tasks. The key difference between RDR and other approaches seems to be that RDR research is working towards an approach which supports the addition of domain knowledge specifically to overcome the limitations of the problem solver. This raises the question (or rather returns to one of the earliest questions for KBS) of whether PSM reuse may be better achieved by facilitating knowledge acquisition with a few coarse grained PSMs rather than developing libraries of highly specific PSMs (Paul Compton, personal communication). It also leads to the further suggestion that modelling should be secondary to KBS development rather than being a necessary precursor.

Having considered some of the implementations and variations to RDR we now consider why RDR were chosen as the foundation for this study and look in more detail at some of the basic features shared by RDR systems, particularly as they relate to the the issues that were found in Chapter Two to be important for *activity-reuse* and knowledge reuse in general.

3.2 Why Use Ripple Down Rules

Having given an overview of RDR it is worthwhile to consider why it was chosen as the knowledge acquistion and representation method on which to base further development. Four major reasons for using RDR include:

1. RDR has addressed a number of the problems associated with KBS which were described in Chapter One, namely the KA, maintenance and validation problems.
2. RDR offers a direct involvement system that gives the user control and ownership of the system which was seen as important requirement of KBS in Chapter One.
3. RDR provides knowledge in context, which was identified in Chapter Two as important in allowing knowledge reuse.
4. Of far lesser importance, was the availability of the existing software and accessibility to those most familiar and knowledgeable of RDR.

The first three points and the issues described in Chapter Two as relevant to the reuse of knowledge are looked at in relation to RDR in the subsections which follow and includes: intuitive KA and maintenance, knowlege in context, verification and validation of RDR KBS and knowledge abstraction.

3.2.1 Intuitive Knowledge Acquisition and Maintenance

The dramatic difference in the simple RDR approach to KA and mainstream approaches that require complex modelling hinges on the fundamental differences in the philosophical foundations of these approaches. RDR does KA with minimal analysis (Compton et al 1993) on the basis that experts use what is in their head to "make up" a solution to fit the situation. The situated cognition view was discussed previously in Chapter One. The capture of behavioural knowledge based on cases appeared to offer a solution which avoided trying to understand what went on inside the experts head.

Mainstream approaches to KA rely on the development of a model of the domain knowledge and reasoning processes of the expert. The development of such models is complex and difficult and requires a KE to mediate between the expert and the ES. The following example demonstrates that the role of the KE is viewed as such an integral part of KBS development that a system which suits the users more than the KE is seen as a negative feature. Hemmann (1993a, p.3), a developer of a KADS style methodology called MoMo, reports that the Framework of Problem Solving (FPS) he developed to support model-based diagnosis fits the needs of specialists in this area so well that it poses "an obstacle for knowledge engineers who typically are rather more generalists than specialists since they have to build different systems for many domains in their lifetime". If such a statement were made in the HCI community the reaction would be shock but in the KBS community it as accepted as a matter of fact and is indicative of the emphasis on having a KE as the mediator of the knowledge. In general, KBS research has become so focused on the development of knowledge level models by the KE that an alternative user-centred approach is seen as undesirable. Mainstream efforts seem to have lost sight of the well-observed and reported fact that users like to have ownership of their data and even more so of their knowledge, see section 1.1.4. Two approaches described below seem to have taken heed of the lessons being learnt from end-user computing and have found a way of circumventing the need for a KE.

Tuhrim, Reggia and Floor (1988) describe how the domain specialist can be used to directly populate an expert system and thereby improve system acceptance. They have developed the object oriented Knowlege Management System (KMS) which uses a high-level non-procedural language for KA. The expert develops the system using multiple inference methods and knowledge representations. The system is claimed to suit any semi-structured domain and to be portable. The user, however, does require some training to perform the task.

The second system, DARN (Mittal, Bobrow and de Kleer 1988) has two features in common with RDR. Modifications to the system may be made by experts without a KE and the system has been trialed on a diagnosis and repair task. An "expert interface" has been implemented to allow modification to the knowledge without a KE. Theory-based general purpose reasoning methods such as dependency directed backtracking and envisionment are used. The system is classification based consisting of an hierarchy of diagnostic states. DARN differs from RDR in that it is a plan-based system. It is suited to applications that have a short life and don't need stored data. DARN provides a framework for experts to review and revise.

In the study reported, hard disk technicians used knowledge which was presented as a plan of attack in fixing the problem. The domain expert first sets up a model. A plan is developed by the system in the form of graphs. The plan is broken into three elements: test, observation and action. Different user interfaces are offered one for the 'consumer' and another for the 'producer' of the knowledge. This is because the producers need to have an overview of the entire KB, whereas the consumer only needs a subset of the system. A browser has been implemented to assist repair plans. The KB could be modified via graph-plans. To assist the trainee technician there was a plan extension (provided in logical sequence) and history interface (provided in chronological sequence). The system was quick to set up. A significant remark made about the system is that due to maintenance problems the system was phased out. RDR is also very quick to set up, however, RDR were developed in answer to the maintenance problem and KA does not become more difficult as the KBS grows. KA time remains constant regardless of the size or life stage of the system. Edwards (personal communication) has even argued that maintenance time reduces as the system grows because there are less differentiating conditions between cases to consider as you move further through the ordered lists.

As in the case of DARN above, maintenance of typical KBS is a problem because of the undesirable side-effects which can occur as new rules are added. A new rule may interfere with the firing and applicability of a previous rule. RDR has overcome this problem because each rule is part of a chain of rules and is only applied in the context of cases that satisfy the same conditions found in the case that prompted the rule to be added.

3.2.2 Knowledge in Context

RDR stresses the importance of context on the appropriateness of conclusions. Context in RDR is preserved by the storing of the case that prompted a rule to be added. Difference lists are generated between the current case and the case/s associated with the rule/s that incorrectly fired. The use of difference lists in RDR allows the user to see how the two contexts differ and decide whether one or more of the differences justifies an alternate conclusion. In Figure 3.3 we see the MAKE screen in MCRDR for Windows where the user selects the conclusion and is then required to select attribute-value pairs to form the rule conditions. The current case is presented with the cornerstone case. The new rule can not be installed until all relevant cornerstone cases have been presented and a sufficiently specific rule developed to distinguish between the current case and all cornerstone cases in the case list.

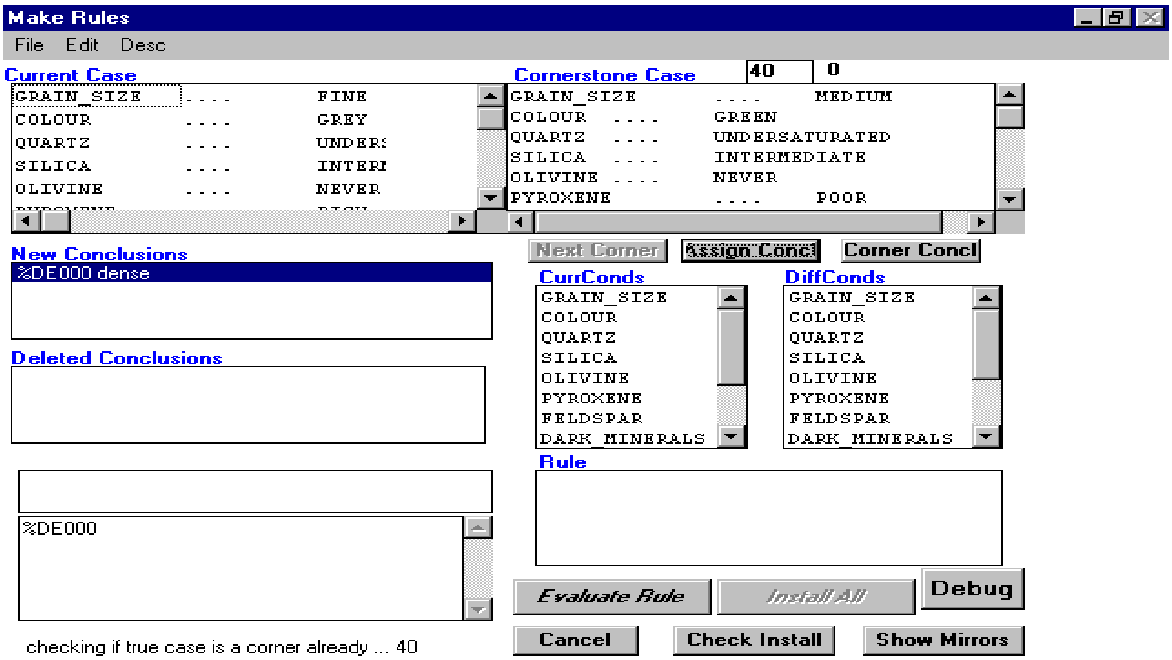


Figure 3.3: The Make screen in MCRDR for Windows where the user makes new rules.

By taking this approach we are asking "What is the best Recommendation for *this* Case ?" instead of the more general question that applies to more than one case "What is the best Recommendation for *these* types of Cases ? In conventional KA the second question is being asked so it is necessary to try to prespecify all of the contexts in which that particular recommendation would apply and more importantly it excludes any other context. The creation of global rules implies that the knowledge is explicit (Tyler 1978). This is giving primacy to the classification, rather than the context and means that the expert has the task of deciding whether the system's output fits the actual problem (Edwards 1996). The way that RDR only considers an individual real case moves the focus back to context and makes the decision applicable to that particular case. The emphasis in RDR is on local knowledge[9] rather than global knowledge.

Due to the RDR structure a rule trace can be used to provide a contextual explanation of why a conclusion has been reached. This is because with single classification RDR we can see not only why a rule has succeeded but also why other rules have failed. This feature is also possible in MCRDR by considering the branches that are not satisfied by the rule, but the decision of how many branches should be included and at what level of ordered list is not as clear. Also to some extent, the stopping rules in MCRDR provide a similar explanatory function to the false branches in single classification RDR. They show under what conditions a conclusion should no longer apply.

Returning to single classification RDR, while the whole pathway to a conclusion is shown, the XRDR implementation only includes the conditions (rule clauses) on the true branches as an explanation. Edwards comments:

> *This approach remains less than satisfactory for a number of reasons...the FALSE nodes evaluated by a case impact on the conclusion, and may contain valuable information for understanding the contexts in which a rule fires. Secondly the results generated depend upon the search statement entered by the user (Edwards 1996, p. 172).*

Handling of the false branches has been problematic in other areas too. In recent studies (Richards, Chellen and Compton 1996) that looked at removing repetition from single classification RDR KBS, using the Induct algorithm (Gaines 1989b) and rough set theory (Pawlak 1991), it was difficult to know how best to handle the false branches. In some cases the false branch represents a situation which must be false for the conclusion to apply but in other situations the rule fails because the rule is irrelevant for this case. This problem is discussed further in Chapter Four and is a reason why the MCRDR representation has been chosen for the work described in this thesis on reusable KBS.

[9] This view of local knowledge is not to be confused with the localization hypothesis (Geertz 1993) which holds that memory is a storage place for things such as words, images which are retrieved unchanged as required.

3.2.3 Verification and Validation of RDR KBS

Many approaches to KBS development attempt to build complete systems that are mostly considered final before being put into routine use. The need for complex modeling as a prerequisite to KA has resulted in the development of verification (Cragun and Streuduel 1987, Preece, Shinghal and Batarekh 1992, Suwa, Scott and Shortliffe 1982) and validation (O'Keefe and Leary 1993) (V&V) techniques that are designed for use before the system goes into routine use. There is little consideration of incremental validation for such systems and maintenance is often a neglected problem (Kang, Gambetta and Compton 1996, Menzies and Compton 1995 and Soloway, Bachant and Jensen 1987). Perhaps validation poses such a problem due to the difficulty of trying to validate models that are by their very nature inaccurate (Clancey 1991a).

RDR does not distinguish between initial KA and system maintenance. RDR develops the whole system on a case by case basis and automatically structures the KBS in such a way to ensure changes are incremental. Validation is designed to be performed on-line by the expert. FCA and Repertory Grids (Gaines and Shaw 1989) also minimise the role of the knowledge engineer and make use of differences but as with most KA approaches require some consideration of the whole domain and do not consider incremental maintenance. With FCA and Repertory Grids, incremental maintenance is often addressed by regenerating implications associated with the revised data set. In the case of repertory grids this is not a major problem as the rules are automatically generated.

The validation of rules provided by the cornerstone cases is not total validation. However, the cornerstone cases ensure that the new rule is sufficiently specific and different from the rule that gave the misclassification to assign the new classification to the current case but not to cover, which would alter the classification of, previously correctly classified cornerstone cases.

Extensions to the RDR approach to verification and validation include computer-assisted validation using formal concept analysis (Wille 1982) and off-line verification, using rough set theory (Pawlak 1991) and are described in Sections 7.3.1 and 4.7, respectively. The general RDR approach to V&V has been briefly presented in this section. For further discussion and evaluation of incremental verification and validation of RDR see Kang, Gambetta and Compton (1996).

This section would not be complete without some consideration of the V&V literature. Verification has traditionally been broken up into basically two categories: consistency and completeness (Gonzalez and Dankel 1993). Using the framework provided by Nguyen et al (1987), for consistency we must consider: redundant rules, conflicting rules, subsumed rules, unnecessary premise conditions and circular rule chains. Completeness involves detection of unreferenced attribute values, illegal attribute values, unachievable intermediate conclusions, unachievable (final) conclusions, or goals and unachievable premises. With Santos, Gleason and Banks (1997) we have found that the "push for *consistency and completeness* as the goal of verification is an artifact of rule-based inferencing mechanisms, and not necessarily a requirement for all knowledge-base verifications" (Santos, Gleason and Banks 1997, p.14). Santos, Gleason and Banks are interested in providing an incremental V&V technique and argue that if we tolerate or even expect our KBS to be incomplete we can not enforce the constraint of completeness. While the conventional list of verification anomalies can not be ignored the full list does not apply to the RDR exception structure. Also the approach to the detection of such anomalies and the decision of what should be done about them may differ. For example, repetition in an RDR KBS is unavoidable as knowledge is patched locally and the same patch may need to be applied in multiple locations. We now review the various verification anomalies identified by Nguyen and consider whether they apply to RDR and how they can be addressed.

3.2.3.1 Completeness

The attributes and values used in an RDR KBS are directly related to the cases being used. There is no predefinition of what values or even what attributes are expected which is in keeping with the RDR philosophy that knowledge is evolving and never complete and the desire to keep the KA task simple and analysis to a minimum. The case driven nature of KA in RDR means that any value found in a case is valid and there is no restriction on or presupposition of what values or even what attributes are valid. This freedom, however, means that the first anomaly, unreferenced attribute values can occur. The second anomaly, illegal attribute values, occurs only when a data entry error has been made as any values that have occurred in the real world are legal values. These anomalies, however, relate to the quality of the cases rather than the need for a verification technique as part of the KA/maintenance process. These problems are addressed by obtaining sufficient cases and by ensuring the cases input are checked for accuracy. In most domains, especially medical, checking of input is a mandatory requirement regardless of the KA approach. However, in RDR the emphasis is on incremental KA which means that cases are dealt with as they arise and it is not necessary to have a full set of cases before KA and usage of the system commences.

Some work has been done to detect what values in a case may indicate an abnormal value by (Edwards 1996) in his work on *Exceptions*, where valid value ranges are maintained by the system and any new values outside of the range are flagged and reported to the user as possibly erroneous. An immature KBS will give many warnings and high user involvement is expected at the beginning. The major frustration to the user occurs when a mature KBS is still giving false positives due to slight variations in values. To speed up this maturation process and reduce the notification of false invalid values the ranges can be preset to expected ranges. The drawback of this, however, is that this may be difficult and the person specifying the ranges must be a true expert familiar with the cases otherwise the ranges themselves may be invalid.

The three last completeness anomalies all concern rules that can not fire due to unachievable intermediate or final conclusions or unachievable premises. Firstly, there are no intermediate rules in RDR, each rule represents a complete pathway through the knowledge capturing the lower level operational knowledge without the use of higher level abstractions which intermediate rules provide. The exception structure and the use of cases as the impetus for KA and maintenance ensures that each new rule is added in context and that there is a situation in which each pathway will be traversed. Recent work by Barr (1997) shows the value and importance of having cases that exercise each conclusion, node and class and advocates building a directed graph of rule pathways that tests this has occurred. If it has not, then she advocates adding cases to the test set to cover those parts not being exercised. Such an approach is automatic in the RDR paradigm and can be further strengthened by keeping all seen cases, not just cornerstones, for rule evaluation.

There is another aspect of completeness that is more of a validation issue. This concerns how much of the domain is covered by the rules. This can not be found through the detection of the above anomalies but the work by Edwards (1996) on providing system *credentials* and subsequent work using simulated experts to perform this task (Compton 1996) does address this concern. The idea behind *credentials* is that if we gather statistics on how often each rule has fired correctly, how many corrections have been made to the rule and how recently modifications have been made we can get some idea of the rules stability. If we gather the same statistics by conclusion then we can get a feel for the stability and coverage of the knowledge for the whole domain and subdomains. If a KBS is seeing 100 cases a day and requiring 5 modifications then we can say that the KBS is 95% accurate and complete. This percentage can be finer-tuned by determining if certain parts of the KBS, which may cover a more recently introduced subdomain of the knowledge or a subdomain where knowledge is still emerging, are the parts that are being corrected. The more established subdomains may show 99%-100% completeness. In the PEIRS system there were a number of subdomains, including blood-gases, thyroid, liver functions, endocrines and catecholamines, that were added incrementally over the four years to make up the 2000+ rule-base.

At some point in time within a mature subdomain we may have seen cases that cover every combination of Attribute-Value (A-V) pairs. Thus, when the A-V space has been covered for that subdomain the knowledge is complete. Using *exceptions* and *credentials* such statistics are being kept so it can be determined when this has been achieved if a list of possible values is enumerated beforehand. The benefits of catching unreferenced or illegal attribute values and determining the completeness of the A-V space using this list must be traded off for the flexibility and ease of KA currently available.

3.2.3.2 Consistency

Now let us consider checks for consistency. The problem of circular rules does not exist in an RDR KBS because each rule provides a final conclusion so the problem of the conclusion of one rule being used as the premise of another rule which points back to the original rule does not occur.

Subsumed rules are an anomaly which can occur in the RDR structure. The use of local patching by exceptions can result in pathways where the rule conditions of one pathway subsume the rule conditions of another pathway and both rules have identical conclusions. This can occur because a rule that, for example, gave the conclusion %VC000 has been amended in the context of a particular case where a different condition present in the current case but not in the cornerstone case of the rule being amended warrants a different conclusion. We can then get a later situation where a new case should have the original conclusion %VC000 because although it also has all the same conditions that form the pathway for the previous exception rule the presence of a different attribute-value pair means that the previous exception is overridden and the amended conclusion does not apply any longer. This situation can occur in different parts of the KBS because the order in which cases are seen will affect the structure and which conditions are chosen first. The problem of subsumed rules is related to the redundancy problem discussed in more detail below. Excessive subsumption is undesirable but it is argued that the history provided by this chain of rules is valuable and also results in the ability to draw the subsumption lattices described in section 5.2. A rule that subsumes another rule can be seen as a higher concept of the subsumed rule and thereby provides us with some information of the abstraction hierarchy which exists implicitly within the knowledge.

Another type of redundancy that can occur is unnecessary premise conditions which occur when we have two rules that give the same conclusion but the condition in one negates the condition in the other while all other conditions are the same, as in the example :

Rule 1: If A=X and B=Y then C=Z   Rule 2: If A=X and B=NOT Y then C=Z

The second condition in both rules is unnecessary and the rules can be combined to form one rule: If A=X then C=Z.

These types of redundancies are unavoidable in the RDR exception structure due to the use of local patching. The problem of repeated and redundant knowledge is described further in section 3.4.1 and work to reorganise RDR KBS is described in Chapter Four.

3.2.4 Knowledge Abstraction

The need to abstract knowledge for the purposes of reuse has been espoused by many and has proven to be important in supporting the reflective modes of use. The key difference between abstraction offered in most rule-based approaches and RDR is that the higher level concepts are derived using formal concept analysis rather than defined by the user. RDR KBS are made up of primitive concepts in the form of attribute-value pairs and it is through the intersection of conditions that non-primitive concepts are found. Derivation of these concepts and how they are used to develop a subsumption hierarchy are described in detail in Chapter Five.

In sections 2.1.1 and 2.2.2 the use of metarules by Clancey (1992) and Richards (1995) were given as ways of implementing abstraction. There is no such feature in RDR. Prior to the work reported in this thesis the only form of abstraction can be seen in the use of user-defined reusable functions (Preston et al 1994) that serve a similar function to metarules. An example of such a function is found in Figure 3:4 below. In earlier implementations of RDR, such as PEIRS, functions were built by the programmer and could not be changed by the user. If a modification to a function was required, this change needed to be input at each node that was to be altered. There was no facility to globally patch or reuse the definitions input at a different node. In Time Course TCRDR the user was able to define these functions and give them a name so that they could be reused. New functions can be added at any stage because they will only apply to later rules. Local patching allows implicit refinements in functions and ensures that only cases traversing the same path will be affected. The functions appeared in the difference lists and could be reused.

```
curr(x)        : x[-1000]

HIGH(x)        : curr(x)  >  RH(x)
LOW(x)         : curr(x)  <  RL(x)
NORMAL(x)      : (curr(x) ≥ RL(x)) & (curr(x) ≤ RH(x))

DECR(x)        : curr(x) < MAX(x)
INCR(x): curr(x) > MIN(x)
```

Figure 3:4: Some user defined functions used by PEIRS
(extract from RDR Engine Reference Manual - Part B)

SICK(x) : (x = AEC) | (x = ITU) | (x = 13CLA) | (x = STV)
WELL(x) : (x = END) | (x = OPD) | (x = CLIN) | (x = SYN)

Figure 3.5: Implementing abstraction through features in RDR.. The location of the patient is seen to be an indicator of the patients general state of health.

These functions can produce abstractions of many types resulting in higher level abstractions which were also a means of reducing repetition of knowledge. One such abstraction was the use of the location of the patient as an indicator of whether the patient was sick or well. In figure 3:5 you can see that if a patient is in "AEC", "ITU", "13CLA", "END" or "OPD" wards they are sick. Alternatively, if the patient attends "STV", "CLIN" or "SYN" (outpatient clinics) they are considered well. Edwards notes:

> *"With more complex time course data it becomes even more difficult to predetermine the type of data abstractions required. In these complex domains, therefore, the value of reusable expert-defined function definitions is further reinforced" Edwards 1996, p.154*

While the use of abstraction will reduce repetition and ease the KA burden, Edwards (1996, p.152) makes the further point that:

> *"abstraction will also obscure the contextual differences between attributes. These differences may be immaterial, but it is possible that cases will arise where the difference becomes important".*

In order to allow functions to be redefined, Menzies (1992) has suggested date and time stamping of changes in function definitions so that only rules added after the change are affected by the change. Other research is considering alternative approaches to reconciling feature refinements. The work on Nested NRDR mentioned in Section 3.1.3 is relevant but does not provide validation. It is envisaged that a secondary RDR KB can be used to handle these functions and act as a supervisor of the domain knowledge.

3.2.5 A Summary of the Features of RDR

From the previous discussion, RDR can be seen as a type of case-based reasoning where rules identify salient features and the exception structure is an index. There is an emphasis on the role of the user as the owner and user of the system and KA and maintenance are simple tasks that involve determining the correct conclusion and identifying the key features in the case that prompted the new rule. Knowledge is seen as context-dependent, evolving and dynamic and so the concern for user-maintained systems is important.

RDR offered a good starting point from which to pursue *activity-reuse* because it was a user-centred system that supported the idea of a flexible, user-adaptable environment and many of the activities, such as KA, maintenance, validation and inferencing, were all ready well tested and were offered within the one system. These activities form what is referred to in this thesis as the reflexive modes that require less thought on the part of the user and were generally performed automatically or subconsciously. RDR also offered a way of maintaining the contextual knowledge which was deemed important when reusing knowledge for another purpose.

While RDR differs considerably from mainstream approaches its philosophical basis and use of cases is not unlike KBS research based on personal construct psychology (Kelly 1955) and case-based reasoning (CBR). A number of other theories were also explored during the course of this study and we now turn to consider other theory relevant to RDR.

3.3 Other Theory Relevant to RDR

RDR represent a paradigm shift in KBS development from the mainstream approaches described in Chapter Two which rely on the detailed analysis of the domain and PSM knowledge and the development of KBS by knowledge engineers. Nevertheless, RDR is not the only KBS technique that aims to minimise the complexity of KA and to allow direct interaction between the system and end-user. Of most note are the systems that have been built based on Personal Construct Psychology (PCP) (Kelly 1955) and these will be described and compared below. Another area that is relevant to RDR is case-based reasoning (CBR) as the reliance and use of cases in RDR makes RDR a form of CBR. Work that was performed on developing a k-distance weighted nearest neighbour algorithm for finding the closeness of concepts is also reported in the section on CBR since the use of a nearest-neighbour algorithm for rule-based reasoning is similar to their usage in CBR to match and evaluate similarity between cases.

In the search for a way to find higher level concepts and the relationships between them a number of theories were considered which included Hamming distance (Hamming 1980), cluster analysis, Rough Set Theory (RST)(Pawlak 1982) and Formal Concept Analysis (FCA)(Wille 1982). On further investigation of Hamming distance and cluster analysis it appeared that the suitability of these approaches for this problem was limited. It appears that Hamming distance is useful to determine if a coding error exists and is useful when considering an individual A-V pair, but inappropriate when the problem concerns the combination of A-V pairs and the closeness of classifications in conclusion space, not just data space. While cluster analysis seemed relevant it did not offer the ability to find abstractions and structure them which was being sought. An outcome of the review of cluster analysis techniques (and CBR) is found in Section 3.3.2.1 on rule-based reasoning which describes a nearest neighbour algorithm that can be used to cluster rules and/or conclusions. RST and FCA, however, did prove more useful and they are described in Chapters Four and Five, respectively.

The theories covered are obviously a very limited selection of the techniques that can be used for finding concepts. In particular, there is a vast literature on data mining and knowledge discovery that could be useful for this purpose. The point that must be stressed regarding the theories that were looked at is that many methods (though not AQ (Michalski and Chilausky 1980) or CUT95 Scheffer 1996), are based on statistics, information theory or bayesian probability and are affected by the numbers of examples of each data profile represented by the cases. Since the focus here was on finding interesting concepts from knowledge in the form of rules it was important that the number of cases of each profile was not a key element of the theory. Very little has been done to analyse knowledge bases for the purposes of reuse or understanding of the concepts therein (Hemmann 1993a). This is probably because the assumption has been that models are prerequisites and therefore already available. The work reported on rough set theory[10] and formal concept analysis is novel because of their application to rules rather than data and even more unique in their use in conjunction with RDR.

Another aspect of some of the theories covered here is their similarity to the RDR use of difference lists for knowledge acquisition. Personal Construct Psychology (PCP)(Kelly 1955) uses repertory grids (Gaines and Shaw 1989), RST uses a discernability matrix and some CBR techniques (e.g. CYRUS Kolodner and Simpson 1984, PROTOS Bareiss 1989) use differences as their basis for comparison. In support of the use of differences the following quote from Clancey appears appropriate:

> "*apparently, the very business of perception is to view the world conservatively (noticing only what is different) in order to adopt previous successful ways of behaving" (Clancey 1991b, p.256).*

Formal concept analysis looks at similarities rather than differences in deriving formal concepts from a formal context. However, it has been shown (Gentner and Markman 1994, Markman and Gentner 1996) that differences only become important if there are similarities or *commonalities* in the first place. Differences which are based on shared features are called *alignable differences* and can only be determined when the commonalities of a pair are known. When we view differences and similiarities in this light we see that all of the approaches mentioned are concerned first with determining which concepts, be they rules, cases, elements, rows or objects, are alike and then finding the differences between them.

---

[10] An exception is research by Colomb and Sienkiewicz (1995) which used RST to reduce the "inflation" that can occur when a propositional ES is converted into a decision table.

We look first at PCP which uses triadic elicitation to compare two similar elements against a third dissimilar element as means of exploring and capturing conceptual models. As will become apparent PCP and RDR share many features and beliefs in common. We look at these features now.

3.3.1 Personal Construct Psychology[11]

Ripple-Down Rules (RDR) research has focused on building KBS that exhibit expert behaviour without committing to the debate on how the task is being achieved by the expert or trying to capture the conceptual model of the expert. As noted in other chapters, the main reasons for this approach are the inherently unreliable nature of models (Clancey 1991b, Gaines and Shaw 1989) and the situated nature of expertise and human action (Clancey 1997, Collins 1997). Approaches based on Personal Construct Psychology (PCP) (Kelly 1955) also do not commit to how the human reasoning is being done. Gaines and Shaw only accept that elicitation allows self-modelling and the "tacit knowledge" acquired is "a theoretical construct imputed to an agent displaying *intelligent* behaviour by an observer" (Shaw and Gaines 1991b, p.13). Gaines and Shaw point out that coming to terms with understanding and encoding the way that experts think is problematic and that we need to learn a lot more about how the mind works before such questions can be answered.

The acceptance that models are difficult and the use of simple techniques for eliciting knowledge directly from an expert without the mediation of a knowledge engineer (KE) by both PCP and RDR suggests that the two approaches share a common philosophical basis that is not shared by most mainstream KBS research. The interest in cases and the use of difference lists in RDR and the dichotomy corollary of PCP are further similarities between the two. The work described in Chapter Five that employs FCA to develop a subsumption hierarchy of the concepts that exist implicitly in an RDR KBS offers a counterpart to the intensional logics and visual language (Gaines 1991b) supported by PCP.

Kelly (1955) found that patients were able to describe what was different between situations or people without being able to formally describe their mental state. In the theory, people develop "personal constructs[12]" or templets through which they anticipate (which is used to include understand, predict and control) the world (Shaw and Gaines

---

[11] Section 3.3.1 follows the paper Richards, D. (1998) Ripple Down Rules with Formal Concept Analysis: A Comparison to Personal Construct Psychology *11th Workshop on Knowledge Acquisition, Modeling and Management,* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:KAT-4.

[12] "The construct is a basis of making a distinction…not a class of objects, or an abstraction of a class, but a dichotomous reference axis" (Kelly, 1970).

1992). From Kelly's geometrical perspective he sees that humans classify elements, which can be seen as examples, and place them in a psychological space. They do this using their constructs which provide a bipolar axis of reference used by people to differentiate between elements. The range over which a construct can be used is known as the range of convenience.

Gaines and Shaw have developed a logic and visual language based on the geometry which is briefly described here and in more detail in section 3.3.1.3. Kelly's notion of a personal construct can be represented by the above structure in Figure 3.6a which is a triple of two disjoint distinctions that are mutually subsumed by a third. The non-directional line between **b** and **c** shows that **b** and **c** are disjoint and form the poles of the construct. The arrows pointing from **b** and **c** to **a** indicate that **a** subsumes both concepts[13]. Since **a** subsumes a dichotomy it is the range of convenience. For example, in Figure 3.6b the elements being considered could be potential houses for purchase. A construct could be the size of the yard. The bipolar axes of the construct could be large and small. Since the concept of yard subsumes the concepts of small to large yards it is the range of convenience.

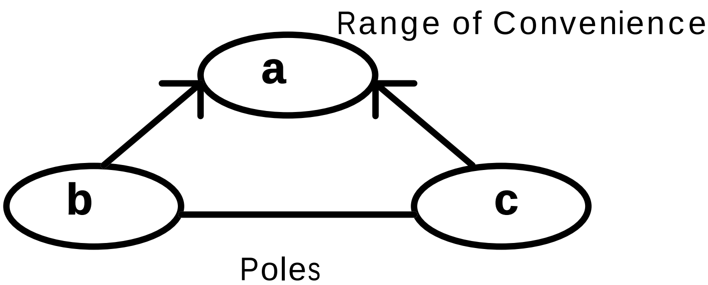


Figure 3.6a: The Triple of Distinctions Generating a Construct (From Shaw and Gaines 1991b)

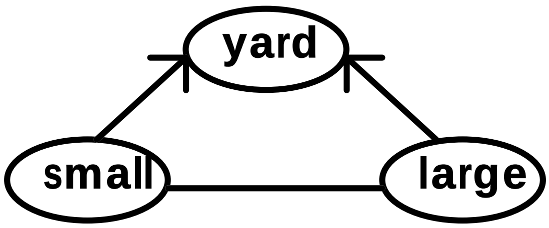


Figure 3.6b: The Yard Construct in the Houses-for-Sale Domain

[13] A concept in PCP is similar to the earlier notion of a percept which carried the connotation of it being a personal act peculiar to the individual and also the current notion of a concept as an abstraction not something concrete.

As a means of acquiring the relevant constructs and elements, Kelly developed the repertory grid technique. The expert is asked simple questions that indicate how well each construct fits each member of the solution set. The repertory grid is a way of finding concepts, their structures and relationships between them without directly eliciting them (Gaines and Shaw 1993a), as shown in Figure 3.7. They can be more successful than attempting to directly elicit a model because:

> *"The repertory grid was an instrument designed by Kelly to bypass cognitive defences and give access to a person's underlying construction system by asking the person to compare and contrast relevant examples" (Gaines and Shaw 1993a, p.52).*

The repertory grid in Figure 3.7 includes five constructs and three elements using the contact lens prescription domain which we use in a number of further examples to aid in comparison of the various representations of this knowledge shown in this section.

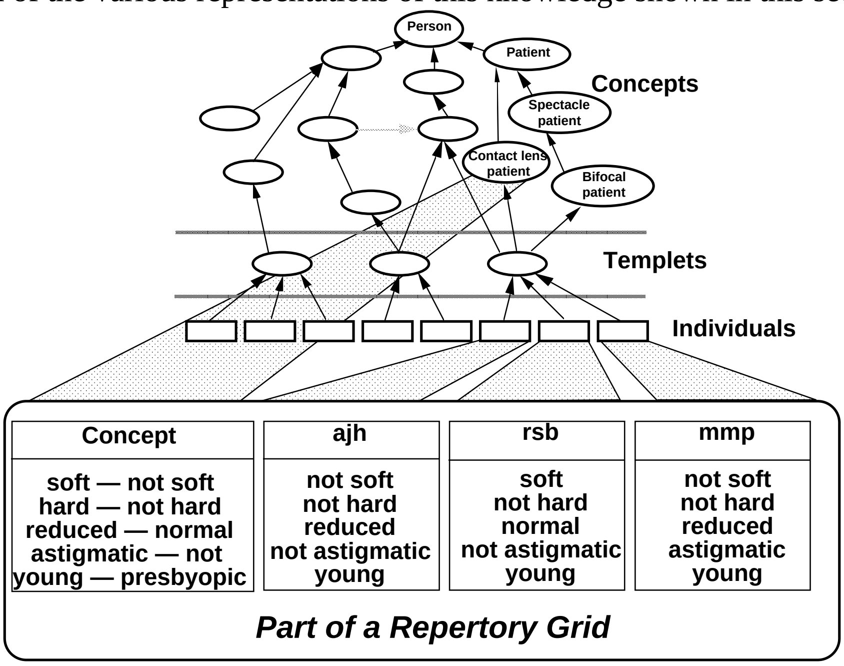


Figure 3.7: The repertory grid as a matrix of concepts, individuals and constraints (from Shaw and Gaines 1992). Note that "concept" at the bottom left should read "construct". Ajh, rsb and mmp are the initials of the patient.

The repertory grid technique has found widespread success and continues to be used both in manual and computerised form (Beail 1985) by educationalists, clinical psychologists and managers (Shaw 1980). A number of systems have been built based on Kelly's ideas such as Knowledge Support System Zero (KSS0), Expertise Transfer System (ETS) and AQUINAS (Boose et al 1989). Gaines and Shaw (1993a) offer the following general framework to assist the elicitation of conceptual structures using repertory grids. The particular supporting tool in KSS0 is given in brackets after the general description.

1. Careful definition of the purpose, the subdomain and context to ensure that the context doesn't change.
2. Selection of stereotypical cases. A full range of cases need to be specified.
3. A wide range of typical KA techniques are employed to gather initial elements or distinctions such as interviews, brainstorming sessions, keyword extraction from text, protocol analysis.
4. Feedback is given along the way by showing the conceptual structures being developed and asking for further distinctions to separate close cases (Elicit).
5. Triadic elicitation is used to randomly select three individuals to state how two are alike and how the third differs (Elicit).
6. Hierarchical and spatial cluster analysis are used to show the structure between concepts and for feedback (FOCUS and PrinCom).
7. Rule induction is performed to derive potential implications between concepts and to find higher level concepts and the subsumption relations (Entail).
8. The concept structure may be elicited directly and combined with indirect development (supported in AQUINAS (Boose et al 1989)).

It is clear from the above framework that feedback and alternative views of the conceptual models derived are integral. Feedback is discussed further in 3.3.1.4. The range of different ways of looking at the concepts helps to "maintain the expert's interest and to explore his or her psychological space" (Gaines and Shaw 1993a, p.71). The KSS0 system is non-modal so that the user can move to whatever part of the system as they choose. In summary,

> *"The psychology has the advantage of taking a constructivist position appropriate to the modelling of specialist human knowledge, but basing this on a positivist scientific position that characterises human conceptual structures in axiomatic terms that translate directly to computational form" (Gaines and Shaw 1993a p. 82).*

3.3.1.1 A Comparison Between PCP and RDR

In this section we compare PCP and RDR and include some comments regarding FCA. Although FCA is introduced in Chapter Five and is not the main focus of this section its compatibility with the two approaches is interesting and confirms the suitability of combining the techniques[14].

---

[14] This thesis reports on the combination of RDR and FCA. Work has also been done by a number of people on the combination of PCP and FCA. One such example is Spangenberg and Wolff (1988).

PCP and RDR offer alternative simple ways of acquiring knowledge accessible to an end-user. However, the nature of the knowledge that is captured is different. As mentioned earlier, the RDR technique develops an assertional KBS or A-box. Using FCA we derive a terminological or T-box KBS based on the MCRDR A-Box. PCP has a different starting point. PCP captures a conceptual model from which a terminological KBS is extrapolated. This terminological KBS provides an anticipatory system for the computer, in keeping with Kelly's view of humans as an anticipatory system. In some of the computer implementations of PCP, Induct is used to derive an A-box from the T-box developed using the repertory grid technique. Regardless of the different starting points, both approaches differ greatly from current approaches that look at building terminological KBS using ontologies (Guha and Lenat 1990, Patil et al 1992, Pirlein and Studer 1994) or general problem solving methods (Chandrasekaran and Johnson 1993, Schreiber, Weilinga and Breuker 1993, McDermott 1988, Steels 1993, Puerta et al 1992) due to the emphasis these latter approaches place on complex modelling as a prerequisite to the capture of domain knowledge which also necessitates using a KE as an intermediary between the expert and the computer. The strength of systems built on PCP is the assistance they give the expert to understand the concepts and relationships of the knowledge they are dealing with, without having to explicitly describe their conceptual or problem solving model and the ability to directly interact with the system without a KE. Similarly the strength of systems built using RDR is the ease and intuitiveness of simply assigning a conclusion and picking some relevant features without the need for a KE.

Just as systems built on PCP and RDR share similar strengths they also suffer from similar limitations. It has been found that ETS and a descendant system KAQ is limited to classification problems and it was also difficult to extract causal, procedural or strategic knowledge (Boose et al 1989). Current research using MCRDR for construction addresses this first problem (Compton et al 1998 and Ramadan et al 1997a) and work by Lee and Compton (1995) has looked at deriving causal explanations from RDR KBS. Since RDR and PCP aquire the same type of knowledge, although the knowledge is used differently, it is expected that PCP could be adapted in a similar manner to RDR to handle configuration problems.

3.3.1.2 The Importance of Context and Cases

PCP, RDR and FCA place a strong emphasis on the importance of knowledge in context, a view supported by much of the knowledge reuse community (Guha and Lenat 1990, Patil et al 1992). Context is used in KSS0 to focus attention in the elicitation process. FCA is also:

"guided by the conviction that human thinking and communication always take place in contexts which determine the specific meaning of the concepts used" (Wille 1996, p. 23).

The RDR focus on context is based on a socially situated view of knowledge and supported by the exception structure and storing of cornerstone cases. Kelly's view of templets and the construction corollary[15] reveals a situated view of the way humans perceive events:

"Man looks at his world through transparent templets which he creates and then attempts to fit over the realities of which the world is composed" (Kelly 1955, pp.8-9).

Kellys sociality corollary[16] does not place as much stress on the impact of social influences on human action compared to the socially situated view of human action. However, Kelly does go on to say that the role a person plays is:

"an ongoing pattern of behaviour that follows from a person's understanding of how the others who are associated with him in his task of thinking" (Kelly 1955 in Shaw 1980, p.22).

In a similar vein, PCP, RDR and FCA do not consider the knowledge captured to be globally applicable but relevant or 'convenient' within the given context as discussed in the range corollary[17]:

Another key similarity between all three approaches is the use of cases or examples to elicit knowledge. Cases are beneficial because:

"from a psychological point of view, cases are abstractions of events or processes with temporal and spatial tabs …[and] are models of the original experience" (Woodward and Shaw 1993, p 13-8).

In the repertory grid technique stereotypical cases are seen as a critical part of eliciting conceptual models and a good set of cases will lead to a good and compact set of rules. Repertory grids assume that people are better able to offer good examples than define some globally applicable rules or heuristics. In stereotypical cases there is less likelihood of non-significant attrbutes. The cases used by RDR tend to be historical or actual cases which clearly have irrelevant attributes. Thus, although each case that prompts a rule to be added is stored, no claim is made regarding the significance of that

[15] The construction corollary states: "a person anticipates events by construing their replications" (Kelly 1955).

[16] The sociality corollary states: "to the extent that one person construes the construction processes of another, he may play a role in a social process involving another person" (Kelly 1955).

[17] The range corollary states: "a construct is convenient for the anticipation of a finite range of events only" (Kelly 1970).

case. It is acknowledged that if a "good" representative set of cases exists then the KBS can be built quicker and will mature sooner. In such a situation it may be better to make use of a suitable machine learning algorithm for KA. In RDR the cases are being used to assist the user to define the key features of the case, whereas in PCP the user is asked to imagine a case and identify the key differences and similarities between them. In the case of PEIRS the use of actual cases was natural because they were available and the problem was to add an interpretation to them. The view of cases in PCP and FCA is more alike and there is a similarity between the PCP repertory grid and the FCA crosstable. We can view the constructs and elements in PCP as analogous to the attributes and objects in FCA, respectively. In FCA the use of cases is similar to that of PCP, that is to uncover a conceptual model and derive some implications. In RDR the purpose is to acquire rules which we later use to derive the conceptual model using FCA.

In all approaches, cases assist the user in defining properties that allow cases to be distinguished from one another. In PCP triadic elicitation requires the user to add constructs which describe how two elements are alike and how the third element differs. In RDR difference lists are used to assist the user with picking features in a case that differentiate between the current case and the cornerstone case associated with the rule that gave the misclassification. In PCP it is necessary to look at the overall clusters. In RDR the concern is with individual cases. The role of cases in providing an extensional definition for a concept in FCA and PCP is discussed in the next section where we look at how the logical foundations and visual language based on PCP compare with the concept matrices and lattices developed using FCA.

3.3.1.3 Logical Foundations, the Visual Language and Implications

Gaines and Shaw state:

> *"there is a very direct relationship between the psychological conceptual structures presupposed in personal construct psychology and those implemented in term subsumption logics, and this is highly significant in supporting expertise transfer as a process of modelling the basis of human skilled performance in operational terms " (Gaines and Shaw 1993a, p.51.*

Shaw and Gaines further describe Kelly's geometry as:

> *"an intensional logic, one in which predicates are defined in terms of their properties rather than extensionally in terms of those entities that fall under them" (Shaw and Gaines 1994, p. 259).*

Kelly's notion of a distinction may be used to carve out regions of psychological space that can be compared to regions carved out by other distinctions according to the subsume $\rightarrow$ and disjoint — relations, see Figure 3.6(a) and (b) and the discussion in Section 3.3.1. The subsumption relation is asymmetric and transitive and supports Kelly's organisation corollary[18] by providing a partial ordering of distinctions (Shaw and Gaines 1991a). As described in section 5.1, FCA uses the subsumption relation $\leq$ to order the set of all concepts. In FCA, concepts that can be reached by ascending or descending paths can be compared in terms of the subsumption relation. In the intensional logic developed for PCP, the intersections of primitive concepts form non-primitive concepts from which the subsumption relation may be derived. The approach in this thesis is to use the MCRDR rules as the objects which form the primitive concepts. Using FCA the intersections of sets of attributes (rule conditions) and the set of objects (rules named by rule number and conclusion) that share those attributes are found to form the non-primitive higher-level concepts. These higher level concepts in the MCRDR/FCA implementation described in Chapter Five can be labelled by the user if they wish and combined with the primitive concepts to form the concept lattice. Similarly, Kelly describes the intersection of sets of attributes as part of his theory of anticipation:

> *"What one predicts is not a fully fleshed-out event, but simply the common intersect of a set of properties" Kelly 1955).*

Like the concept lattice derived using FCA, a semantic network or overall task or domain ontology can be found by determining the ordinal relations between concepts derived from the constructs and elements in PCP (Gaines and Shaw 1993a). This aspect is discussed further in Section 8.1.2.1. Gaines and Shaw (1993a) further point out the 'is-a' relation may be computed for non-primitive concepts whereas the 'is-a' relation is defined for primitive concepts.

Kelly's dichotomy corollary[19] is supported by the disjoint relation. It is a symmetric relation that is asymmetrically defined because the sequence in which terms have been defined may result in reference to a term which has not yet been defined (Shaw and Gaines 1991a). In geometrical terms a disjoint relation occurs where regions do not overlap. In FCA the disjoint relation can be seen as $(X_1,Y_1) \cap (X_2,Y_2) = \varnothing$. In terms of the concept lattice a disjoint relation between two concepts exists where paths of the two concepts do not intersect. Note that although all paths intersect at the top or bottom concept the intersection set may be empty.

---

[18] The organisation corollary states "each person characteristically evolves, for his convenience of anticipating events, a construction system embracing ordinal relationships between contructs" (Kelly 1955).

[19] The dichotomy corollary states "a person's construction system is composed of a finite number of dichotomous constructs" (Kelly 1955)

From Gaines and Shaw (1993a), using the subsume and disjoint relations four possible binary relations can be formed: a→b, b→a, a—b, or none of these. If the two subsumption relations hold they form an equivalence relation on distinctions. The disjoint relation is inherited through subsumption so that:

a—b and c→a ⇒ c—b

These relations are also present in the FCA concept lattice. An equivalence relation would be shown as a merged concept where the intent includes all the attributes of the two concepts and the extent covers all the objects associated with the two concepts. The inheritance of disjoint relations via subsumption is obvious since any concept that subsumes another will also be disjoint from any concepts from which a subsumed concept is disjoint.

Shaw and Gaines (1991b) state:

> *"a visual language that is both comprehensible and formal offers attractive possibilities not only for the comprehension but also for the editing, and for parts of the elicitation process itself" (Shaw and Gaines 1991b, p.9).*

As can be seen in Figure 3.8, the visual language mentioned in Section 3.3.1 offers a graphic representation of the subsume and disjoint logical relations. The concept lattice is the visual counterpart in FCA. Both approaches seek to tap into the psychological benefits associated with graphics and that of semantic networks in particular. The concept lattice structure offers:

> *"hierarchical conceptual clustering of the objects (via the extents) …. and a representation of all implications between the attributes (via its intents)" (Wille 1992, p. 497).*

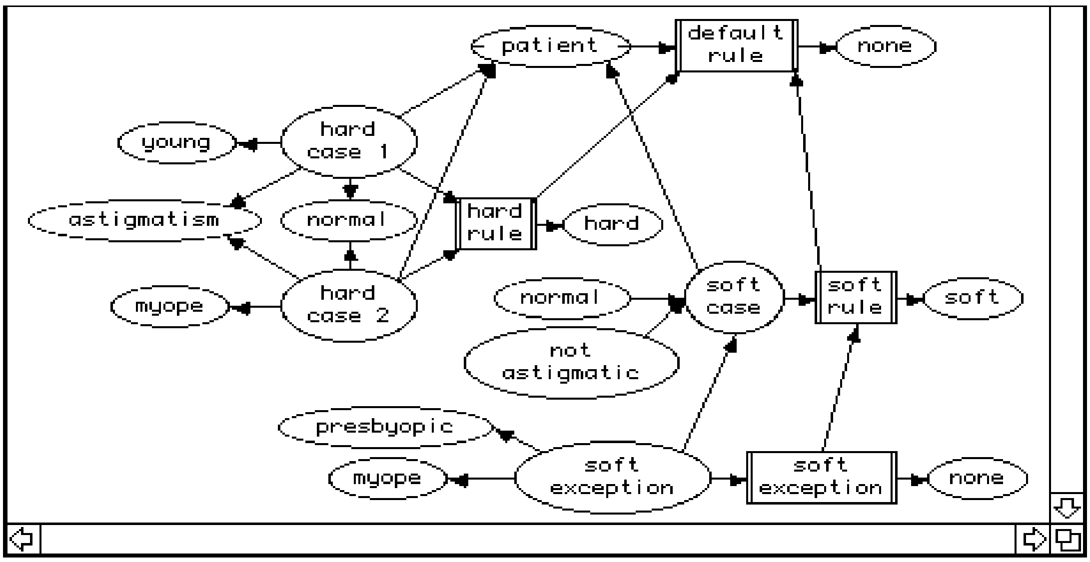


Figure 3.8 Contact Lens Rules Represented in the Visual Language (from Gaines and Shaw 1993a)

Shaw and Gaines (1991b) make the point that such a representation is not unlike other work, such as that by Cruse, that manually produces diagrams but that the computational representation allows deductive inference as well as graph-theoretic analysis. This point and its relevance to the FCA line diagram as a taxonomic representation is taken up again in Section 8.1.2.1. However, it is noted that the use of anticipations as rules to support inferencing is not a necessary but a possible relation. In FCA through the techniques known as attribute (Wille 1989b) and concept exploration (Stumme 1997) it is also possible to derive implications. However, concept exploration is still under investigation and both techniques can be time-consuming for the user requiring a process of evaluating, accepting or rejecting possible implications and the offering of suitable counterexamples if the implication is rejected. In RDR the concept lattice is not used to derive rules but as an alternative higher-level model of the rules already captured. The RDR approach to capturing rules may be more manageable for the user than the definition of counterexamples.

A major difference between the KL-ONE (Brachman 1979) like approach used by Gaines and the use of FCA is that FCA also describes concepts extensionally. This aspect has been criticised since an intensional definition implies an extensional definition but the converse is possibly but not necessarily true (Zalta 1988). In this thesis, the objects upon which a formal context is based are the rules which already constitute abstractions from the real world cases, so the objects (rules) we are dealing with are not as specialised as the cornerstone cases. Also the notion of a context is highly relevant because the applicability of the assertions being made is restricted to the context provided and are not meant to apply globally. However, for certain *activity-uses* the extensional definition is seen as problematic and too restrictive. As a result when the concept lattice is used to assist KA and validation of new concepts (Richards and Compton 1997a and Section 7.3.1 on critiquing) only the intensional definition of a concept is displayed.

Like the RDR lack of commitment to developing globally applicable rules:

> *"Extensional forms such as "for all" are avoided because it is not reasonable to assume that statements can be made about an indefinite extension whose members are not yet identified and may never be known" (Shaw and Gaines 1991b, p. 5).*

However, in a given context the author also agrees:

> *"if two distinctions have the same extension it may be regarded as evidence that they have the same intension" (Shaw and Gaines 1991b, p.15)*

Shaw and Gaines admit, as does the author, that such a hypothesis is flawed and suggest that the information contained in the extension is worth considering but with reservation. Intensional implication can be seen as a commitment to use terminology in a defined way. In a KR context this would mean a T-box description or in a KA context the focus is on whether the commitment is honoured in usage (Shaw, personal communication). Similarly, the use of cases requires that we approximate the subsume and disjoint relations based on the intensional properties of the cases, or objects. We can only approximate:

> *"because of the one way implication between intensional subsumption and extensional inclusion in an open world context, or, in psychological terms, that past behaviour is only an indicator, not a determiner, of future commitments" (Shaw and Gaines 1991b, p.14).*

3.3.1.4 Feedback and Analysis

Much KBS research has shown that it is important to understand the conceptual structures that underlie expertise and is explained by:

> *"If the person becomes more aware of the structure and organisation within the structure he becomes more able to make adequate predictions and act according to them" (Shaw 1980, p.7).*

Feedback is a key element in supporting KA so that the knowledge can be verified and validated "at every stage of development" (Gaines and Shaw 1993a, p.51). This is one of the strengths of computerised over manual repertory grid methodologies (Shaw and Gaines 1991b). However, it is not just the automation of the approach but the use of value-added tools such as FOCUS, PrinCom and Socio in KSS0 that provide "a deeper understanding and a reconstruction of a person's system" (Shaw 1980, p.149). The importance of feedback is demonstrated in the benefits of PEGASUS over MIN-PEGASUS which automated the repertory grid but didn't provide feedback and analysis tools (Shaw 1980). While a person can offer feedback there is a danger of interpersonal interactions that may bias or distort the results. The FOCUS work aims:

> *"to comply with the spirit of psychologists such as Rogers and Kelly [so that] one must aim to interpret the results as little as possible, leaving this to the subject" (Shaw 1980, p.33).*

By minimising the role of the KE and giving the user tools such as the focused grid allows the person to reflect on their own without the introduction of external bias. The three main analysis tools in KSS0 are Socio, PrinCom and FOCUS. The range of analysis tools available in the MCRDR/FCA implementation include: rule traces and browsing, the concept matrix (Figure 8:13), the concept lattice (Figure 8:14) and a k-distance weighted nearest neighbour algorithm. None of these correspond directly to PrinCom or FOCUS although the concept matrix and lattice provide hierarchical clustering as well as a visual representation of the relationships between objects and attributes. We take a brief look at each of the analysis tools in KSS0, spending more time on FOCUS. The nearest neighbour algorithm is described in Section 3.3.2.1.

Socio is used for comparison of conceptual structures from multiple sources and a good account is given in Shaw (1988). The work on requirements engineering (Richards and Menzies 1997 and 1998), which is reported in Chapter Six, is concerned with this type of comparison and focuses particularly on conflict detection and resolution. A further tool being developed by Biederman (1997) is the use of a triadic representation that allows multiple crosstables to be shown concurrently and it is hoped that this representation will be beneficial to the requirements engineering research described in Chapter Six. See Appendix B for an example.

PrinCom uses principal components analysis (Slater 1977) to provide spatial clustering. It is one of the most popularly used methods of grid analysis using techniques such as the Euclidean distances divided by the 'expected' distances to find inter-element distances. PrinCom produces a map which gives a visual representation of the relationships between objects and attributes. Shaw argues that such approaches which extract factors or components can distort the results because they can encourage bias in naming, data collection and experimentation (Shaw 1980). Nevertheless, PrinCom is a valuable inclusion in KSS0 because of its widespread usage with repertory grids. The nearest neighbour algorithm in Section 3.3.2.1 gives a score between 0 to 1 of the degree of closeness of one rule/concept to another but unlike PrinCom does not provide a spatial drawing of each concept in relation to each other.

FOCUS provides hierarchical clustering. The focusing algorithm developed by Shaw (1980) is a simple technique that does not impose as many restrictions as principal components analysis and interference with the subject’s data is minimal. Following Shaw (1980, p. 33-34), the construct matching score is scaled to provide a percentage matching score. The distance between two constructs i and j (3.1) is given as:

$$d_{ij} \rightarrow \frac{-200d_{ij}}{(n - 1)\, e} + 100 \tag{3.1}$$

where d_{ij} is the sum of the difference for each element (the city block metric), n is maximum value of the rating scale and e is the number of elements. This produces results between -100 and 100, where -100 is a perfect cross match, 0 is no match and 100 is a perfect match. Similarly the element matching score (3.2) is computed as:

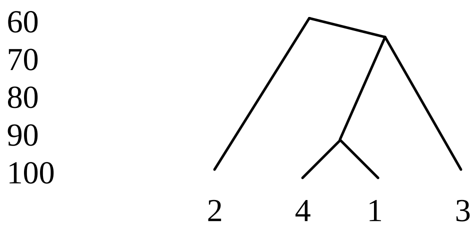
$$d_{ij} \rightarrow \frac{-100d_{ij}}{(n - 1)\, c} + 100 \tag{3.2}$$

where c is the number of constructs. The distance is multiplied in this case by -100 instead of -200 because elements are not bipolar and all results should be positive. As shown in Figures 3.9(a) and (b), the matching scores are placed in a symmetrical matrix and used to determine which construct/element should be linked together and the ordering of the constructs/elements within the tree. This new ordering is used to reorder the grid with one tree for the elements and another for the constructs which are attached to the corresponding nodes.

These measures, and those used by PrinCom and Socio, are useful and appropriate for the repertory grid input where the user assigns a number that indicates where along the construct an element should be placed. While attributes-values in cases are often numeric there is a very large space of possible values and some type of categorisation is necessary. We could use a fuzzy approach such as has been considered in Martinez-Bejar, Shiraz and Compton (1998) or use a nearest neighbour algorithm to compare the rule strings. In this case a better concept than distance is proximity, P, because the higher the score the closer the concept.

	1	2	3	4
1		28	69	87
2	28		54	61
3	69	54		35
4	87	61	35	

3.9(a)



3.9(b)

Figure 3.9. Building Hierarchical Trees in Focus.
The symmetrical matrix of the construct/element matching scores in (a) are used to build the hierarchical tree of the relationships between the constructs/elements in (b). The highest score for each row is plotted between the two elements with this score. Lines are then drawn from the elements to the point ensuring that no lines cross.

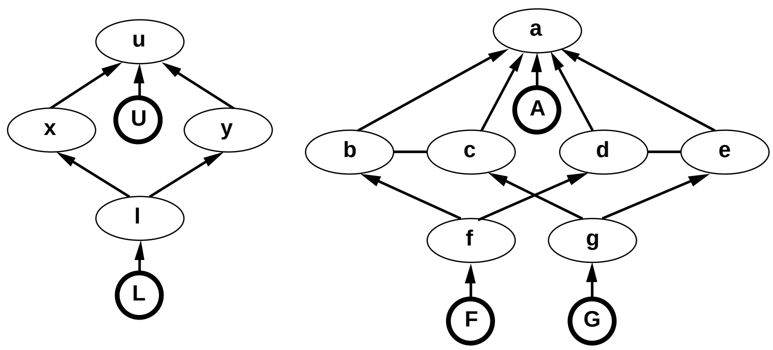


Figure 3.10 Calculation of distance measures between concepts and between constructs.
(From Gaines and Shaw 1993a)

$$\text{“}\mathbf{x}\text{ distance }\mathbf{y}\text{”} \qquad d(\mathbf{x}, \mathbf{y}) = C\mathbf{U} - C\mathbf{L} \tag{3.3}$$

Gaines and Shaw (1993a) also describe a distance metric between two concepts in the visual language in an extensional context, as shown in Figure 3.10. The distance measure simply considers the number of individuals in the extension of the minimal upper bound (CU) of the two concepts minus the number of individuals in the extension of the maximal lower bound (CL) of the two concepts (formula 3.3), where U and L are the extensions of u and l, respectively and C refers to the cardinal value of L or U. This can be extended to dichotomous constructs by subtracting the number of individuals in the maximal lower bound of one pole and then the other. When the scales are numbered linearly it equates to the city block distance measure used in FOCUS. The distance between the intensions of two concepts is given in the relational structure of the visual structure. The distance measures are used to cluster concepts for an individual and to compare conceptual models between experts. As part of the requirements engineering research, there is a need to compute the distance between two or more viewpoints and this is discussed futher in Section 6.4.6.

3.3.1.5 Validation and Maintenance

Gaines and Shaw say that as KA tools mature they should be able to:

> *“extend in scope from initial elicitation, through detailed knowledge modelling, to validation and knowledge base maintenance.” (Gaines and Shaw 1993a, p.76).*

In the RDR approach, maintenance and validation are inextricably entwined with KA and there is no clear distinction between the three. The exception structure, the use of cornerstone cases and difference lists ensures that previously correctly classified cases are not affected by the addition of new exception rules. Maintenance (also KA and validation) is designed to be performed on-line while the system is in routine use. In PCP and FCA validation and maintenance of the repertory grid or crosstable are supported as evidenced by the various analysis tools in KSS0. However, when it comes to deriving rules RDR supports incremental acquisition of rules which is not supported by PCP or FCA. These latter approaches require the set of implications to be regenerated. This is not a problem if regeneration is quick and automatic as in the case of Induct in KSS0. If generation of implications requires input from the user (in the case of FCA the user must supply counterexamples) then this is not only time-consuming but difficult, frustrating and error-prone. Another aspect of validation that differentiates PCP and RDR is that in RDR validation is performed by the expert whereas in PCP validation is primarily a task performed by the KE (Gaines and Shaw 1993b).

In addition to the automatic validation supported by the use of cornerstone cases in RDR, the concepts derived from FCA and the nearest neighbour algorithm previously mentioned support the validation of new knowledge (Richards and Compton 1997a). One of the strengths of starting with a performance system is that we are able to validate the knowledge being acquired by checking that the conclusion reached by the system matches either the conclusion reached by the user or the conclusion in the case, depending on the nature of the cases available. The calculation of the measures described in Shaw (1988) and the distance measures as described above are also ways of assessing the validity in terms of consistency between and within subjects. However, it is felt that starting with a conceptual model or T-box makes validation more difficult and to some degree inappropriate since we are dealing with descriptive models that are by their very nature imperfect representations. The subjectiveness of validating a model is exemplified in the following statement:

> *"The validity of the analysis [provided by FOCUS] is only measured in terms of the subjective feeling of personal significance assessed by the occurrence or otherwise of what has been called the 'aha' experience" (Shaw 1980, p.33).*

The lack of distinction between KA, validation and maintenance in RDR leads us to consider the differences between the RDR KA process shown in Figure 7.3 and the system development life cycle of systems based on PCP.

3.3.1.6 A Comparison of the RDR and PCP KA Processes

The stages in the PCP system development life cycle are summarised in Figure 3.11 (taken from Gaines and Shaw 1993a, p.81). In RDR the stages are not so clearly defined. It seems reasonable that any system will commence with some stage similar to Stage 0. This is also the case for RDR KBS where there would be a number of meetings to determine what the system was to achieve and the domain to be covered. Depending on the domain a date model may be necessary. The process, however, would not be as structured as what is probably performed in the interviews, protocols and media processes outlined in Stage 0 of the PCP development life cycle.

Stage 0 -	Acquisition of informal knowledge via interviews, protocols and media.
Stage 1 -	Use output from stage 0 to elicit major coherent sub-domains.
Stage 2 -	Use repertory grid technique to elicit relevant attributes and critical cases for subdomains.
Stage 3 -	Grids from stage 2 are used to induce concepts, their structure and rule.
Stage 4 -	The subdomains are linked together.
Stage 5 -	Test overall knowledge base and refine it by going back to the appropriate stage.

Figure 3.11: A General PCP System Development Life Cycle

Stages 1 and 4 are not applicable to the RDR methodology. In PEIRS knowledge was captured which covered a number of subdomains such as blood-gases, thyroid, liver functions, endocrines and catecholamines. The expert decided which subdomain to add first and others were added later. Particularly with the use of MCRDR there is no reason why all subdomains can not be added at the same time. The case-driven nature of RDR will to a large extent determine what subdomains are populated first. The decision to add subdomains concurrently or sequentially is not a restriction of the system and there is no need for integration between subdomains as is performed in Stage 4. The PCP and RDR approaches to handling subdomains differs because it is important to the repertory grid technique that the context be restricted and fixed so that the purpose and domain do not change during elicitation (Gaines and Shaw 1993a, p.61) which would distort the conceptual model being built. In RDR new attributes and conclusions can be added at any point and while the case provides the context, cases from different subdomains can be shown in any order. An MCRDR which covers a number of subdomains is equivalent to a set of individual trees for each subdomain, but the KA task and repetition is reduced by dealing with a number of subdomains concurrently. It is also envisaged that valuable relationships between subdomains will be more apparent when they are in one system.

As described in the previous subsection, the absence of stage 5 in which testing occurs is in keeping with the incremental nature of maintenance and validation provided by the RDR KA technique. Even the use of new reflective modes of KA, such as critiquing (Richards and Compton 1997b and Section 7.3.1), are designed to be performed on-line as part of the KA process each time a rule is added.

This leaves us with Stages 2 and 3 which highlight the core differences between the PCP and RDR paradigms. As previously described, RDR acquires an A-box and derives a T-box. In contrast, Stage 2 acquires a conceptual model that can be viewed as a T-box using the visual language and Stage 3 derives an A-box. A further difference is the use of repertory grids in Stage 2 to elicit the critical cases. The discussion on the role of cases in RDR compared to PCP was given earlier but it is worth offering our reasons for being less interested in stereotypical cases as it sums up our reasons for the absence of a system development life cycle. In the RDR approach the development of KBS is seen as an iterative and evolving process that requires incremental acquisition and that an over emphasis on acquiring “good” cases is similar to the over reliance on acquiring “good” models that we see in much other KBS research. Similarly, there is no attempt to define the relevant attributes for a domain but the input cases define what attributes exist and the rules the expert develops define which attributes are the significant ones for a particular case. As with PCP this avoids external interference in the KA process.

The PCP approach has been strengthened through the incorporation and use of other computer technologies such as hypermedia, the web, group decision support systems and collaborative learning. As reported in this thesis, modelling tools have been added to MCRDR based on FCA. These tools compare conceptual models and support requirements engineering. RDR research is also moving in this direction with a web-based on-line help desk (Kang et al 1997) and a web-based version of RDR (Thong 1997).

The combination of PCP and RDR is seen as beneficial. It is conjectured that the use of PCP as a means of eliciting cases would be useful for RDR in domains where cases do not exist or are difficult to develop. The use of triadic elicitation for KA in RDR could also be considered. Currently, in MCRDR all cornerstone cases that can reach a particular conclusion must be differentiated and are offered to the user when a rule is being defined. What the MCRDR system is doing is selecting all cases that are similar and asking the user to differentiate. In triadic elicitation the user specifies which two cases (elements) are the same and a third case which is different. Perhaps it would be beneficial to allow the user some input into which cases should be differentiated. However, in RDR systems there tend to be many cases being dealt with and there is also the problem of how the new knowledge would be incorporated into the KBS. In KSSn there is already the option to output rules as RDR-exceptions although the storage of cornerstone cases and an incremental maintenance strategy are not facilitated (Gaines 1991a).

From the discussion presented in this section it is apparent that the focus on systems used directly by the end-user, the use of simple techniques based on cases to acquire knowledge and the ability to support inferencing and explanation is shared by both PCP and RDR. The major difference is that RDR begins with a performance system and derives the explanation system and PCP starts with a conceptual model and later derives implications. From the point of view of validation and maintenance we see that the RDR approach is stronger but from the point of view of conceptual modelling the tools offered by PCP are more developed and extensive. The approaches are complementary rather than competing.

It can be seen that while RDR and PCP have much in common both have distinguishing features. Case-based reasoning is considered next and again it is shown that while RDR is definitely a form of CBR it has features that distinguish it from other CBR approaches.

3.3.2 Case Based Reasoning

The use of cases in RDR is fundamental. The founders of RDR saw the benefits of using cases, some of which have been mentioned above and are described further below, and also the benefits of rule-based system. However, both approaches had weaknesses. Rule based systems did not provide context and grounding in the real world. CBR suffers from the indexing problem associated with retrieving appropriate cases and deciding how best to adapt cases. This has resulted in RDR being a hybrid approach which has drawn on the strengths of both and attempted to avoid the limitations of both. Let us first get an overview of what CBR is, with some comment along the way regarding the RDR approach. This section concludes with a subsection on rule-based reasoning which reports the development of a well-known CBR technique, a k-distance-weighted nearest neighbour algorithm. Unlike the typical use of such algorithms to determine which cases should be selected, the algorithm has been designed to determine the closeness of one concept (rule condition or conclusion) to others for the purpose of discovering the relationships between concepts which was found to be important in supporting *activity-reuse.* We begin with an overview of CBR.

It is argued that the concept of CBR is used by most of us everyday in solving problems (Kolodner 1993). Lawyers and arbitrators are taught how to use cases as part of their jobs. Medical workers and technicians typically draw on past experience with prior cases to either recommend a current course of action or as a starting point from which to develop a new solution. The use of past experience seems to be a natural way that humans deal with problems. That is why experience is one of the traits that separates the expert from the novice. Using past cases has the benefit of saving time that would be required to derive a totally new solution plus the benefit that the soundness of the solution to the problem has been tested. CBR is not the same as reusing a problem solving method. The current case is actually checked against other cases and an index of differences and similarities is calculated. Depending on the indexing and retrieval system used the most appropriate case is selected. The differences between the retrieved case and the current case may warrant a modified recommendation to be made. Generally the user can ask to see the most similar case/s and be shown what differentiates the two cases. CBR, and RDR, thus also suffer from a similar problem to the generic problem solving method approach. If the case retrieved is not suitable or the adaptation is inappropriate then a poor or incorrect recommendation will be made.

The use of cases in RDR is similar to the Case Based Reasoning (CBR) approach. As described in sections 3.1.1 and 3.1.2 cases are used to:

- assist the user to develop rules,

- provide the appropriate context of the new rule being entered through storage of cornerstone case and
- validate the entered rule by ensuring the current case is differentiated from other cases associated with the incorrect rule.

When the expert picks the salient features in the case they are in effect selecting the index by which to retrieve the case. The creation of indexes is one of the major problems in CBR because most approaches try to generate these automatically to some extent. Getting the expert to perform this task substantially simplifies the task. However, while RDR does place importance on the role of cases, each case provides a local context and there is no attempt to define generally applicable rules or landmark cases. CBR techniques usually try to identify important cases that are used as significant cases which can be used to classify new cases.

Kolodner (1993) argues that in domains where models are not well understood, that is causal knowledge is not known, cases can be used to capture knowledge about the domain. Kolodner defines a case as:

> *"a contextualised piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goal of the reasoner" (Kolodner 1993, p.13).*

From this definition, the two key aspects of a case are its ability to teach one or more lessons and specification of the context in which the lesson/s apply. In CBR, the capturing of context is done by the assignment of indices to cases. As with RDR, the "purpose of indexing is to differentiate a case" (Kolodner 1993, p.266).

Kolodner (1993) describes three main different aproaches to indexing: difference-based, checklist-based and explanation-based. Difference-based indexing concentrates on the differences between the current and stored cases, while checklist-based indexing looks at which features tend to be predictive of solutions and outcomes. These two techniques are simple to implement but may retrieve an excessive number of cases due to overgeneralisation of cases and are not able to determine which features caused a failure. RDR uses a form of difference-based indexing because the index (the rule conditions) are based on the differences between the current case and the cornerstone case. However, the main difference is that the choice of index is controlled by the expert so the problem of retrieval of too many cases does not occur.

Explanation-based indexing trys to determine why a feature caused a failure or helped to solve the problem. SWALE (Schank and Leake 1989) uses explanation patterns (XPs) which are knowledge structures generated from and used like cases but are more general. XPs are used to enable the SWALE system to input stories from newspapers and reason about them. An XP includes:

1. a representation of the anomally it covers,
2. set of valid world states
3. set of useful states
4. a pattern of beliefs and relationships
5. set of prior episodes that have been explained.

The basic algorithm involves: anomaly detection, XP search, XP accepting, XP tweaking and finally XP integration. Reuse is seen as a act of creativity, using knowledge in a different situation or different way. There is a spectrum of creativity.

The rules and cornerstone case in an RDR KBS equate to points 1 and 5, respectively. The exception structure provides the set of valid world states, point 2. Points 3 and 4 are the more difficult issues and take us back once again to the issue of relationships between concepts. It is obvious that understanding how close one concept is to another is a key in knowing when a rule applies and when or how it should be adapted. Much of the knowledge that is input to a case-based reasoner involves manipulation of the raw case into a more general and more acceptably formatted case. Reformatting of the case is similar to the use of a preprocessor in RDR but generalisation of the case is not performed. In explanation-based indexed systems much of this reorganisation must be performed manually, whereas implementation of the preprocessor tends to require initial analysis of the raw data but once set up the transformation is performed automatically by the preprocessor.

SWALE uses its knowledge to find anomalies that it then tries to resolve based on cases. MOPS (memory organisation packets) are used in a similar way to scripts to allow memory to be dynamically used. There is more consideration of individual cases and they are able to determine the cause of failure. However, overspecificity may result in missing relevant cases. The key to developing successful indices is to abstract to the point where the explanation is still valid. This will provide the most generally applicable indices that retrieve a managable number of relevant cases. Unfortunately, explanation-based indexing is significantly more complex to develop and has only had success for narrowly defined problems (Kolodner 1993).

The way that a Ripple-Down rule is used to differentiate between cases combines the approaches of difference and checklist-based indexing and the failure-driven nature of learning in RDR makes it similar to explanation-based indexing. The use of explanation patterns, such as mentioned above in SWALE, is similar to a not-so-successful technique tried by Kang and Edwards (Kang 1995, pp:120-126) in the system known as

WISE, which was described briefly in Section 3.1.3. By determining and comparing all rule pathways that can lead to a particular conclusion it should be possible to perform blame assignment, which is an element of failure-driven learning which RDR has not currently addressed.

Kolodner (1993) sees the combination of all three types of indexing as necessary. There is no use just being able to determine if a case is close to another and then being unable to determine what differences exist. Additionally, the inclusion of a causal model will help us to determine not only why something failed but also why it succeeded and will facilitate deeper reasoning. Ideally a causal model could be built which supports explanation-based reasoning from cases. However, it has been found in PROTOS (Bareiss 1989) and MEDIATOR (Kolodner and Simpson 1989) that although the system assists by asking the appropriate questions it is necessary for a human expert to enter this information. The work on Teiresias (Davis 1977) also came to this conclusion and is further supported by the work on causal modelling and RDR by Lee (1996).

Indexing is only part of the story. Matching and ranking of cases must also be performed. The purpose of matching is to determine how close the new case is to those in the case library. This requires search procedures. Kolodner defines the following search methods:

- Flat memory, serial search
- Shared feature networks, breadth-first graph search
- Prioritised discrimination networks, depth-first graph search
- Redundant discrimination networks, breadth-first graph search
- Flat memory, parallel search
- Hierarchical memory, parallel search.

A detailed description with a summary of the advantages and disadvantages of each is provided in Kolodner (1993, pp;291-318). The last two search methods offer reduced processing time to read through cases which is achieved through the use of parallel processors. Basically the approaches above either read through every case (the flat memory approach) or they structure the knowledge so that only certain cases are examined and in a particular order. The benefits of structuring the cases, whether hierarchically or otherwise, is that it can reduce retrieval time and produce a smaller set of cases that are more relevant. The risk is that the structure may preclude consideration of cases that are relevant but appear in a section that is not being searched. Hierachically structured cases can also provide a visual organisation of the knowledge that is cognitively more sensible (Kolodner, 1993, p. 317).

Regardless of the search method, the cases read need to be compared against the current case using the following criteria (Kolodner 1993):

1. Are the goals the same ?
2. Do the constraints match ?
3. Are the descriptive features the same ?

If the goals, constraints or features are not the same it may be difficult to determine just how close they are. The third point is least likely to match exactly. We need to compute the degree of match along each dimension as well as the importance of each dimension. Qualitative values may need conversion into quantitative values to determine the degree of match. Using an example such as hot is closer to warm than to cold, we could convert hot, warm and cold into the values 1, 2 and 3, respectively. We can then consider the distance between the two values. If we are trying to compare across domains we may need to use abstract descriptions. Abstraction hierarchies can be used to compute the degree of similarity, such as is done in Section 6.4.6. The most specific common abstraction (MSCA) is determined by weighting the node. If the hierachy is well-balanced the distance between nodes can also be used.

The calculation of distance is an important one. It is possible to make use of qualitative ranges to determine such things as the closeness of age 28 to age 35. The problem that can occur is those values that fall near the boundary regions of these ranges can distort the results. If one category is ages 25-35 and another 36-40 it would appear that 35 is closer to 25 than it is to 36. Another technique is to consider the amount of distance. 35 is only a distance of one from 36 and a distance of 10 from 25. Another possibility is to look at the length of the chain of inference. Kolodner points out that it is important to score each dimension consistently and adjust later according to the importance of that dimension. She says that the assignment of importance values to dimensions is very difficult for experts. This was also the findings of Richards (1994) and is to be expected if one takes a situated view of expert action. In RDR KBS the categorisation of discrete values into qualitative ranges is the concern of the RDR preprocessor which provides refinement with respect to raw data. In the Garvan data borderline categories were defined to handle borderline values. A similar approach is described in Section 5.2.1.7.

The results of a search need to be matched or ranked. “When matching criteria become more local (associated with one or a small number of cases or conditions) matching becomes more dynamic” (Kolodner 1993, p.329). Some matching algorithms give a score which indicates the degree of match and some use comparison structures where each dimension is given a score. Match scores can be absolute or relative. An absolute score is more concerned with matching. A relative score provides ranking.

In summary, matching and ranking involves: finding correspondences, computation of the degree of similarity between corresponding features and assigning the importance to dimensions. The three main methods for selecting cases is to use a numeric nearest-neighbour algorithm, a heuristic method using evidence rules or preference heuristics or a combination of both.

The approach to KA used by Case-Based Reasoning also considers the importance of differences between cases as a means of determining the appropriate conclusion for a given case. The assumption is that if the features in two cases are similar then the conclusion should be similar (Kriegsman 1993). After determining which cases are most similar, adaptation is a key element of CBR where old cases are used to explain new cases and critique new solutions.

Determination of the important features in a case is a difficult issue. To try to find these features using machine learning on cases tends to require a large set of well-classified cases which are not readily available in many domains. Schank and Leake (1989) argue "Inductive learners have no criterion for importance of features except for their frequency of occurrence, they often make faulty generalisations when they deal with limited sets of data" (p.356). A problem faced by all learners is how to preselect the important features, especially since these may change with the domain or the context. The example Schank and Leake (1989) give is the importance of *time* when considering a car accident. In most cases the time is immaterial, however if the accident occurs around the time the pub closes the importance of determining the blood alcohol level of the driver increases.

Kriegsman (1993) describes the addition of a symbol hierarchy to assist in the organisation and retrieval of cases and their important features in a help desk application. The hierarchy is searched to find cases that solve a similar fault. A nearest neighbour algorithm is applied to rank the similarity of the cases. If no sort of indexing is used the look up time to find a good match will be O(n-k) where n = number of cases and k = the number of cases retrieved. If indexing is used, k good matches can be found in O(logn - logk) time. Kriegsman argues that use of a nearest neighbour algorithm is not sufficient on its own. While a nearest neighbour algorithm will be useful for finding similarities within a class, it is slow and faulty when used to determine similarities between classes. The symbol hierarchy allows the class to be approximated. The help desk application used induction on a log of 250 problems to generate a decision tree with about 80 decision nodes. This tree is used to select cases. The number of cases to select can be specified. Once selected the following algorithm (3.4) was applied:

$$\text{Total score} = \frac{\sum \{\text{Similarity}(\text{In}[i]\ ,\ \text{Ret}[i]) * \text{Weight}[i]\}}{\text{Sum of weights}} \qquad (3.4)$$

where, *i* is a feature used for matching, In[i] is the input case value, Ret[i] is the retrieved-case value. Weights can be applied where an attribute has been identified as important or otherwise to adjust the similarity index computed.

It may be worth considering whether RDR could benefit from the creation of a more abstract cornerstone case, rather than simply storing the case as it is presented. The context profile offered in *Prudence* (Edwards 1996) can be seen as an abstract case. The problem that can occur with abstraction is that while the reason why a fault occurred can roughly be identified it is difficult to specify the actual cause of the failure. It seems that it is necessary to identify why a particular case actually failed as well as learning how this knowledge can be applied to other situations that are similar but not identical. Alternatively, Gaines (personal communication) has suggested that all cases be kept. The power of CBR over the use of general heuristic rules is that specificity is retained, which Kolodner (1993, p.14) argues is recognised by the AI and Cognitive Science communities as being better and constitutes *strong knowledge* as opposed to general rules which are seen as *weak knowledge.* The RDR KBS approach combines both *weak* and *strong* knowledge through the use of rules and cases, respectively. The findings in Chapter Four of this thesis suggest that these two form a powerful combination because the knowledge provided by the rules and cornerstone case could not easily be reorganised without loss of accuracy.

3.3.2.1 Rule based reasoning

In RDR a rule is a case where the salient features have been identified to allow matching. The rule is not the same as a rule in a conventional rule-based system but is more like Clancey's SSM described in Section 2.1.1. The approaches by Kolodner (1993) and Kriegsman (1993) mentioned above make use of k-nearest neighbour algorithms for selecting cases and appeared to be suitable for selecting rules in an RDR KBS which were close to the concept under consideration. This was seen as potentially useful in assisting such *activity-reuses* as critiquing user-input recommendations, proposing new rules and conclusions, and for improving understanding of the concepts in the KBS and the relationships between them. Rather than CBR a more appropriate term is rule based reasoning.

The first step in comparing rules was to change them to a flat structure. Due to the problems that were encountered in the removal-of-repetition studies in Chapter Four involving the appropriate handling of the ambiguous false branches in single-classification RDR it was decided to use MCRDR which only has true exceptions. The rules were flattened by taking one end node at a time and working back through parent nodes, adding the parent’s conditions to the flattened rule. This resulted in some repetition of conditions, which were filtered out so that only one of each condition was present in a rule pathway. It appears that the expert’s focus is on the current case and the cornerstone case rather than the history of cases or consideration of the rules that have fired and their conditions. Due to this focus on the current context, repetition of conditions is found within a pathway.

To determine closeness, a flat memory, serial search against each rule is performed. This meant that the file of flat MCRDR rules was sequentially traversed and a comparison made between the rule in question and all other rules one by one. Each condition in the premise of one rule was compared against each of the conditions in the premise of the other rule. A straight comparison between rule premises was not made but each premise condition was broken down into three parts, the attribute, operator and value. A matching score was based on the distance between these parts. For example, if the condition to be matched was RAIN = High then a complete match would be given a score of 3. The same attribute and operator with a neighbouring value such as RAIN = Normal would gain a score of 2 and a match on attribute but not a bordering value such as RAIN = Low gains a score of one. This score was then divided by 3 and added to the total score. The reasoning behind giving a medium score to neighbouring values is to overcome differences in perception that occur between users and within a user at different times (Gaines and Shaw 1989). Giving a score of 1/3 to matches on attribute alone is done to show that there is a greater relationship between conditions that use the same attribute even if the values are opposite than there is with another condition that doesn’t use that attribute at all. It may be that the relationship is a negative one and it may be preferrable that the score provided be shown as positive or negative as is done with the construct matching score used in FOCUS. This has not currently been investigated.

Some work was first done on a weather knowledge base where it was suitable to break each condition into three parts. However, as a further test of this technique another MCRDR KBS known as 105 was chosen which contained rules for the blood gases domain. The 105 KBS had been developed from the 2000+ PEIRS cases and it is used again in the discussion and evaluation of the FCA modelling tools in Chapters Five and Eight. In Figure 3.12 the flattened MCRDR file has been sorted into ascending order on the conclusion code to see if there was anything obvious from manual perusal of the rules. We can see that rules 2 and 60 are identical. This situation has occurred because the expert has first seen a case that satisfied the conditions in rule 2 where %AB001 is correct. Two later cases were seen which resulted in two separate refinements to rule 2. Resulting in the rules:

```
35 2 0 22 %NC000 : (MAX(BLOOD_PH) >= 7.45)
38 2 0 35 %NC000 : (MAX(BLOOD_PCO2) >= 46)
```

Both refinements are stopping rules that apply when either of the rule's conditions are met. Finally, another case is seen which reached the null conclusion but which should have the original conclusion "%AB001- No disturbance of acid base status". This required a new rule, rule 60, to be added at the top level. Although there is repetition, we can see that RDR successfully performs verification at KA time since there are no inconsistencies in the rules which would result if there were rules with the same sets of conditions and different conclusions. We can also see the benefit of using a coding method for assigning classification codes. It is apparent that the codes entered by the user capture a meta-class of conclusions. This is important because it provides us with a way of evaluating how well the indexing algorithm is working because we would expect conclusions that share a conclusion prefix would be more related than those that do not.

```
61
2 %AB001 :(NORMAL(BLOOD_PH) = TRUE) ; (NORMAL(BLOOD_PCO2) = TRUE);
60 %AB001 :(NORMAL(BLOOD_PH) = TRUE) ; (NORMAL(BLOOD_PCO2) = TRUE);
5 %AB002 :(HIGH(BLOOD_PH) = TRUE) ; (NORMAL(BLOOD_PCO2) = TRUE);
58 %CG001 :(BLOOD_PREG = POS);
59 %CG001 :(URINE_PREG = POS);
29 %MC001 :(CURR(BLOOD_PH) <= 7.42);(NORMAL(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE) ; (LOW(BLOOD_PCO2) = TRUE);
11 %MC001 :(LOW(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE);
49 %MC002 :(CURR(BLOOD_PH) <= 7.36);(NORMAL(BLOOD_PH) = TRUE);(HIGH(BLOOD_PCO2) = TRUE) ; (NORMAL(BLOOD_PH) = TRUE);
14 %MC002 :(HIGH(BLOOD_PCO2) = TRUE) ; (NORMAL(BLOOD_PH) = TRUE);
10 %MC002 :(HIGH(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE);
19 %MC002 :(INCR(BLOOD_PH) = TRUE) ; (DECR(BLOOD_BIC) = TRUE);(NORMAL(BLOOD_PH) = TRUE);(HIGH(BLOOD_PCO2) = TRUE) ; (NORMAL(BLOOD_PH) = TRUE);
15 %MC002 :(LOW(BLOOD_PH) = TRUE) ; (HIGH(BLOOD_BIC) = TRUE);
9 %MC002 :(NORMAL(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE) ; (LOW(BLOOD_PCO2) = TRUE);
47 %MC003 :(INCR(BLOOD_PH) = TRUE) ; (INCR(BLOOD_BIC) = TRUE);(LOW(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE);
22 %MC003 :(MIN(BLOOD_PH) <= 7.34) ; (MIN(BLOOD_BIC) <= 23);(NORMAL(BLOOD_PH) = TRUE) ; (NORMAL(BLOOD_PCO2) = TRUE);
40 %MC003 :(MIN(BLOOD_PH) <= 7.34) ; (INCR(BLOOD_BIC) = TRUE);(CURR(BLOOD_PH) <= 7.42);(NORMAL(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE) ; (LOW(BLOOD_PCO2) = TRUE);
44 %MC003 :(MIN(BLOOD_PH) <= 7.34) ; (INCR(BLOOD_PH) = TRUE) ; (MIN(BLOOD_BIC) <= 23);
57 %MC004 :(DECR(BLOOD_BIC) = TRUE);(LOW(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE);
54 %MC004 :(DECR(BLOOD_PH) = TRUE) ; (DECR(BLOOD_BIC) = TRUE);(LOW(BLOOD_PH) = TRUE) ; (LOW(BLOOD_BIC) = TRUE);
55 %MC004 :(NORMAL(BLOOD_PCO2) = TRUE);(DECR(BLOOD_PH) = TRUE) ; (INCR(BLOOD_PCO2) = TRUE);(LOW(BLOOD_PH) = TRUE) ; (CURR(BLOOD_PCO2) >= 41);
16 %MK001 :(NORMAL(BLOOD_PH) = TRUE);(HIGH(BLOOD_PCO2) = TRUE) ; (NORMAL(BLOOD_PH) = TRUE);
25 %MK002 :(CURR(BLOOD_PCO2) >= 37) ; (CURR(BLOOD_BIC) >= 27);(HIGH(BLOOD_PH) = TRUE) ; (NORMAL(BLOOD_PCO2) = TRUE);
26 %MK003 :(DECR(BLOOD_PH) = TRUE) ; (DECR(BLOOD_BIC) = TRUE) ; (MAX(BLOOD_BIC) >= 27);(HIGH(BLOOD_PH) = TRUE) ; (NORMAL(BLOOD_PCO2) = TRUE);
50 %MK003 :(MIN(BLOOD_PCO2) >= 37) ; (MAX(BLOOD_BIC) >= 27);(DECR(BLOOD_PH) = TRUE) ; (MAX(BLOOD_PH) >= 7.45) ; (INCR(BLOOD_PCO2) = TRUE);
```

Figure 3.12 A part of the 105 KBS developed in MCRDR by Glenn Edwards has been flattened and sorted by conclusion code.

The rules in the blood-gases domain are more complex than in the weather domain. In this situation we have conditions like (HIGH(BLOOD_PH = TRUE) which includes four components - the function HIGH, the attribute BLOOD_PH, the operator = and the truth value TRUE. We also have to consider how to handle time course data and continuous data such as (CURR(BLOOD-PH <= 7.42). The way I treated such data was to break them down into these smaller components and compare the corresponding component in each condition. The attribute was compared first, if there was a match it received a score of 1 and then other components were matched. Any further matches gained a score of 1. This meant a total of 4 was possible which was then divided by 4. If there had been a complete match the two conditions gained a final matching score of 1. If it was a partial match the score was between 0 and 1.

While the use of continuous (numeric) and nominal (symbolic) attributes in the distance metric offered is practical for the example offered it may pose problems with other rule-sets as it is not well grounded. The handling of mixed attributes is an unresolved problem in the Machine Learning community (Motoda, personal communication).

It is interesting to note that using this matching method rules 58 (BLOOD_PREG=POS) and 59 (URINE_PREG=POS), which both conclude "%CG001 - Pregnant", will not be matched because they use different attributes. This same problem occurs in FCA since each rule condition is treated as an attribute and it not broken down into smaller parts. The relationship between these two rules is obvious to the human eye but more difficult for a machine to learn. We can see from the rules that getting a positive result to a pregnancy test, which can be determined by a blood or urine sample, is the critical factor in determining if a patient is in fact pregnant. This limitation can be overcome by increasing the number of parts to break the string into for comparison. For example if the attribute URINE_PREG is treated as two separate attribute strings it would result in a match of three (PREG,=,POS) out of the four components. The drawback is that it means a pointless match on the word BLOOD which is present in all conditions in the KBS besides the one in rule 59. This type of problem is evidence of the difficulty faced by CBR systems in matching cases which often require some sort of tweaking such as hand coding and identification of appropriate cases for a specific domain. An alternative is specification of a higher level concept which covers both types of pregnancy test.

Having determined a way of comparing two conditions, I was more interested in determining the closeness of concepts such as evaluating which conclusion was the closest to another specified conclusion or which rules are closest to a specified rule.

This closeness measure, P for proximity, uses a distance-weighted nearest neighbour algorithm (3.5) that considers the number of string matches compared to the number of conditions being compared.

$$P_{ij} = \frac{\sum M}{C_i} \; * \; \frac{\sum M}{C_j} \qquad (3.5)$$

Where i and j are the two rule premises being compared, M is the matching score as determined from above and C is the cardinal number of conditions. M is similar to the numerator in Kriegsman's formula 3.4 in that it considers the similarity between the input case and the retrieved case and a weight is used on the value of the feature but M is more complex in that features are only compared once a match on attribute is found.

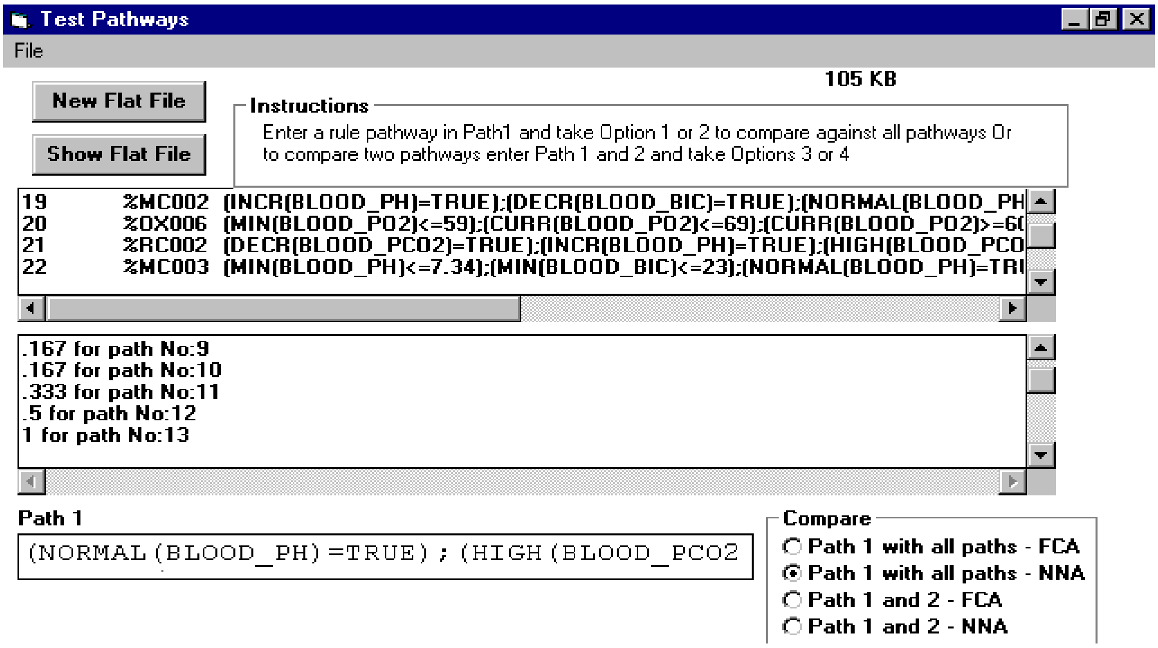


Figure 3.13 The Test Pathways (TESTP) screen in MCRDR/FCA.

Currently use of the algorithm is to support the user in exploring if and to what extent a relationship exists between rules and conclusions and in particular to assist them to determine whether a new proposed rule relates to the existing rules. The nature of the relationship is not given but can be found through comparison between the concepts derived from FCA. The TESTP (test pathways) screen, shown in Figure 3.13, has been developed in MCRDR/FCA to assist the user in evaluating a particular rule pathway either against all pathways or another selected pathway using the nearest neighbour algorithm. Also a comparison can be made between the proposed pathway and the intensional definition of the concepts derived using FCA. The equation in 3.5 has been applied only to rule premises but there is no reason why the rule conclusion could not be treated as another condition used to compute the similarity score. In the current use of this algorithm the user is shown the conclusion as part of the result so that they make their own judgement on whether the score seems appropriate. Since the algorithm relies on string matching, inclusion of the conclusion as a condition would only be benefiical where the conclusion codes either matched or were part of the same family of conclusions. Alternatively the free-text description associated with each conclusion code could be compared on a word by word basis with the string being condsidered. However, this approach may produce results that are less reliable than those based on rule conditions which use more strictly defined terms and structure.

In Figure 3.13, the rule pathway (NORMAL(BLOOD_PH)=TRUE) and (HIGH(BLOOD_PC02) = TRUE) has been entered into the field Path 1 so that related rules could be found. This screen can also be accessed from the MAKE RULES screen which would place the proposed rule conditions into this field. The user then clicks the radiobutton "Path1 with all paths-NNA" which uses the nearest neighbour algorithm described in this section to show a list of scores between 0 and 1 for each other pathway in the KBS as shown in the second scrollable listbox. Using this information the user can look at the pathway in the top listbox to reflect on whether the conditions and conclusion in the path is consistent with the hypothetical or proposed rule path and the associated conclusion.

The nearest neigbour algorithm presented here has been evaluated on a number of domains and KBS. The algorithm can be used to find the closesness of concepts. For example, using the CLIPS Animal KBS, discussed in Section 8.1.1.1, we could determine which animals are closest to man. To do this, the rule for the man concept ((backbone=yes);(warm.blooded=yes); (has.breasts=yes); (can.eat.meat=yes); (fly=no); (opposing.thumb=yes); (prehensile.tail=no); (nearly.hairless =yes)) was selected and the algorithm used to compare this rule against all other rules in the KBS. The following Proximity scores were given:

0.42	flatworm, worm, leech, centipede, millipede, insect, lobster, crab, jellyfish, protozoa, sea anenome, coral, sponge, clam, oyster, squid, octopus, leech,
0.167	fish, shark, ray, turtle, frog, salamander, crocodile, aligator, snake, snail
0.292	bird, penguin
0.417	rhino, horse, zebra, whale, dolphin, hippo, camel, girrafe, rabbit, rat, possum, kangaroo, koalo, mole, shrew, elephant, sheep, goat, deer, antelope, moose.
0.542	bat
0.667	bear, tiger, lion, walrus, cat, wolf, dog, coyote, fox.
0.792	monkey
0.917	gorrila, baboon, orangutan, chimpanzee.

From the results it appears that gorillas, baboons, orangutans and chimpanzees are the closest concept to man, followed by monkeys and so on. Many of these animals (such as the last four with 0.917) were grouped together. It may be that modification to the concept is considered necessary to move a concept closer or further away but generally the scores given above do appear intuitively reasonable at a coarse level. As can be seen in Figures 3.13 and 7.11, both the nearest neighbour algorithm and the concepts developed using FCA are used in the MCRDR/FCA system as they offer different features. The NNA provides a score which is concise and a good first step in seeing what concepts are alike. The FCA concepts show in what way the two concepts are alike. Specifically they describe the concept as a match, sub or superconcept of the concept in question.

3.3.3 Summary of the Relevant Theory

In this section we have looked at PCP and CBR and seen how RDR relates to both of these theories. A number of strengths of RDR over these approaches were noted, with the maintenance and validation strategy and the use of rules for indexing cases being the key benefits. As noted in the last subsection, it was important for *activity-reuse* that the relationships between concepts be found and that this was unsupported by RDR. We look further at the weaknesses of RDR and in particular consider those that affect the reuse of knowledge and which have been addressed in this thesis.

3.4 The Shortcomings of RDR

There are a number of criticisms of RDR that particularly affect the reusability of knowledge. The first one concerns the exception structure of the knowledge representation which results in the need to repeat rules that should be globally applied in a number of locations in the tree.

A second problem relates to the absence of higher-level models in an RDR KBS which restricts the range of activities for which the knowledge is useful. The simplicity of KA, maintenance and inferencing with RDR is seen as one of its strengths but there are situations where people want to reflect on the knowledge captured for such purposes as explanation, teaching and documentation. Similarly, RDR has been restricted to classification tasks but that is changing with the recent positive results with a configuration task. This thesis is particularly concerned with addressing the inability of RDR to handle a wide range of activities. The ability of RDR to handle a wide range of tasks is not the direct concern of this thesis.

A third limitation also faced by other KBS approaches relates to the growing interest in requirements engineering (RE) and the ability to acquire and merge knowledge from multiple sources of expertise that RE necessitates. This interest can be seen as an activity that requires different questions to be asked such as “how does the knowledge in KBS A acquired from Source 1 compare to the knowledge in KBS B from Source 2 and how can they be reconciled into a combined body of knowledge?” RDR research has previously not considered this problem and the development of an RDR approach to RE is seen as a further enhancement to the robustness and reusability of the RDR paradigm. RE is an issue yet to be faced by most KA approaches and the work in this thesis represents early work in this field.

A related limitation which impacts on the *activity-reusability* of RDR has been identified by Menzies (1997b) and concerns the representation of PSMs. Before a RDR tree executes, the feature extractor functions find all the features in a case. The delta logic is constructed via computing the difference between the case features and the features found in the rules along the path to the faulty rule. Consequently, RDR is focused on the details of the specific case at hand. PSMs may not be expressible with respect to the specific problem at hand. For example, consider an RDR tree maintaining taxi driver knowledge. Our student taxi driver may learn many tricks about navigating from Sydney to Canberra through specific streets. However, in an RDR framework, she may never learn the generalised PSM: open the map, find your current location, find your destination, compute the shortest distance path from here to there. More generally, to date, the functions called in RDR conditions do not access the meta-level reasoning used in problem solving. These representational problems are a particular drawback for RDR. Menzies (1997b) argues that to resolve these problems, we will need a library of functions that can access the internal data structures of the inference engine. Further, we will need to offer a maintenance strategy for those functions.

The higher level concepts that are derived using FCA in Chapter Five provide abstractions of the data but do not provide any of the meta-level reasoning that may be part of the problem solving process as described above by Menzies. The results reported by Compton et al (1998) for the ion chromatography configuration task question the importance of strong problem solving methods and minimal data versus weak problem solving methods, as offered by RDR, and lots of data. This thesis, and Menzies (1997a), also argues that PSM knowledge has yet to stabilise and sees PSMs as models of assumed behaviour. While one may argue that the general taxi driving PSM is acceptable this may be because it is understood well enough that it can be defined and perhaps also therefore does not need to be acquired or taught. The process involved in solving more complex tasks may not be as standard across the population. It has been the goal of RDR research to see whether the development of individual task specific PSM can be avoided and whether a general purpose inference engine can be built. The research in this thesis has also avoided the reuse of PSM issue to a large extent as it has focused on uncovering higher level concepts based in existing knowledge and thus it would be impossible to uncover higher level PSM knowledge if the lower level PSM knowledge has not been captured.

The three limitations concerning knowledge repetition, lack of modelling and handling multiple sources of expertise, however, are addressed in this thesis and are described further below.

3.4.1 The Repetition Problem

Due to local patching, there have been concerns that the RDR method would result in large numbers of repeated rules. Compton, Preston and Kang (1994 and 1995) show that manually acquired RDR produced KB as compact as those that used inductive techniques to acquire the knowledge. It appears that experts usually enter general rules (Mansuri et al 1991) so that repetition is not excessive. However, when it comes to reusing this knowledge for such things as explanation it seems expedient to clean up what repetition does exist. Since the RDR pathways have been shown to provide a more useful descripton (Compton et al 1991a, Richards and Compton 1996) than the conventional rule trace, cleaning up of redundant pathways should further enhance the comprehensibility of the explanations provided.

In addition to the repetition from local patching, another reason why repetition occurs is because experts often create new classifications rather than reusing already defined classifications. This is done for two reasons. Firstly, creating a new classification saves time searching through existing classifications. Secondly, due to the context aspect associated with each case, the expert often wants to word the conclusion slightly differently for individual cases. The conclusion in RDR is input by the user in a free text format. This has resulted in numerous conclusions that appear very similar. Obviously identification of the valid classes is important in any classification task and the use of synonyms and homonyms found when free text is used has resulted in repetition of the knowledge. Currently the user can search on a keyword and all conclusions with that keyword are displayed. This approach however does not appear to be satisfactory. There does not seem to be enough incentive for the expert to use this facility. The expert needs to be constrained to input more consistent conclusions and to keep the number of conclusions to a minimum. The approach used by Zacklad and Fontaine (1993) may be useful in finding a solution to the problem of repeated conclusions.

Zacklad and Fontaine (1993) cite much literature that accepts that classification is necessary and useful, but they argue that sometimes we end up with a combinatorial explosion or at least a large problem space when we try to determine all the combinations of all the variables and their values. They say there are two main structural principles of classification - similarity (based on the appearance of similar features which they call 'metaphoric') and genealogy (based on their inheritance 'metanymy' or function based, for example, hoover is equated to vacuum). There are various types of metanymies such as: causal, functional and instrumental. Experts often classify things in one way such as by similarities but they also need to classify in other ways (eg causal).

Zacklad and Fontaine (1993) see two distinct types of representations, feature-oriented and dimension-oriented. They argue most systems are concerned with dimensions which cover attributes such as size. C-KAT takes a different, feature-based approach. The technique they use has four steps:- elicit the attributes, design the dependency network, design the structure of classifications, instantiate the classifications and match them. They point out that missing values become a problem and that not all features apply to every attribute. The C-KAT system is not used until the fourth stage. In designing the classifications, composition is performed which uses the main attribute differences to incrementally add structure. Incompatability constraints are used to reduce the large numbers of combinations that appear. 'Is-a' feature trees and constraints between them allow the generation of a reasonable number of object classes.

Zacklad and Fontaine (1993) give a practical example of its use and compare their work to other classification work. They claim their work allows for creativity at all stages. KA is a development process and results in changes to the users perception, it must therefore be a creative and flexible process. A further claim is that C-KAT has a future as part of the workbench toolkit such as ETS and KSS0. The compatibility of the RDR approach with these systems based on personal construct psychology, discussed in Section 3.3.1, may mean that the C-KAT approach may be a useful addition to the RDR toolkit. Validation of this conjecture however is beyond the scope of this thesis. In Chapter Nine one solution is suggested for the problem of too many classifications which is the retrospective analysis of rule conclusions to see if they can be combined into a smaller set of conclusions by finding the closeness of conclusions using the nearest neighbour algorithm or the FCA concept hieararchy.

Other work concerning the reduction of the repetition problem has been done by (Chellen 1995). In that study a method was developed for decreasing the size of manually built RDR systems by a factor of 2 or more. The method of compaction was Vinduct. This algorithm was a close reimplementation of the Induct/RDR algorithm developed by Gaines (1989b). In Induct a search is made for the premise of a particular conclusion that is least likely to predict that conclusion by chance and assume that premise is the best measure of correctness of a rule. This algorithm was reimplemented so that it would run on a Unix platform as an experimental testbed.

In the approach to the removal of repetition, attribute-value pairs in cases that were not evaluated by rules were replaced by missing values. To test this approach the original data set from which the manual rules were developed was run with the cleaned-up set of rules. The amount of compaction after reorganisation was encouraging but the associated misclassification error rate was unacceptably high. This was thought to be due to two reasons. One was the way a default conclusion was assigned and it was assumed that an expert would be able to do this more intelligently. The second reason was the inclusion of the attribute-value from the case when a false path had been taken rather than the negation of that value. For example if the value of the attribute on the false path was 1 then the test should have been for not 1, not for the case value of 2 or 3. Possibly the main reason was the nature of the Garvan domain where the most common conclusions covered a number of spurious situations that should have been given more specific classifications. Induct first finds the most popular classification and develops rules to cover the classification. If the most common classifications tend to cover strange and possibly inconsistent examples it will be difficult to find good selectors. These issues are considered further in the Chapter Four which describes further work that was done in collaboration with Chellen. Her results were compared to the experiments I conducted using rough set theory to remove repetition.

The experiments described in Chapter Four use single classification RDR KBS. MCRDR have been shown to produce less repetition and the interpretation of pathways is less ambiguous because there are no false branches. It was unfortunate that MCRDR was still in its infancy when the repetition studies were being conducted and it is believed that the results would have been better if MCRDR KBS had been used. A proposal for removal of repetition from MCRDR KBS is given but the major problem that remains in testing this proposal is the fact that there are still few large MCRDR KBS in existence which would be suitable candidates for experimentation.

3.4.2 Lack of Modelling Features

While it was argued in Section 1.2 that most expert action is reflexive there is still the need at times to reflect on knowledge. So, although it was not necessary to develop abstractions for KA, inferencing or maintenance of the knowledge, it was worthwhile to find the higher-level symbolic models contained in our KBS due to their "explanatory value as psychological descriptions" Clancey 1993, p.89) and their usefulness in instruction (Schon 1987).

Most RDR research has concentrated on KA and maintenance. However, the work on reflective ES (Edwards et al 1995) and causal modeling (Lee and Compton 1995) have found RDR to be an extendable representation. Additionally, the rule pathway provided by the RDR exception structure has been shown (Compton et al 1991a, Richards and Compton 1996) to offer a better explanation of how the knowledge has evolved, why a rule has both succeeded and failed and what alternative pathways are possible than conventional rule traces. However, explanation often requires the presentation of a model of the concepts that a rule or collection of rules represents. To support critiquing it is necessary to determine whether the user's conclusion was the same or within acceptable limits of the system's conclusion. This required determining the closeness of rule pathways and some measure of the extent of similarity. The nearest neighbour algorithm described in Section 3.3.2.1 provided some of this information but not the structure of the knowledge or higher level concepts. Thus in trying to find higher-level models in the KB rules it seemed useful to investigate Formal Concept Analysis.

The acceptance of RDR to some extent has also been limited, particularly in other research projects, because the lack of explicit modelling produces insufficient documentation or formalisation that is often considered desirable output for a research project. It was explained to me by a colleague at the Australian AI'95 conference that they would have used RDR for a particular problem that they had but that they needed a methodology that showed evidence of analysis and a software-engineering approach. The retrospective analysis tools added to MCRDR and described in this thesis may provide some of the structure and higher level models that were needed.

3.4.3 Handling Multiple Sources of Expertise

It is becoming more widely recognised that part of the KA activity will involve capturing knowledge from a range of sources (which may include not only more than one human expert but sources such as books and work documents). The management of conflict between these sources has been postponed by many as no easy solution is seen to be readily available. The management of multiple conflicting viewpoints is a requirements engineering task. The modelling tool developed in Chapter Five using formal concept analysis seemed to provide a framework within which RE could be further explored. In Chapter Six the RDR KE approach is extended to support a RE task, using the SISYPHUS III knowledge sources.

3.5 Summary

In this chapter we have been introduced to RDR. We have considered the two main RDR structures, single classification and multiple classification RDR, and a wide range of implementations that use these structures. RDR has been compared to PCP and CBR as they share many common features. A nearest neighbour algorithm was described which calculates the closeness of concepts and supports rule-based reasoning. Together with the strengths of RDR, a number of limitations have been described that impact on the reusability of RDR for different activities. These limitations are addressed in the following three chapters. Chapter Four looks at the removal of repetition and some more issues concerning the verification and validation of RDR KBS. Chapter Five describes the incorporation of FCA tools into MCRDR for modelling purposes. Chapter Six looks at handling multiple sources of expertise and conflicting viewpoints. First, let us look at the repetition problem and the related experiments.

Chapter 4

4 The Removal of Redundancies and Off-line Verification of RDR KBS[20]

In the previous chapter the problem of repetition in RDR KBS was introduced. It was seen that redundant knowledge was not ideal when we seek to understand and explain the knowledge in a KBS and that removal of repetition should improve the ability of RDR to support the reflective activities. In the previous chapter there was also a description of how validation and verification (V&V) is handled in an RDR KBS. This chapter continues the discussion of repetition and V&V in more depth, describing and evaluating work that was aimed at providing a method for compacting RDR KBS and a strengthening of the RDR approaches to V&V. In Section 2.3 it was pointed out that this study was interested in two questions related to reuse. The first one being how to reuse what we already have and the other being the development of a technique that facilitates future reuse. This chapter is concerned with the first question.

Studies by Mansuri et al (1991) and Compton, Preston and Kang (1995) using real and simulated experts[21], respectively, have shown that the repetition problem is not as serious as expected and that the size of an expert built RDR KBS compares favourably in size with various inductively built KBS.

While experts appear to add very simple general rules that result in minimal repetition (Compton et al 1991a), it was the aim of this study to review the repetition and redundancy that does exist to improve the clarity of the knowledge base and its suitability for reuse. It has been claimed that the rule trace provided by an RDR KBS offers better explanation facility than conventional rule based systems (Compton et al 1991b). This is because the user can be shown the path the expert followed in creating the rule and can see not only the rules added to deal with various cases, but also the case which prompted a new rule to be added. The inference pathway reveals the history of cases from which the rules have evolved. It is expected that this explanation facility will be further improved if there are less pathways leading to the same conclusion and if superfluous conditions are removed. Further, it was the goal of this study to preserve the existing knowledge acquired from an expert while gaining the benefits of inductive

---

[20] Sections 4.1 to 4.4 follow closely after Richards, Chellen and Compton (1996). The work on Induct reported in this chapter has been primarily performed by Vijaletchmee Chellen as part of her honours thesis but is included here for completeness and clarity.

[21] The use of a simulated expert is described later under Section 4.2 Experimental Method.

learning which could include: detection and correction of inconsistencies; identification and coverage of gaps in the knowledge; removal of redundancies; and simplification of expert-derived decision rules. Section 4.7 considers the first two benefits. In Sections 4.1 to 4.5 the main focus is on the two latter points which were to be achieved by making rules more general, thereby avoiding the addition of later specific instances. It was also hoped that greater insight into the knowledge captured would be obtained so that the knowledge could be potentially used for other purposes such as teaching and modelling.

The next section gives an overview of the two machine learning (ML) algorithms which were employed. Section 4.2 describes the experimental method used in this study and Section 4.3 looks at the results of the experiments. Section 4.4 reviews the results and discusses the issues raised. Section 4.5 proposes how repetition could be removed from MCRDR KBS and Section 4.6 summarises the findings of the repetition studies. Section 4.7.1 looks at the use of rough set theory for the verification of conventional rule-based systems and 4.7.2 applies the verification techniques to MCRDR KBS.

4.1 Compacting a Manually-built RDR KB using Machine Learning[22]

Machine learning can be seen as one way to alleviate the KA *bottleneck* problem. Knowledge bases can be built much more rapidly using ML and usually have many fewer rules than those built manually. KA using manual RDR is not as quick as with a ML technique but it has been found to produce rule bases of similar size to those built using ML techniques such as ID3, C4.5 and Induct and matures more quickly, requiring a small number of cases to achieve high accuracy levels (Mansuri et al 1991). Manual RDR is a rapid method of KA, approximately 40 times faster than traditional KA techniques (Compton and Jansen 1988). The compactness of manual RDR and it's exception structure prompted Brian Gaines to modify his Induct algorithm to output RDRs. The modified algorithm is known as Induct/RDR (Gaines 1991a, Gaines and Compton 1995). Using the Garvan clinical pathology dataset of 9551 cases, with the first 8000 cases as the training set and testing on the remaining 1551 cases, Induct/RDR generated 195 rules with 2.31% errors (Gaines and Compton 1992) which is comparable

[22] Sections 4.1 to 4.4 follow closely the paper “Richards, D., Chellen, V. and Compton, C (1996) The Reuse of Ripple Down Rule Knowledge Bases: Using Machine Learning to Remove Repetition *Proceedings of Pacific Knowledge Acquisition Workshop PKAW’96*, October 23-25 1996, Coogee, Australia, 293-312.

to 550 rules with 2.8% errors and 502 rules with 2.71% errors on 9514 cases reported in Mansuri et al (1991) with human built systems.

The above results raise the question of whether manual RDR is needed. However, the successfulness of using ML to derive knowledge depends on the availability of sufficient well-classified cases. As noted above manual RDR outperforms ML techniques when only a small number of cases are available. It has also been shown that the expert enters new concepts that do not exist in the historical data (Compton et al 1991a) and the rules output by ML techniques, such as C4.5 are not as comprehensible to experts as the RDR exception structure (Catlett 1992)[23]. This is one reason why Gaines implemented Induct/RDR. One alternative proposed by Gaines and Compton (1992) is to use manual RDR to build an initial KBS and when the KB becomes large, use ML, Induct in particular, to minimise the size of the rule base. This study pursues Gaines' proposal.

The ML algorithms chosen for this study were Induct (Gaines 1989) and rough set theory (RST) (Pawlak 1991). The use of Induct/RDR was an obvious choice since it already produced RDR output. At the time of these studies RST was being considered as a way of finding the key concepts and relationships in an RDR KBS and the ability to strip out redundant attributes was attractive for the repetition studies. A key feature of RST was that it only used one of each case and did not rely on multiple instances in generating rules unlike methods based on statistics or information gain. This seemed important when the goal was to analyse rules not cases. Additionally RST was similar to RDR in its use of differences to assist classification. An overview of both Induct and RST follows.

4.1.1 An Overview of Induct

Induct uses induction to build a knowledge base in terms of classes, attributes, attribute values and rules (Gaines 1989b). Figure 4.1 below reveals the basis of the statistical tests underpinning Induct's rule generation. The aim is to find a measure, *r*, which determines how good is a selector S (attribute-value combination). That is, we want a selector which is least likely to predict the target class by chance.

The first step is determine "what is the probability that random selection of the same degree of generality would achieve the same accuracy or greater" (Gaines 1989b). To calculate this probability we let Q be the target class in our universe of entities, E, for

[23] Catlett (1992) has further suggested that because of RDR's comprehensibility that it be used as a mediating representation, where rules generated by ML are converted into RDR format for perusal by experts.

which Q(e) holds and S be the selected entities in E for which S(e) holds. We are interested in finding the probability of C(e) which is the intersection of Q(e) and S(e) and is given by:

$$C \equiv \{e: e \in E \wedge S(e) \wedge Q(e)\} \qquad (4.1)$$

The probability of getting an instance belonging to the set Q when an entity is selected at random from E is given by:

$$p = \frac{|\,Q\,|}{|\,E\,|} = \frac{q}{e} \qquad (4.2)$$

Thus, the probability of getting c (and possibly more) instances at random from Q when s items are selected from E is given by:

$$r = \sum_{i=c}^{s} {}^{s}C_{i}\, p^{i} (1-p)^{s-i} \qquad (4.3)$$

This is a measure of correctness of the rule produced and is an exact calculation of the probability of an event occurring unlike the entropy algorithm used in C4.5. The aim is to choose a selector S which minimises r, that is, a selector S for which the likelihood of having c entities at random from E is low. Assume the better the selector, the lower the value of r. “The advantage of selecting r as a measure of the correctness of a rule is that it is easily understood as the probability that the rule could be this good at random and that it involves no assumptions about the problem such as sampling distributions” (Gaines 1989b, p.99).

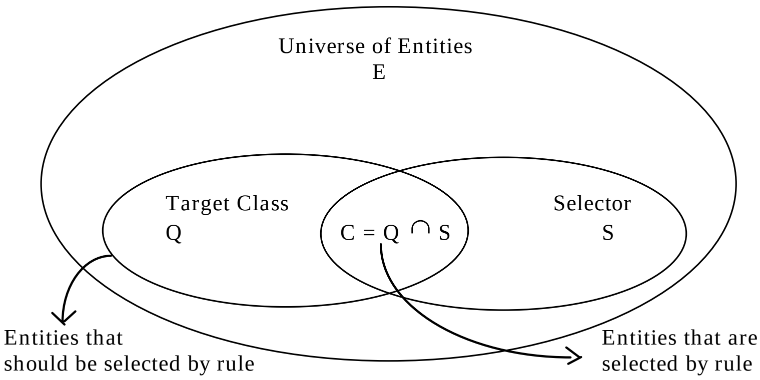


Figure 4.1 - Induct’s statistical basis (from Gaines 1989b)
where

E represents the set of instances in the dataset.
Q represents the set of instances of the target class.
S represents the set of instances satisfying a predicate P.
P consists of a set of attribute value pairs.
C represents the instances of the target satisfying the predicate P.

Induct was reimplemented on the Unix platform so that we could add our own code to support the various clean up algorithms used on the cases as described in Section 4.2.

Vinduct (Vija Induct) is our Unix implementation and the algorithm employed to generate rules is presented in Figure 4.2. We have not developed a new algorithm but we also do not claim that the algorithm is identical to Gaines' Induct algorithm. The Vinduct algorithm included various heuristics to determine situations where the selector value was the same and a one step lookahead to determine the impact of adding a condition to the selector value. From personal communication between Chellen and Gaines similar heuristics have been used and the results obtained by our implementation is satisfactorily close to the results of Induct.

```
        ReadCases (Dataset)
        Find Default Conclusion
        Process (Dataset, Default)

Process(Dataset,Default)

        While (Dataset contains cases of class other than Default)
          Target Class =Second most frequent conclusion other than default
          MakeRule (Target, Default)
          SelectedCases = Cases from Dataset covered by rule
          Process (SelectedCases,Target)
          Remove SelectedCases from Dataset
        end while

MakeRule (Dataset,Default)
        Rule = { }
        lastprobability = 1

        While (Number of attributes in rule is less than the Total Number of attributes)
          Find the attribute-value pair Av1 not already in rule which in conjunction with the
                  clauses in rule produces the minimum measure of correctness r , rmin 1
          If (rmin1 > lastprobability)
                  Add Av1  to rule
                  lastProbability = rmin1
          else
                  Find the attribute-value pair Av2 not already in rule which in conjunction  with
                        Av1  and the clauses in rule produces the min. measure of correctness r,  rmin2
          end if
          If  (rmin2  < lastProbability/10)
                  Add Av1   and Av2  to rule
                  lastProbability = rmin2
          else
                  exit
          end if
        end while
```

Figure 4.2: The Vinduct Algorithm

4.1.2 An Overview of Rough Set Theory

Pawlak (1982) first introduced rough set concepts as a means of discovering relationships in data. The idea of consistency is central to RST and hinges upon the concept of indiscernability rather than the elusive concept of truth. We are concerned that the condition attributes (P) are able to discern the decision class Q_i from another decision class Q_j, where $i \neq j$. RST has been used successfully to support the classification of uncertain or incomplete data in a range of domains (Fibak, Slowinski and Slowinski 1986, Krysinski 1990 and Slowinski, Slowinski and Stefanowski 1988). Pawlak defines a knowledge base to be a family of classifications[24] in the universe of objects being investigated. Each family member forms a subset or category. Membership of a subset is sometimes uncertain and the concepts of upper and lower approximations, $\overline{R}X$ and $\underline{R}X$ respectively, are used to approximate membership, where X is the concept and R is the knowledge held. If a concept is known, that is all objects with particular attribute value (A-V) pairs belong to the same classification, then those objects belong to the lower approximation and form the positive region. More formally we define $POS_R(X)$ as the set that can be certainly classified as X using knowledge R. Those objects that certainly don't belong form the negative region. If the information we have is not sufficient to classify an object then such objects belong to the boundary region. The upper approximation is the set theoretical union of the positive and boundary regions. Using these regions, classified sets are used to help approximate the classification of the unclassified set. This replaces the notion that set membership is a primitive relation with the view that the membership relation is based on what knowledge is held.

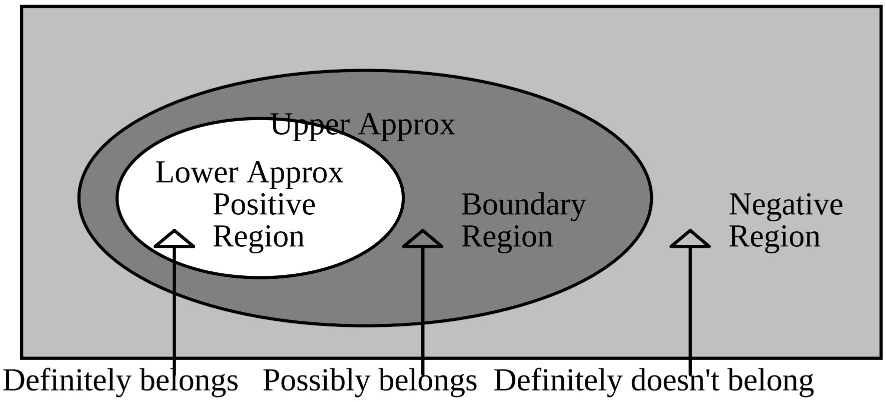


Figure 4.3: The Partitions in an RST Universe

The regions described above and shown in Figure 4.3 are used to find dependencies (rules) through the generation of the core and the reducts of a knowledge base. Reducts are collections of attribute value pairs by which a conclusion can be reached and can be seen as alternate rule pathways. The core is the intersection of the reducts. Calculation of the core and reducts can be used to simplify decision tables. This is achieved through the removal of redundant attributes (columns), rules (rows) and superfluous attribute values (Pawlak 1991, p.71).

---

[24]The discussion that follows is restricted to problems of classification, although the usefulness of RST for a construction problem is being investigated.

In the following example it is shown how reducts and cores are found using RST. The data used in this example are taken from an article which appeared in the Sydney Morning Herald (SMH) during the constitutional convention held in February 1998 in Canberra. There are six parties and thirteen issues which are shown in Figure 4.4 and referred to as objects and attributes, respectively. The data have been simplified into a decision table as shown in Table 4.1.

1- Breakaway Republican 1(Tim Costello)	a -President as Head of State
2- Breakaway Republican 2 (Peter Beattie)	b -Governor General as Head of State
3- Aust. Rep. Movement (ARM)(Malcolm Turnbull)	c -HOS political
4- McGarvie Model (Richard McGarvie)	d -HOS not political
5- Monarchist (Kerry Jones)	e -Popular Election
6- Non Aligned (Phil Cleary)	f- President elected by 2/3rd Parliament
	g -Self Nominating
	h -Nominees Voted
	i -Nominees voted for by a mixed council
	j - Nominees chosen by parliament
	k -Nominee chosen by PM
	l- HOS appointed
	m - HOS appointed by Queen
	n - HOS appointed by 3-person council

Figure 4.4: The parties (objects) and issues (attributes) in the Head of State Debate

U	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*
1	1	0	**1**	**0**	1	0	1	**1**	1	0	**0**	**0**	0	0
2	1	0	**1**	**0**	1	0	0	**1**	0	1	**0**	**0**	0	0
3	1	0	**1**	**0**	0	1	0	**0**	0	0	**1**	**0**	0	0
4	0	1	**0**	**1**	0	0	0	**0**	0	0	**0**	**1**	0	1
5	0	1	**0**	**1**	0	0	0	**0**	0	0	**0**	**1**	1	0
6	1	0	**1**	**0**	1	0	0	**1**	1	0	**0**	**0**	0	0

Table 4.1: The Decision Table for the Selection of a Head of State. The attributes shown in bold are later removed using RST concepts as they are dispensable.

Decision tables may be reduced through a three step process (Pawlak 1991, Chptr Six):

1. Eliminate duplicate rows.
2. Eliminate superfluous attribute values.
3. Compute the reducts of condition attributes.

Since there are no duplicate rows (each party has a different viewpoint to each other party) we can not reduce the table using step 1. We can however remove superfluous attribute values. As mentioned, discernability is a key factor in RST. From table 4.1 we can see that some issues (attributes a and c; b, d and l; e and h; f and k) have the same values for each party. Duplicate columns are shown in bold and can be removed. Discarded attributes are termed dispensable as they do not affect the decision outcome. We can also see that the table remains consistent (that is, the same condition attributes do not result in different decision attributes) by dropping either attribute a or b. After the various attributes above have been dropped we end up with Table 4.2.

U	*a*	*e*	*f*	*g*	*i*	*j*	*m*	*n*
1	1	1	0	1	1	0	0	0
2	1	1	0	0	0	1	0	0
3	1	0	1	0	0	0	0	0
4	0	0	0	0	0	0	0	1
5	0	0	0	0	0	0	1	0
6	1	1	0	0	1	0	0	0

Table 4.2:Reduced set of attributes

From Table 4.2 we compute the core values which represent the minimal characteristics of each participant. How this is performed is now given for the first object. The reducts and core values for each other object are found using the same process. We consider each attribute at a time and find the set of objects that have the same value as the first object for the attribute being evaluated. The parties are identified by their number in Table 4.1.
This can be shown as:

$\{[1]_a, [1]_e, [1]_f, [1]_g, [1]_i, [1]_j, [1]_m, [1]_n,\} = \{ \{1,2,3,6\},\{1,2,6\}, \{1,2,4,5,6\},\{1\}, \{1,3,4,5,6\}, \{1,2,3,4,6\},\{1,2,3,5,6\}\}$

We are interested in finding the intersection of these sets to determine what issues Breakaway Republican 1 shares with the other parties. $[1]_{a,e,f,g,i,j,m,n} = [1]_a, \cap [1]_e, \cap [1]_f, \cap [1]_g \cap [1]_i \cap [1]_j \cap [1]_m \cap [1]_n = \{1\}$. This set also represents the decision category since we are treating each attribute as a condition and decision attribute. To find the reducts and cores we drop one attribute at a time to see if the intersection of the remaining attributes is in the decision category. For example:

1. $[1]_a, \cap [1]_e, \cap [1]_f, \cap [1]_g \cap [1]_i \cap [1]_j \cap [1]_m = \{1\}$
2. $[1]_a, \cap [1]_e, \cap [1]_f, \cap [1]_g \cap [1]_i \cap [1]_j \cap [1]_n = \{1\}$
3. $[1]_a, \cap [1]_e, \cap [1]_f, \cap [1]_g \cap [1]_i \cap [1]_m \cap [1]_n = \{1\}$
4. $[1]_a, \cap [1]_e, \cap [1]_f, \cap [1]_g \cap [1]_j \cap [1]_m \cap [1]_n = \{1\}$
5. $[1]_a, \cap [1]_e, \cap [1]_f, \cap [1]_i \cap [1]_j \cap [1]_m \cap [1]_n = \{1,6\}$
6. $[1]_a, \cap [1]_e, \cap [1]_g, \cap [1]_i \cap [1]_j \cap [1]_m \cap [1]_n = \{1\}$
7. $[1]_a, \cap [1]_f, \cap [1]_g, \cap [1]_i \cap [1]_j \cap [1]_m \cap [1]_n = \{1\}$
8. $[1]_{e,} \cap [1]_f, \cap [1]_g, \cap [1]_i \cap [1]_j \cap [1]_m \cap [1]_n = \{1\}$

U	*a*	*e*	*f*	*g*	*i*	*j*	*m*	*n*
1				1				
2						1		
3			1					
4								1
5							1	
6				0	1			

Table 4.3: The core values - the core issues

We can see that the results for all equations except number 5 (the one that dropped attribute g) are within the decision category {1}. Thus, those attributes form the reducts for the first object (Breakaway Republican 1). For example in the first equation the reducts are a, e, f, g, i, j and m. Each result within the decision category offers an alternative set of reducts (or rules). The core is the intersection of the reducts which results in the core value being g(1) = 1 (meaning the value of attribute "g" for object "1" is "1"). When the cores are computed in the same manner for all parties we get the cores shown in Table 4.3. A textual description of Table 4.3 is given in Figure 4.5 and reveals the key issues for each party. It must be noted that the cores found using RST are not necessarily the core or key issues in the decision but are what differentiates one party from another.

Breakaway 1:	HOS should be self nominated.
Breakaway 2:	Parliament should select 3 nominees for HOS.
ARM :	HOS should be elected by a 2/3rds majority of Parliament.
MCGarvie :	3 person council appoints the HOS.
Monarchist :	Queen appoints the HOS
Non Aligned :	Nominees voted by mixed council.
	HOS should not be self nominated.

Figure 4.5: The key issues in the HOS Issue

In this study RDR KBS have been converted into decision tables which were used as input and RST has been used to find the reducts which form the new and reduced set of output rules. Cores were not computed as they tend to strip out too much relevant information. The various cleanup algorithms described later in Section 4.2.2 and shown in Figure 4.6 altered the content of the decision tables and thereby affected the output but the process described in this section for computing reducts was not changed.

Previous work using rough sets has been concerned with analysis of cases rather than rules[25]. However, it seemed that RST would be eminently suitable for the compaction of rule sets since RST is concerned with reducing the initial set of consistent algorithms to a smaller perhaps less consistent set through the removal of multiple instances (objects in rough set terminology) and redundant attributes. RST is able to handle missing values and the rules developed are not affected by the number of each example as are many other ML techniques, such as C4.5 or Induct. The implementation of RST used in these experiments will be referred to as RS/RDR and comprises C code which converts an RDR KBS into a flat decision table structure and back again and routines, also written in C, from the Rough Sets Library (RSL) (Gawrys and Sienkiewicz 1993) which generate reduced rules from the decision table. The interface, compaction and verification procedures and code are the same as those used by Vinduct.

4.2 The Experiments

As stated previously, it was critical that compaction be achieved without loss of the existing knowledge acquired from the expert which meant no loss in classification

[25]An exception is research by Colomb and Sienkiewicz (1995) which used RST to reduce the *inflation* that can occur when a propositional ES is converted into a decision table.

accuracy. Numerous experiments were conducted to determine the most appropriate technique for compaction. The factors affecting the design of the experiments are discussed in the next section followed by a description of the experiments.

4.2.1 Design Considerations

An RDR knowledge base consists of a set of rules together with a set of cornerstone cases, that is, the cases which resulted in the creation of a rule. It was important that we did not lose information needed to maintain the existing classificatory accuracy of our KBS. Cases and rules together provided information that was not available in cases or rules alone. The cornerstone cases provided the context in which a rule applied and the rule showed the salient features in the case. Use of the cornerstone cases alone was not appropriate for training as they only represented a small number of the total cases and may have contained attributes not used in determining a conclusion. By filtering the cases using the rules it was possible to extract the irrelevant attributes from the set of cornerstone cases. It was anticipated that the cleaned up set of cases would be a suitable training set for a machine learning algorithm, like Induct/RDR, which would produce a smaller knowledge base that performed at least as well as the original knowledge base.

It is an inherent feature of RDR that the size of the knowledge base produced is dependent on the order in which the cases are presented and will affect the amount of repetition. To ensure that the worst case is obtained in terms of repetitious knowledge, the data used in the following experiments were randomised to produce several datasets containing the same data but with different ordering of the cases.

The study was carried out on three different domains using the following datasets : Chess and Tic Tac Toe from the University of California Irvine Data Repository and the Garvan data from the Garvan Institute of Medical Research, Sydney.

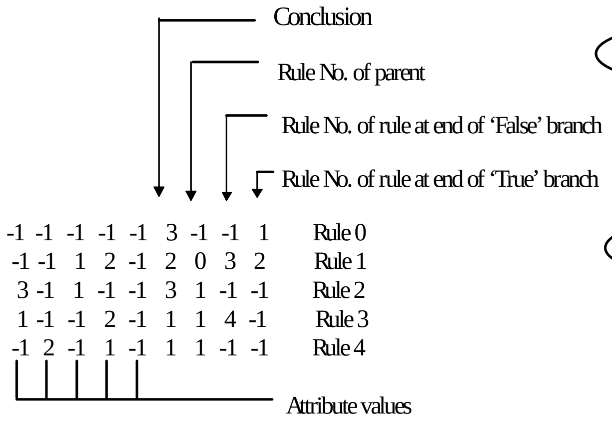
4.2.2 Experimental Method

Compton, Preston and Kang (1995) proposed the use of another KBS as a simulated expert from which knowledge can be acquired because of the unavailability of experts for controlled studies. This study attempts the compaction of a knowledge base built using a simulated expert as the source of expertise. A different synthetic expert was built for each of the domains, using machine learning techniques. The whole dataset was used as the training set so as to provide maximum expertise. The three datasets were then randomised prior to the construction of the expert systems to avoid ‘lumpiness’ (Compton, Preston and Kang 1995). The Garvan dataset was randomised to produce four different datasets. Both the Chess and TicTacToe datasets were randomised to produce nine different datasets. For each dataset 25% was kept as the test set, while the other 75% was used as the training set.

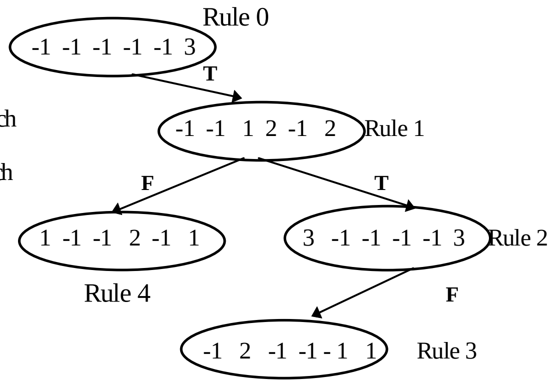
For each of the randomised datasets of each domain, fractions of the training set were used to produce knowledge bases of varying sizes using the method described in Compton, Preston and Kang (1995). New rules were incrementally added to the expert system under construction whenever the conclusion provided by the expert system for a case was found to disagree with the conclusion reached by the simulated expert. The rules were created by selecting four conditions from the intersection of the rule trace of the simulated expert, for the case and the difference list for the case. The difference list consists of conditions in the misclassified case that differed with the cornerstone case of the rule in the expert system that caused the misclassification of the case. All the conditions in the intersection were used in rule formation in the event of the intersection containing less than four conditions. An empty intersection resulted in four conditions being selected from the difference list or less if four were not available.

For each of the resulting knowledge bases, the cornerstone cases were filtered to remove all the irrelevant attributes. The clean-up process involved taking a set of cornerstone cases, a set of rules and running each case against the rules keeping for each case only those attributes that caused a rule to either fire or those that caused a rule to misfire, while replacing all other attributes by missing values. Figures 4.6.(c) - (e) shows the steps involved in removing the irrelevant attributes from the case ‘ 2 1 1 2 3 2’ by using the ruleset in figure 4.6(a). This method of combining cornerstone cases and rules and creating a new *case* with relevant A-V pairs will be referred to as Cleanup 1. From figure 4.6, we can see that a case was initialised to missing values (-1) and then traversed the RDR tree, adding back case values for attributes used in rule conditions.

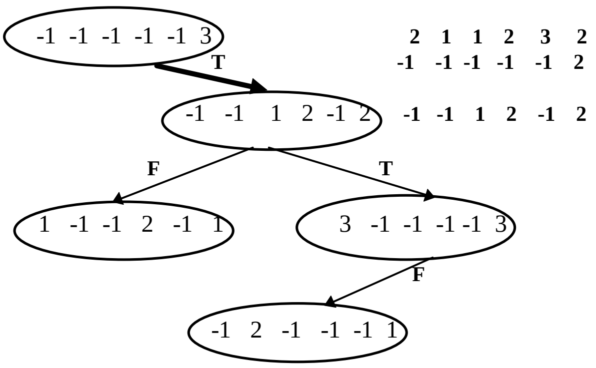
Cleanup 1 captured the attribute value pairs that determined the path. An alternative approach, known as Cleanup 2, applied to the handling of conditions on the false branch and attempted to generalise and replace a value that caused a rule to fail with the negation of the value in the rule. This meant that if the possible values were 1-4, the value in the case was 1, the value in the rule was 2, the value in the cleaned-up case was -2 and would treat the values 1,3 or 4 as matches. This was seen as a better interpretation of the knowledge. It was assumed that when the expert gives the attribute in the rule a particular value, the expectation is that any cases with all values except for the one specified will fail. Where only two values were possible such as true or false, if the value in the rule was false, the value in the case was taken. This is the same as Cleanup 1. To reduce complexity and facilitate the techniques described as Cleanup method 1 and 2, the initial KB were converted into a format where attributes only had positive values and negation was not present.



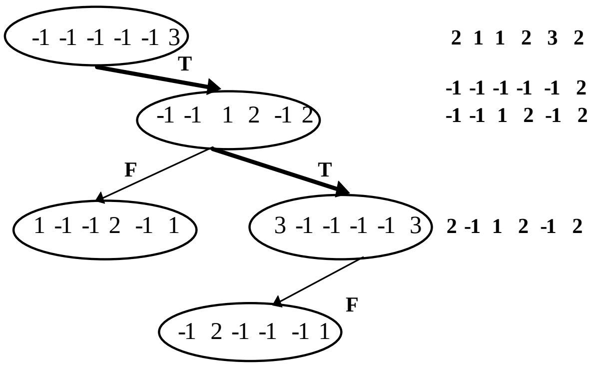
(a) A sample ruleset. The first five numbers are values assigned to the five rule condition attributes. The sixth-ninth numbers are the conclusion, the parent rule number, its leftchild, and rightchild. -1 indicates the absence of any value



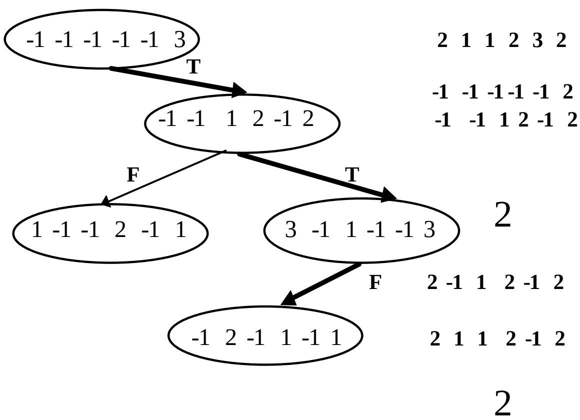
(b) The binary tree representation of the ruleset given in (a)



(c) *The case '2 1 1 2 3 2' is run through the tree to extract relevant attributes. Rule 0 fired unconditionally, so no attribute in the case is responsible for the rule firing. Rule 1 fires because of the third and fourth attribute so these are kept in the filtered case.*



(d) Rule 2 fails to fire because the original case has value 2 instead of 3 for the first attribute. So the new filtered case is ' 2 -1 1 2 -1 2'



(e) Rule 3 fails to fire because the case has value 1 instead of 2 for the second attribute.

Initial case : 2 1 1 2 3

Filtered case : 2 1 1 2 -1

Figure 4.6: The Steps in Cleanup Method 1

Filtered sets of cases, using Cleanup 1 and Cleanup 2, were then used as the training sets for Vinduct and RS/RDR. This resulted in a set of reduced knowledge bases. The percentage error of the original knowledge base and the compacted knowledge base on the test set was then determined. Plots of the original knowledge base size and compacted knowledge base size against the percentage of total cases used as the training set were obtained for each randomised dataset of each domain. The percentage error of the original knowledge base and the percentage error of the compacted knowledge base versus the percentage of total cases used as the training set were also plotted. One other method of compaction, termed Flat True rules, was used with RS/RDR which took only the true branches and ignored the false branches. The purpose of this was to gain some understanding of the impact of the false branches. The results, discussed later, are interesting.

Care was taken that the only difference between the use of Vinduct or RS/RDR was with respect to how the rules were generated. Both techniques made use of a default diagnosis to catch cases not fired and the Cleanup algorithms and method of evaluation were identical, in most cases the same C routines were used.

4.2.3 Missing Values

The handling of missing values needs special attention because typically ML techniques like Induct and RST are used on cases rather than on rules or combinations of rules and cases. When we are dealing with rules we have many more missing values than we would expect to find in a set of cases. For Induct, a selection based on missing values are allowed to contribute to false positives but not to correct positives (Gaines 1989). Thus when measure r, is being calculated for a selector S, those cases having missing values for any of the clauses in S, are considered to belong to S if and only if they have a classification other than the Target class Q. Therefore, as shown in Figure 4.7, cases with missing values can affect the value of $\underline{s}$ but not that of $\underline{c}$.

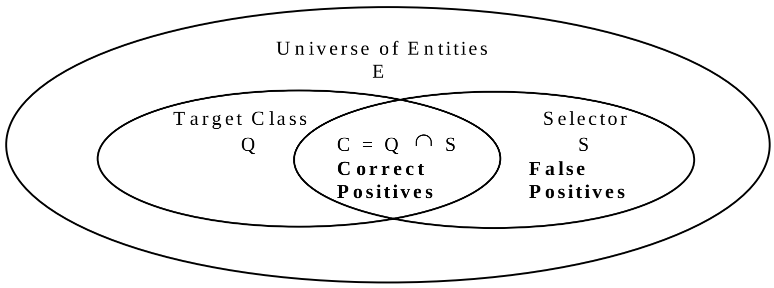


Figure 4.7- Set Representation of the Dataset (Adapted from Gaines 1989b)

When a rule has been induced and the dataset has to be divided into the 2 subsets consisting of those cases in the dataset that are covered by the rule and those that are not covered by the rule, cases with missing values for any of the conditions in the rule are considered to be in both subsets. This is good because it allows for the worst case since a case can be both covered and not covered by a rule depending on the value assigned to the missing attribute value. For the purpose of the repetition studies this approach allows us to treat rules as cases with missing values and using Induct we can reduce the rule-set (Gaines 1989b). The handling of missing values in RST is similar as they are treated as a match on all values. The main difference is that when a rule is generated a missing value is not used to discern two objects so they will not cause rules to be generated for each possible value and do not affect the generation of reducts (rules).

4.3 Results

The graphs of Figures 4.8(a) and 4.8(b) represent the growth curve and the error curve, respectively, for the knowledge bases built by the simulated expert (Initial KB) and the compacted knowledge base (Compacted KB) produced by Vinduct for the Garvan domain using the Cleaned Up (using Cleanup 1) cornerstone cases. The shaded area represents the range of values obtained from the four different randomised datasets. The x-axes of all the graphs represents the proportion of the total dataset used as the training set for the manual expert system. Similar graphs were produced for TicTacToe and Chess.

It is clearly visible from the graphs that compaction tends to decrease the size of a knowledge base by a factor of 2 or more. However, this promising result is shadowed by a considerable increase in error rate. For example, at 60 percent of the number of cases in the Garvan dataset, the knowledge base size decreases from an average of 800 rules to 250 rules. The error rate on the other hand, increases from an average of 3 percent to 20 percent. Cleanup method 1 is the one shown as the results taking the negation of the rule, Cleanup method 2, were worse. Table 4.4 shows similar figures for the Garvan data produced by RS/RDR. More description of the RS/RDR results is given below the table.

Table 4.5 provides a comparison of the results obtained from Vinduct and RS/RDR on the Garvan dataset randomised on seed 10. A synopsis of the figures accompanies the table. The results obtained on the TicTacToe and Chess datasets were similar to the Garvan results using Vinduct or RS/RDR.

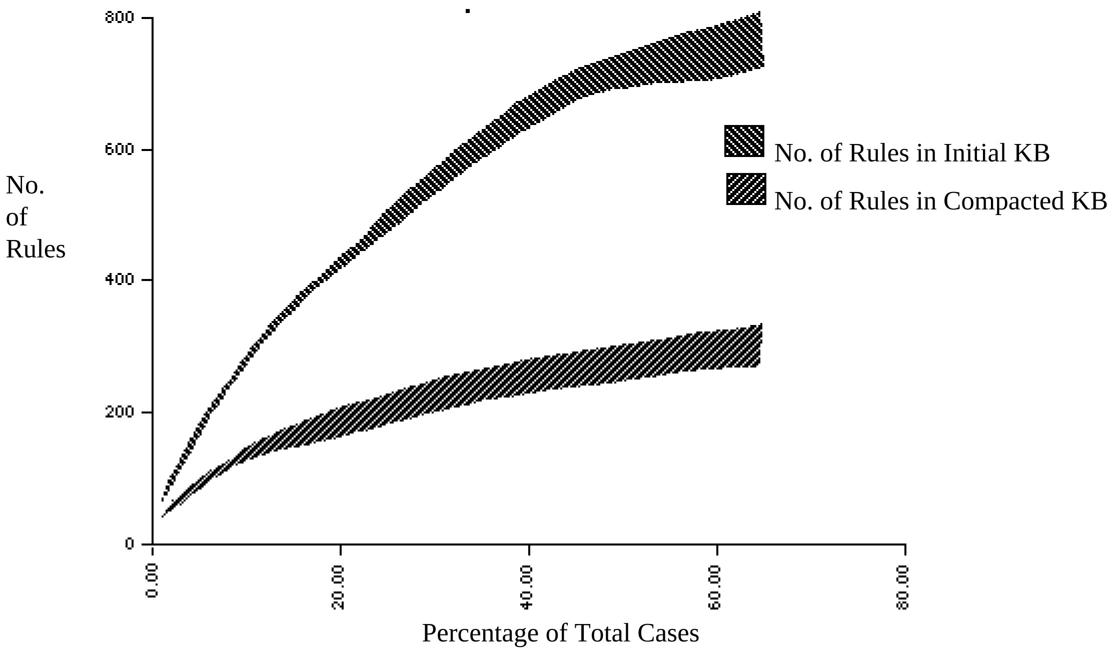


Figure 4.8(a): Growth Curve

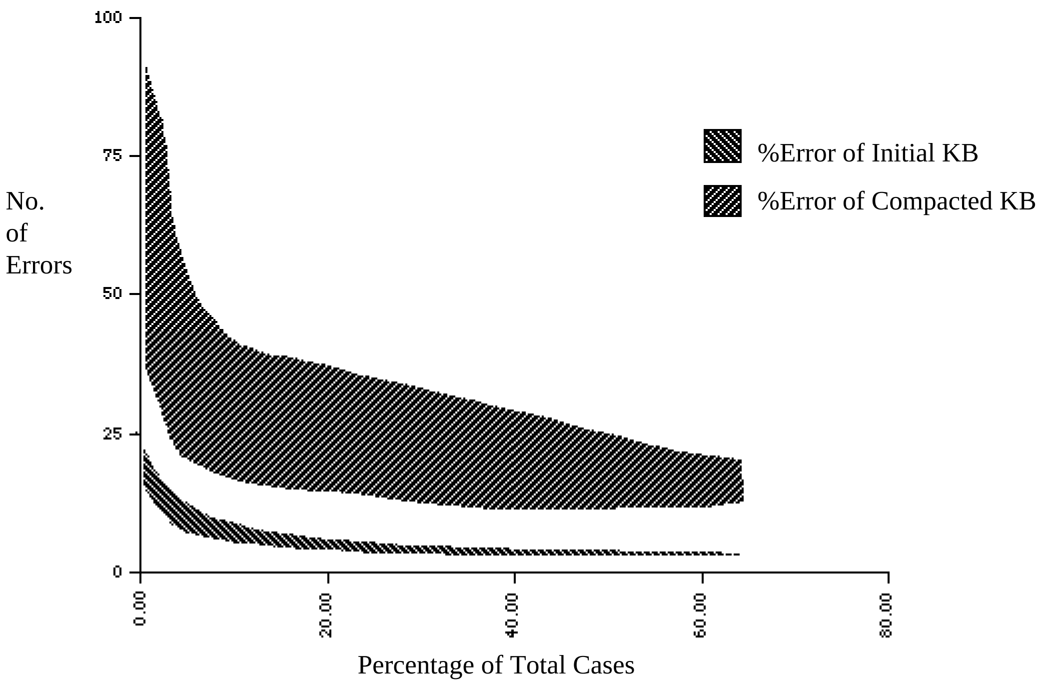


Figure 4.8(b): Error Curve

Figures 4.8(a) and (b): Comparison of Growth and Error Rates for the Initial KB and the Compacted KB (using Cleaned-up Cornerstone Cases) in the Garvan Domain using the Vinduct algorithm.

No. Rules	Random-isation Seed	%error Initial KB	True Flat rules #rules	%error self	%error testset	Cornerstone cases raw data #rules	%error	Cornerstone cases Cleanup 1 #rules	%error
100	10	9.74	63	46.94	50.47	48	33.04	61	47.88
	20	10.37	66	54.08	23.42	54	42.79	70	73.25
	30	10.09	64	44.90	26.28	59	38.83	66	41.12
	40	13.10	68	37.76	22.69	55	56.15	67	58.55
500	10	2.91	187	68.47	28.00	184	43.93	230	44.05
	20	3.01	245	61.65	23.95	170	36.27	254	40.87
	30	3.59	231	67.27	26.50	178	22.99	256	29.61
	40	3.37	249	61.65	23.15	172	42.33	263	45.34
958	10	0.86	319	71.89	26.83	172	42.34	386	32.01
	20	0.56	388	70.93	17.48	299	29.30	441	30.73
	30	0.34	319	71.54	26.83	290	29.99	457	20.27
	40	0.44	319	71.73	26.83	288	32.11	438	42.48

Table 4.4: The Garvan Results using RS/RDR on Cornerstone cases

Column 1 shows the number of rules in the Initial KB. Column 2 shows the seed used to generate random datasets. The third column gives the number of errors on the initial rules when verified against the test set, which was the last 25 percent of the 21822 Garvan cases. The fourth column shows the number of rules and percentage of errors on the test set when the rules were flattened, which meant taking only the true branch and ignoring any impact that the false branch might have. It is interesting to note that the errors substantially decreased on the test set but was very high on the cornerstone cases (self) associated with the rules that were flattened. The fifth column shows the number of rules and percentage of errors when the cornerstone cases alone, (ie. no attributes removed) are used to generate a reduced set of rules. The sixth column shows the reduced number of rules and percentage of errors when Cleanup Method 1 is used on the cases. In Cleanup 1, all attribute-values on true branches are kept. Attributes that are used by rules on the false branch and whose values differ from those in the cases are also kept. All attributes not used by the true or false rules are replaced by missing values. Where "self" is not shown the results were 100% accuracy on the associated cornerstone cases.

Cleanup 1 was the initial attempt at producing a compacted set of cases that only included the pertinent features of the data in the expectation that a smaller set of rules would be produced. However, the poor results from Cleanup 1 which used the value in the case for rules on the false branch resulted in the conjecture that the rule was not really saying take the value in the case but take the negation of the value in the rule. However, the results using both RS/RDR and Vinduct showed a further decrease in accuracy with this approach. On some datasets the number of rules generated exceeded the number in the Initial KB. This is similar to the ‘inflation’ problem described by Colomb and Sienkiewicz (1995) which results when we expand the negation to the range of possible values minus the one being negated. All results shown in tables and graphs refer to Cleanup Method 1.

All of the abovementioned results show higher error rates when Cleaned Up cornerstone cases are used, rather than straight cornerstone cases. All error rates were unacceptably high. It had been desirable to see if the approach would work on cornerstone cases as this reduced the storage and processing requirements and the cornerstone cases provide a smaller set of possible A-V pairs, thus reducing the search space for good classifiers. However, the poor results necessitated a whole new approach. Instead of using the cornerstone cases in the clean up process, the original cases from the dataset were combined with the rules generated by the simulated expert to produce a new set of Cleaned Up cases. For example, rather than compacting the 88 cornerstone cases and 88 rules generated by the simulated expert for the first 545 cases, the 545 cases were compacted using the 88 rules and the 545 Cleaned Up cases were used as input.

# Cases used to learn	Initial # rules	KB %error	Raw corners #rules Induct	Raw corners #rules RST	Raw corners %error Induct	Raw corners %error RST	Cleaned corners Cases #rules Induct	Cleaned corners Cases #rules RST	Cleaned corners Cases %error Induct	Cleaned corners Cases %error RST
545	88	11.23	36	44	52.33	31.48	43	53	39.58	41.92
1090	138	9.43	50	65	33.82	25.89	58	81	29.66	49.23
3270	295	4.89	88	118	43.77	29.19	124	150	29.88	46.52
5455	402	3.39	117	148	37.47	44.20	156	190	20.23	42.07
10910	624	1.99	165	219	32.16	26.82	221	272	39.56	39.74
16365	796	1.26	205	259	40.49	25.39	268	339	9.38	31.53

Table 4.5: Comparison of the Garvan results using Vinduct and RS/RDR using Cornerstone Cases

The first column is the number of the 21822 Garvan cases used by the simulated expert to build the Initial KB. The number of cases represent 2.5, 5.0, 15.0, 25.0, 50.0 and 75.0 percent of the total dataset. The number of rules and percentage of errors for the Initial KB are given in the second column. All testing was performed on the last 25 percent of the 21822 Garvan cases. The same rules and cornerstone cases were used by Vinduct and RS/RDR to generate a reduced set of rules. Column three shows the number of rules generated by both algorithms when the cornerstone cases are used for training the system. There is at least a two-fold reduction in the number of rules, with Vinduct producing smaller rule sets. The fourth column shows the error rates on the new set of rules generated from the cornerstone cases. Neither algorithm performs well, but RS/RDR's error rate is lower on all but one set of cases. The fifth and sixth columns give the number of rules and percentage of errors when a Cleaned Up set of cases generated from the cornerstone cases and rules are used to derive a smaller set of rules. The number of rules generated by RS/RDR and Vinduct goes up and the number of errors also increases compared to the use of cornerstone cases alone, with RS/RDR performing the worst.

Table 4.6 shows that when the algorithms are applied to the original cases, RS/RDR produces minimally more compact KB than Induct/RDR but produces more errors until 50 percent of cases are seen and then performs better than both Induct/RDR and the Initial KB. At 25% of cases RS/RDR produces less errors than Induct/RDR on cleaned up cases and has less errors than both ML algorithms when used on straight cases. Both algorithms produce larger KBs with Cleaned Up cases than with cases alone, with Vinduct producing smaller rule sets than RST. The error rates for Induct/RDR and

Vinduct are about 2% with and without Cleaned Up cases and therefore about 1% higher than the Initial KB. RS/RDR error rates drop using Cleaned Up cases as opposed to straight cases, showing a marked improvement over the Initial KB after 50 percent of cases are seen. The error rates for both algorithms are much closer to those in the Initial KB and suggests that this is a feasible approach to reducing repetition from manually built KBS that are incomplete and where a sufficiently large and complete set of cases is not yet available for machine learning.

Percent of Cases used to learn	Initial KB #rules	Initial KB %error	Cases only #rules		Cases only %error		Cleaned up Cases #rules		Cleaned up Cases %error	
			Induct	RST	Induct	RST	Induct	RST	Induct	RST
2.5	88	11.23	34	22	11.69	23.25	45	51	10.13	13.41
5.0	138	9.43	59	35	8.33	20.18	72	83	8.47	10.68
15.0	295	4.89	128	117	5.73	8.96	156	172	6.73	6.12
25.0	402	3.39	171	165	4.34	5.00	203	241	5.22	4.19
50.0	624	1.99	268	226	2.42	4.12	318	343	3.48	1.69
75.0	796	1.26	296	281	1.96	0.86	363	440	2.01	0.56
100.0	958	0.86	358	328	0.00	0.00	428	503	0.05	0.11

Table 4.6: Compaction on the Garvan Data using unmodifed Cases rather than Cornerstone Cases.

Columns four and five give the results produced from using the number of cases indicated in the first column and the straight Induct/RDR or RST algorithms. Columns six and seven show the number of rules and percentage of errors, respectively, produced by Vinduct and RS/RDR when the number of cases in column one are cleaned up using the rules in column two as input. The final row is included to show that when all cases are used for testing only the ML algorithms have no errors. In the case of cleaned up cases this is because the cases have been altered and in the case of the initial KBS manually built KBS may have errors as rules are only developed for the cornerstone cases and misclassification of other cases may have been missed.

4.4 Discussion of the Repetition Experiment Results

The results shown in Tables 4.4 and 4.5 using Vinduct and RS/RDR on the cornerstone cases show a decrease in the size of the knowledge base, which was accompanied by a significant decrease in performance of the system in terms of error rates between the Initial KB and all compacted KB, whether derived from straight cornerstone cases or cornerstone cases that had been filtered using one of the cleanup variations. More promising results are shown in Table 4.6 and Figures 4.9(a) and 4.9(b) where, for the Induct algorithm, the combined use of cases and rules to cleanup the original dataset obtains similar results to those achieved using ML alone on the same set of cases, producing a knowledge base roughly half the size of the original RDR knowledge base and the error rate for a fully matured KB went up by roughly 1%, from 1% to 2%. These results show both ML algorithms produce more compact rule sets. Induct performed similarly to the Initial KB for all percentages of cases seen. Error rates for RS/RDR were worse for both cases and cleaned up cases initially but after 25 percent of cases are used for training the error rates are lower than Vinduct on the cleaned up cases and both ML algorithms on straight cases. When 50 percent of cases are used for training RST/RDR shows a substantial improvement in the reduction of errors, which even exceeds the performance of the Initial KB.

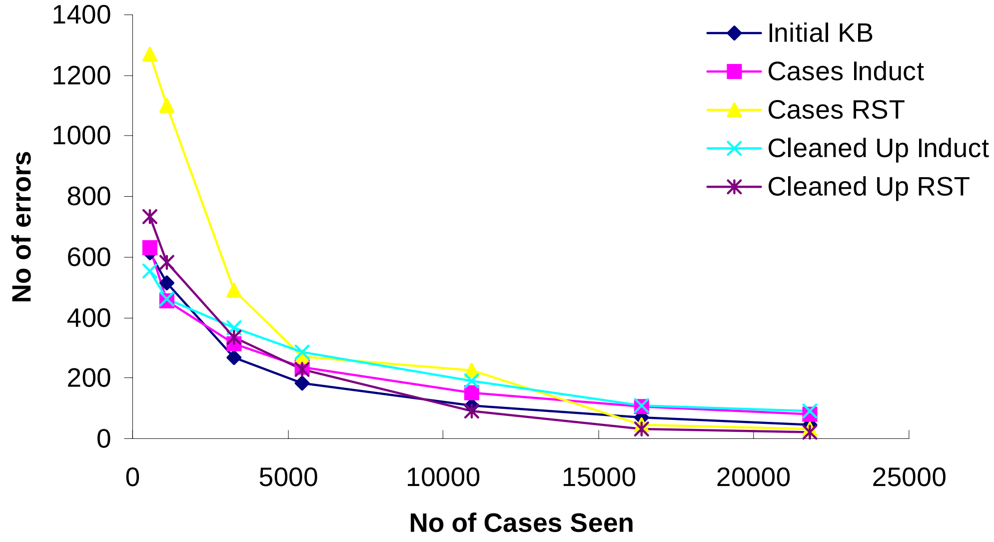


Figure 4.9(a): The Number of Errors for Garvan using Cases

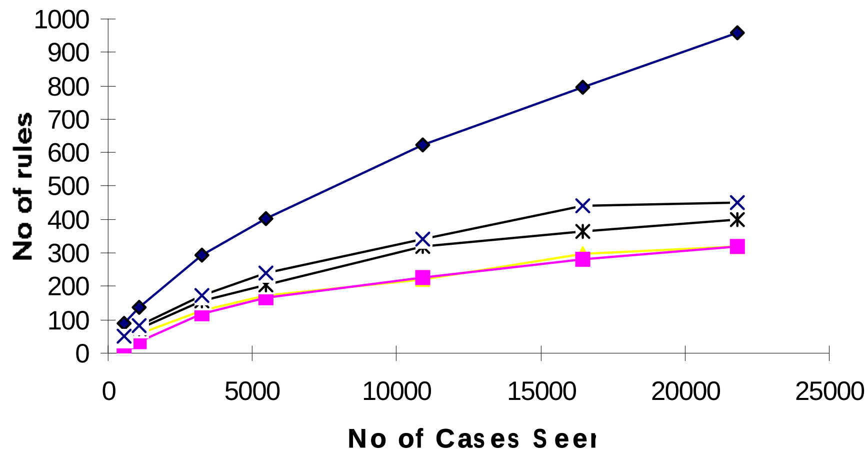


Figure 4.9(b): The Number of Rules for Garvan using Cases

The high accuracy levels and small KB size produced by ML, support the use of ML from cases for the population of knowledge bases and questions the need for the extra filtering step involved in either cleanup methods. However, as previously noted, cases are not always available and they may be unintelligible to an expert. As pointed out, RS/RDR's error rates were lower than the ML algorithms when only 25 per cent of cases have been seen. The overall improved results using all cases seen so far rather than just cornerstone cases supports Brian Gaines suggestion (personal communication) that all cases, not just cornerstones, seen by a knowledge base should be kept.

In the RS/RDR technique the original RDR KBS was being converted first into decision table format for input into the RSL routine. After rules (reducts) were produced they were being reconverted back into the RDR exception structure. It was a concern with the RS/RDR technique that the conversion from an RDR tree-structure to a flat structure and back again might alter the knowledge represented by the tree**.** Colomb (1993) has shown that conversion from a non-singulary decision tree, such as those produced by RDR, is equivalent to an unambiguous decision table. Various techniques were used to verify the conversion processes in this study. The new rules were evaluated for accuracy against the test set while flat (in decision table format) using one routine and again after conversion into RDR format. The results were within half a percent of one another, which given the large error rates using cornerstone cases, was not of great concern. Also each set of rules were evaluated against the test set as well as against the cornerstone cases in the Initial KB. Rules generated by RS/RDR and Vinduct from Cleanup Method 1 and the cornerstone cases resulted in 100% accuracy. Rules generated from the flat rule technique, only performed for RS/RDR, did not achieve high accuracy on the original cornerstone cases. This is expected since the flat rule method does not account for the knowledge in the false branches. The rules generated by Cleanup Method 2 using Vinduct covered all cornerstone cases, but this was not the case with RS/RDR and it has been found that proper handling of negation would require changes to the way reducts are generated (Sienkiewicz, personal communication), thus altering the original RST.

Another concern was that the structure of the new KB produced by RS/RDR was flat, with each new rule being added to the false branch of the previous rule. Two heuristics were applied to try to change this by adding more general rules at the top. The first heuristic was to add rules which covered the largest number of objects. If more than one rule covered the same number of objects a second heuristic was used which added the rules with the smallest number of conditions/clauses first. While these heuristics changed the order of the rules added, they did not appear to have much impact on the flat structure. An alternative not explored in these experiments is to use the Ghidora algorithm (Catlett 1992) which converts knowledge in a decision tree to RDR format.

An importance ranking (IR) is computed which gives the "expected benefit of the rule to the accuracy of the ruleset" (Catlett 1992, p.162). This IR measure is used to determine the order in which rules should be added.

Handling of the false branches has been noted as a problem in determining the most appropriate clean-up algorithm to use. A similar concern is how to handle negation. Cases usually have one or no value for each attribute. There was considerable effort required to both Vinduct and RS/RDR to deal with negation, which can be seen as the situation where all but one value can match. One attempt at avoiding making any changes to the RSL routines was to expand the negation. In the case where only two values were possible, such as true or false, if the rule condition said NOT FALSE the value in the case was taken, which would be TRUE. This is effectively the same as Cleanup method 1. Where more than one value was possible, a case was generated for each allowed value. This means duplicating each object n-2 times where n is the number of possible values. This will result in n-1 versions of the same object if only one attribute has a negated value. There are 32 condition attributes in the Garvan data. Thirteen attributes take more than two values so the inflation becomes excessive. In an Initial KB of 100 rules, 162,120 cases were generated. The memory needed for the RST tables were prohibitive and any further investigations down that path were stopped.

The large number of errors prompted investigation of the cause of these errors. It was frequently found that a small number of rules accounted for the vast majority of the errors. It seemed possible that if a few rules were removed then the error rate may drastically reduce. To test this hypothesis, a small experiment was conducted on a rule base that had one particular rule responsible for over 4000 errors. In addition, the conclusion for this rule was diagnosis 28, which was known to be a diagnosis that was difficult to accurately predict due to the large number of situations in which it could occur. Since many of the incorrectly classified cases belonged to the default class, the offending rule was removed. The result was that another rule with the same conclusion then had a similar error rate. When this rule was also removed another rule became the major cause of incorrect conclusions. This time the conclusion was diagnosis 36, another difficult classification. At this point the exercise was abandoned as the approach did not seem to solve the original high error rate problem. Another approach considered, but not tested, was the modification of the compacted rule base by an expert who could provide the correct conclusion and rule refinements. This may still result in a smaller size KB with the original or better error rates. There does not seem any easy solution. It is likely that the offending rules would require numerous patches for so many misclassified cases.

It appears that these troublesome conclusions (28, 36 and 51) are used by many rules and that the rules are quite different within each class. This means that there are many ways in which conclusion 28 can be given and this makes the task of the machine learner more difficult. To exacerbate this problem, Induct determines the most common conclusion to find rules for first. Since these conclusions are the most common these poor rules will be added to the RDR KBS first and may produce a poor structure for the KBS. RST begins to find rules in the order in which classifications are found in cases and may have faired better with cleaned up cases than without since some of the irrelevant attributes were removed beforehand. RST is primarily concerned with being able to distinguish between classifications and sometimes strips out attributes that are actually important but with the given set of cases were not necessary for differentiation.

The results produced by the two ML algorithms were similar for the three domains, however each algorithm revealed various strengths and weaknesses. Table 4.6 shows that RS/RDR produced larger but more accurate KB from the cornerstone cases than Induct after 15% of cases are seen. The algorithm used by Induct uses probabilities and generalisation to determine rules for the most common classification and tries to find the attribute value pair that identifies cases with this classification. Once this rule has been added the cases are removed and new rules do not consider whether previously classified cases would now be misclassified. This problem is exacerbated by the way missing values are handled because a case may be treated as being covered by a rule (a false positive as shown in Figure 4.7) because a value was missing when in fact if the value were known the case would not be covered and a new rule would be needed to cover it. RS/RDR produces more specific rules that are generated from comparison of each case with every other case with the same conclusion. Only after evaluation of all cases are rules generated. Also as stated in Section 4.2.3, RST does not allow assumed values to missing values to contribute to the rule developed. Therefore, as confirmed by the figures in this study, the initial error rate with Induct is lower than with RST, but as more cleaned-up cases are used for training RST produces rules with less errors. Induct produces rules of around 2.5[26] clauses, which is bigger than the 1.3 clauses for manually built RDR but closer than for RS/RDR which averaged 3.3 conditions per rule. It is also interesting to note that when similarity indices were computed between sets of rules using the nearest neighbour algorithm described in Section 3.3.2.1 it was found that the rules in the KBs produced using rough sets were closer to the Initial KB than the rules in the KBs produced using Induct.

[26] An average of 3.8 clauses per rule is reported in Catlett (1992) from Gaines (1989a).

One reason the approach with cornerstone cases may have failed is that the sample size needed to be larger. It has been argued (Sammut 1996, personal communication) that the training set will never be large enough because not enough cases are kept. A similar view was given by one of the reviewers of Richards, Chellen and Compton (1996) who said that the study was destined to fail because it was founded on the false assumption that the cornerstone cases held some special meaning or value. As noted in the comparison between PCP and RDR no such view of cornerstone cases is held. While the cases which prompt a new rule to be added may be representative of a class of cases it is realised that there are possibly many other cases that could have been equally representative with different attributes and that it was essential to remove irrelevant attributes that do not contribute to classification. This is why the filtering process with the various clean up methods was tried. Just as it was not appropriate to use cornerstone cases alone it was also not possible to use the rules alone for reducing the size of the KB. The problem in a single classification RDR KB or any method that uses “if-not” or false branches is that we can not reconstruct the pathways without knowing what values caused the false path to be taken. We can not take the negation of all conditions on the false branch because some of the conditions may have been true. While all conditions in a rule must be true to fire we only need one condition to fail for the whole rule to fail. The combined use of rules and cases allows us to determine the actual values of attributes because we can take the value in the case on false branches. As noted in Section 3.3 the real problem remains, however, as we can not be sure that the failure is due to an irrelevant A-V pair or whether the failure is important in reaching that conclusion.

The greatly superior accuracy rates when using all available cases supports the criticisms against using cornerstone cases and favour the use of all available cases for compaction. When the reduced rules produced from either the straight cornerstone cases or cleaned up (using Cleanup 1) cornerstone cases were evaluated on the related set of cornerstone cases the accuracy was 100%. In contrast, the error rates were high when evaluated on the test set of 5457 unseen cases. When we are dealing with rules/cornerstone cases we have a much smaller training set of cases than when the whole dataset is used (using 75% of cases for training we had 796 cases using cornerstone cases or 16365 cases when using available cases). What seems even more relevant than the sample size is the need for the distribution of cases to be statistically appropriate when statistical methods such as Induct are used. As mentioned, there were a large number of cases in the Garvan KBS that were poorly classified and it is believed that this has affected the results achieved.

Another possibility for the poor results on the cornerstone cases is that the use of a KB produced by a simulated expert is not an appropriate starting point. It may be that the simulated expert does not produce repetition of knowledge to the extent or in the manner that a human expert would and that attempting to remove repetition is a futile exercise. Use of a system built by a human expert has been considered, but is difficult due to the presence of time course data, real data or incompatible file formats of existing manually built RDR KBS. In retrospect, the problems that were encountered with the experiments possibly could have been avoided or at least miminised by using multiple classification RDR because there are no ambiguous false branches in MCRDR.

The poor results obtained when straight or cleaned up cornerstone cases were used for learning implies that the structure of an RDR KB and the storing of cornerstone cases capture knowledge which is not easily re-represented without loss of predictive power. If the knowledge is to be reused it would seem desirable to make the knowledge more perspicuous and that this would involve removal of redundant knowledge and appropriate handling of the knowledge in false branches. The varying results obtained when using case values found in the conditions of true branches (Flat True rules), case values from attributes in true branches and false branches (Cleanup 1) and case values from attributes in true branches and the negation of values in rules that don't match on the false branches (Cleanup 2) prompt the questions: what knowledge is really being represented by the false branch and can the pathways be flattened without loss of information ? Cleanup 1 yielded better results than Cleanup Method 2, where the number of errors rose and in some cases the number of rules generated was greater than with the Initial KB. The increase in errors is unexplained, however the increase in the number of rules is possible when we expand the negation by creating multiple cases to cover the rule that has negated values. It is interesting that when only the true branches were used for filtering cases the error rates were much lower on the larger test set than on the cornerstone cases associated with those rules and the results with RS/RDR are better than with either Clean Up method that tried to capture the knowledge in the false branches when run on the cornerstone cases. It appears that the impact of the false branches decreases as the size of the dataset increases.

Two important concerns not addressed in this work are the question of when the reorganisation should be performed and how to proceed after the reorganisation has been done. The issue of when will depend largely on the individual KBS and the amount of redundancy that exists in it. As the whole process described for both RST and Induct is automated it would be possible to periodically determine the extent to which the KBS could be reduced. If the gain is considered worthwhile it could be performed. It is difficult to quantify the answer to “when to do” when the value of and answer to “if do” is still an open question. The second issue of how to proceed after reorganisation is of greater significance. The use of cornerstone cases for validation and assistance with KA is an integral part of the manual development of RDR. As new rules are manually added, the new case can be stored for future use but it would be necessary to develop a modified KA technique if the misclassified rule did not have an associated case. This issue has not been explored but would need to be resolved before reorganisation of RDR KBS became acceptable practice.

The results of these experiments support the removal of repetition using a ML algorithm such as Induct or RST on all available cases but the decision of whether to clean up the cases first depends on the number of available cases and their coverage of the domain. The important question that remained from these experiments was whether the reduced rules were *useful* and in what way. An evaluation of the comprehensibility of the compacted KBSs was performed but abandoned after it appeared that most of the examples chosen for comparison in each KBS (original, straight cases - Induct, straight cases - RST, cleaned up cases - Vinduct, cleaned up cases - RST) appeared to be unusual cases and the explanation provided by the conditions on the rule pathway were difficult to understand for all KBS. Again the problem was related to the anomalous classifications given to a large number of different cases which have distorted the results. To some extent this outcome is disappointing as a strong conclusion was desirable. The main reason that further investigation has not been pursued is that many of the problems were seen to be possibly related to the single classification RDR structure and research efforts were moving into the stronger MCRDR representation. In keeping with this movement to MCRDR, which is also the direction the rest of the work in this thesis has taken, a proposal for the removal of repetition from MCRDR KBS is now offered but the actual investigation is left to future research.

4.5 A Proposal for the Removal of Repetition in MCRDR KBS.

As noted in the last section the value of removing repetition from single classification RDR to support reuse of the knowledge for explanation activities was inconclusive and has not been pursued further as research interests have moved to the MCRDR representation. In this section a brief proposal for the removal of repetition from an MCRDR KBS is given. Since at this point in time there are few manually built MCRDR KBS available for testing, particularly large KBS that would have enough repetition to consider reorganisation, this section is a proposal of what could be done rather than a description of what has been done. It also appears that the MCRDR structure may be undergoing further change in the future so that experiments to restructure an MCRDR to have less repetition would be premature until the representation has stabilised.

The removal of repetition can be performed as a three step process.

1. Expand the rule pathways
2. Apply ML algorithm to remove redundancies and restructure KBS
3. Sort list of rules into MCRDR format.

The first step is to expand the rule pathways by taking each rule with its conditions and conclusion and adding all conditions of parent rules that can be traced back to the first default rule. This is the same process that is used in the following chapter to convert an MCRDR KBS into a crosstable for use by the FCA algorithms.

The second step is to use some algorithm to remove or modify rule pathways according to the type of repetition to be removed. For example we would definitely want to remove rules that had identical conclusions and conditions but we may or may not choose to remove rules that reach the same conclusion but the set of conditions in one rule are subsumed by the set of conditions in another rule. Depending on the types of redundancies to be removed we can select the appropriate algorithm. For this purpose we could again use Induct, RST or the CUT95 algorithms (Scheffer 1996) to reorganise our KBS. Without performing any experiments using these various algorithms, apart from the small experiment with RST described in section 4.7.2, it is not possible to say whether the resultant KBS would meet the requirement that KBS accuracy in classifying cases must not diminish.

After reorganisation of the knowledge it would be possible to use the flat structure to which exception rules could be added as new cases are seen. However, it may be desirable to convert the flat structure into an MCRDR to gain the potential benefits of exception structures such as the naturalness and comprehensibility of the representation by human experts; the support they offer for incremental changes; the use of context and the maintainability of the knowledge due to the localisation of changes (Gaines 1991a, Li 1991). To achieve this the algorithm by Scheffer (personal communication) outlined below is proposed to reformat the flat rules back into MCRDR format.

An alternative and possibly safer (less interference with the existing KBS) approach to these three steps is to define the types of redundancy to be removed and to look for those anomalies and remove them. For example, we can obviously remove any rules that are complete duplicates (same set of conditions and conclusion). We can also remove any subsumed rules which have a larger set of conditions but include all the conditions and have the same conclusion as another rule. Rules with unnecessary premise conditions can be modified so that the redundant condition is dropped and then the duplicate rule can be removed. By targeting particular anomalies and fixing or removing those, rather than a complete reorganisation of the KBS using a ML algorithm, we should be able to maintain our current accuracy levels, save most of the history of changes stored in the KBS structure and keep the relationship between rules and cases that is not supported in the ML approach. We can even avoid the first and third steps of flattening and then reformatting the RDR KBS if the clean-up strategy used updates the various parent and child links to reflect the changes. For example, if a rule which is part of another rule's pathway is dropped the rule left must be modified to include the dropped rule's conditions unless they are redundant. A maintenance strategy to ensure that there is no loss in consistency, completeness and accuracy of the MCRDR KBS is not proposed here but the alternative of directly updating the KBS is worth considering especially since it should be possible to retain the links between rules and cornerstone cases that already existed.

As noted in Section 4.4, it was found in the single classification RDR repetition studies that the KBS produced from the machine learners tended to be flat structures with all rules except the default rule being a leaf node on the tree. The algorithm shown below is designed to produce KBS that take advantage of the MCRDR exception structure. It is speculated that this algorithm could be used as a way of producing MCRDR format KBS from single classification RDR. The main problem anticipated with this suggestion relates to the problem of how best to expand the rules in a single classification RDR KBS as the best interpretation of the false branches is not clear. Whether single or multiple classification RDR KBS are used as input the following algorithm could be employed for the output of MCRDR KBS.

4.5.1 Sorting Lists of Rules into MCRDRs

After steps One and Two have been performed to remove repetitious and redundant knowledge the following algorithm is designed to produce MCRDR structured KBS.
The expected input to and desired output from step Three is:

Input- A list of rules with possible overlap. That is, some rules premises may be subsets of the premises of other rules. Figure 4.10 shows the possible input.

Output - An MCRDR structure such that every rule premise which is covered by another rule is explicitly marked as an exception. Figures 4.11 and 4.12 show the intermediate and final output, respectively.

For example:

Input:
$a \rightarrow +$
$ab \rightarrow -$
$b \rightarrow +$
$abc \rightarrow +$

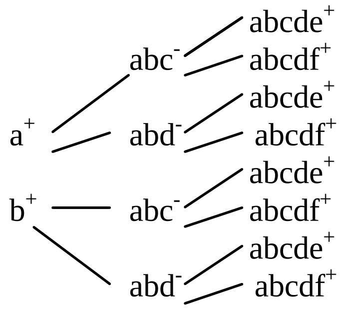
Output:
a^{+} —— ab^{-} —— abc^{+}
b^{+} —— ab^{-} —— abc^{+}

As can be seen above ab is subsumed by a and by b and abc is subsumed by a, b and ab. In the worst case: if we have two rules on the first level, two rules on the next level and so on, all rules of level *i* cover all rules of level *i+1*, $\rightarrow$ n/2 level $\rightarrow$ $2^{n/2}$ rules in MCRDR.

For example:

Input: a^{+}
b^{+}
abc^{-}
abd^{-}
$abcde^{+}$
$abcdf^{+}$

Output:



The following sorting list algorithm is described where:

"more-than-general-graph"- see figure 4.10
V= vertices and E= edges.
input degree zero = there are no input edges.
"dangling edges" = edges that do not connect to a vertice.

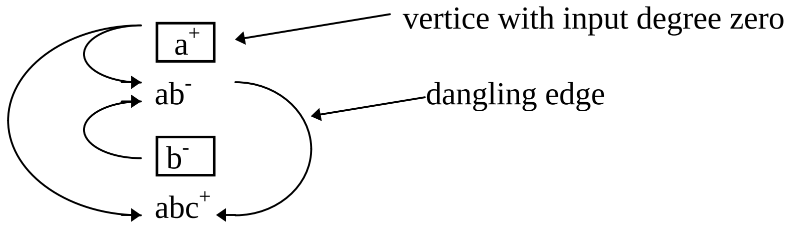


Figure 4.10: More-general-than-graph

Algorithm:
Set up "more-general-than-graph" (V, E)
All nodes with an input-degree of zero become rules, ie the graph vertices.
Do while no more "dangling edges"
look at only the subset of nodes V′ s.t. (V,V′) ∈ E
E′ = E - "dangling edges"
All nodes with input-degree zero are the rules (exceptions to V)
End While

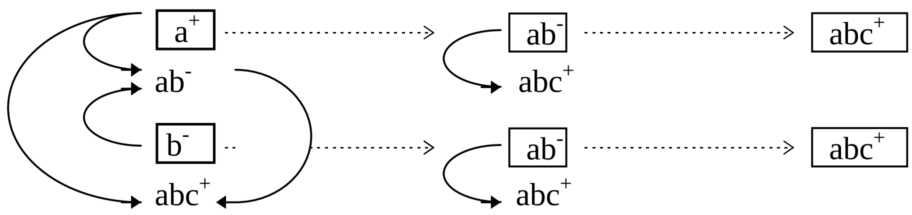


Figure 4.11: Final expanded "more-general-than-graph"

$$\boxed{a^{+}} \dashrightarrow \boxed{ab^{-}} \dashrightarrow \boxed{abc^{+}}$$

$$\boxed{b^{-}} \dashrightarrow \boxed{ab^{-}} \dashrightarrow \boxed{abc^{+}}$$

Figure 4.12: The final MCRDR Structure

The three-step process and sorting algorithm described above have not been implemented. As was found in the removal of repetition from single classification RDR KBS it is envisaged that modifications and alternative methods will need to be considered. This brief proposal has been offered as suggested future work that will best be tested when manually built MCRDR KBS of reasonable size are available and when the optimum MCRDR structure has been determined. Both of these events will be realised in the not too distant future.

4.6 Summary of the Repetition Studies

Some experiments using the Induct and Rough Set algorithms to remove repetition were described in sections 4.1 to 4.4. The results using cornerstone cases were poor but the results using all available cases (this does not mean the full dataset but all cases seen at various stages of KBS maturation) were favourable, producing KBS at least half the size with error rates the same or better than the original error rates once 50 per cent of cases were seen using RST and increased error rates of about 1 per cent using Induct. The benefits of removing repetition for explanation purposes, which was the motivation for the studies, remains an open question. The repetition experiments concerned single classification RDR and the mixed results can be attributed to some of the ambiguity of the false branches together with certain features of the algorithms used. As research interests have moved to the newer MCRDR representation further experimentation with alternative algorithms has not been pursued but a brief proposal for the handling of repetition of knowledge in MCRDR was given in Section 4.5. The proposal was not implemented as it was felt that more MCRDR KBS need to be developed first and changes to the MCRDR are likely in the near future.

This does not mean that the experiments were fruitless. The experiments raised many issues concerning the single classification structure and the ML algorithms that were used. This discussion was given in Section 4.4. The investigation of RST has also led to other benefits. As stated in the opening section of this chapter, two of the benefits of inductive learning are the removal of redundancy and inconsistency and the detection of missing knowledge. The repetition studies have been concerned with the redundancy issue. However, while exploring the potential use of RST for knowledge base compaction and reuse it was found that the routines provided in the RSL could be used to identify and remove various other verification anomalies. An algorithm was developed and applied to a subset of the rule base for AusVit, the Australian Viticulture Expert System (Richards, Gambetta and Compton 1996). This algorithm has also been applied to the Garvan, TicTacToe and Chess datasets to verify consistency and remove redundancy. In the following section we look at the use of RST for verification of rule-based systems in general and in section 4.7.2 we look at a modified approach to support the verification of MCRDR KBS.

4.7 Using Rough Set Theory for Off-line Verification[27]

In Section 4.7.1 we review work that has been done using RST for verification of traditional rule-based systems and an extension of that approach to handle MCRDR in Section 4.7.2. The process is referred to as off-line verification because it is a batch process in contrast to the existing verification method, as described in Section 3.2.3 which occurs incrementally and online as the rule base is being used.

4.7.1 A General Approach to Verification of Rule-Based Systems.

The work by Colomb and Sienkiewicz (1995) on using rough set theory to reduce the inflation problem when converting from a decision tree or propositional KBS into a decision table using rough set theory inspired the use of RST for finding the key concepts and functional dependencies between attributes in a rule-base. During these investigations it appeared that RST was eminently suitable for finding verification anomalies. In this section we consider a number of common verification anomalies: redundancy, inconsistency (ambivalence), imcompleteness (deficiency) and circularity and how they are addressed by RST.

RST defines redundancy in the same way as Preece, Shinghal and Batarekh (1992). When dealing with knowledge in decision table format, as in Table 4.4, if we remove a column, row or value without affecting the hypotheses derived then it can be removed. This concept can also be employed to calculate the significance of attributes. If the attribute is dispensable then that attribute is not significant. When dealing with a set of rules one would expect that all attributes would contribute to at least one outcome so the measure determined is one of comparative significance. If discarding one attribute changes the classifications of many objects then it is more significant than another

[27] Much of Section 4.7 is taken from the paper "Richards, D ., Gambetta, W. and Compton, C (1996) Using Rough Set Theory to Verify Production Rules and Support Reuse *Proceedings of the Verification, Validation and Refinement of KBS Workshop, PRICAI'96* 26-30 August 1996, Cairns, Australia, Griffith University, 57-66".

attribute which only alters a few objects when it is dropped. In rough set terminology, which was introduced in Section 4.1.2, the objects move from the positive to the boundary region if the attribute is dropped.

The issue of ambivalence is addressed by computation of the degree of dependency κ which is simply the number of cases in the positive region (card POSp(Q)) divided by the number of all cases (card(U)) (see formula 4.4). The dependency coefficient says that the set of attributes Q, the conclusion, depends in degree κ on the set of attributes P, the conditions, for the decision algorithm PQ. If $\kappa = 1$ the knowledge base is consistent. If the degree of dependency is greater than zero but less than one we can determine what cases are outside of the positive region and investigate why they are inconsistent.

$$\kappa = \text{card } POS_P(Q) \,/\, \text{card }(U) \qquad (4.4)$$

note: card = cardinal number of, POS = positive region, U = universe of objects.

Deficiency is seen as the situation where no hypothesis can be inferred. The quality of classification can be tested in RST with the classification coefficient (formula 4.5):

$$\gamma_R(X) = \text{card}(\underline{R}X \cup \underline{R}\text{-}X) \,/\, \text{card}(U) \qquad (4.5)$$

This formula takes the number of objects in the union of the lower approximation of a set and its complement and divides this by the number of all objects in the domain. This is checking whether all objects that are consistent can be classified. If we get a coefficient of 1 it means that all objects are consistent and can be classified.

Rough set theory was applied to the work by Richards(1994) who adapted an existing consultation-style ES so that it could be used in a "what-if" decision support system style. Richards found that determining the significance of a variable was important in allowing reuse of the knowledge. In that study, the AusVit[28] rule-base for the disease Botrytis was examined and decision tables for the final conclusion s*pray need* and the intermediate conclusion, *disease risk*, were created. For simplicity and to save space, the discussion that follows will be restricted to the *spray need* outcome, Table 4.7 shows the original 18 rules.

[28]AusVit is the Australian Viticulture ES that assists grape growers to manage various pests and diseases. The system won the Decision Support for Farmers Award at the 1996 Sydney Royal Easter Show.

Rule#	Stage	Dis Risk	Spr Active	Grape Use	Summer Wthr	Spr Need
1	6-14	not nil	not	EW/LW	---	high
2	10-14	not nil	not	ET/LT	---	high
3	16-17	---	not	not dried	not warm & dry	high
4	18-20	not nil	not	not dried	not warm & dry	high
5	18-21	high	not	LW/ET/LT	warm & dry	high
6	6-9	not nil	not	ET/LT/Dried	---	low
7	10-14	not nil	not	dried	---	low
8	16-17	---	not	---	warm & dry	low
9	16-17	---	not	dried	not warm & dry	low
10	18-20	not nil	not	dried	not warm & dry	low
11	18-20	high	not	EW/dried	warm & dry	low
12	18-20	low	not	---	warm & dry	low
13	1-5	---	---	---	---	nil
14	22-24	---	---	---	---	nil
15	6-14	nil	not	---	---	nil
16	6-21	---	active	---	---	nil
17	15-21	--	not	---	---	nil
18	18-20	nil	not	---	---	nil

Table 4.7: The rules for the final conclusion Spray Need from the Ausvit rule base. Each row in the table corresponds to a rule in the KB. "---" means "don't care" and "/" is an OR condition.

Convert KBS rules into rough set format (SYS)
While reducts (rules) of SYS show redundancies exist
  Determine why attribute, object or value was removed
  Fix objects
End-While
Calculate
While dependency coefficent $\kappa < 1$
  List all objects not belonging to the positive region and the object that they conflict with.
  Fix objects
End-While
While classification coefficient $\gamma < 1$
  Find objects not in the lower approximation or it's complement
  Fix objects
End-While
Output new rules (These may be converted back into the original format or used as tables).

Fix objects.
If coding error, fix.
Else If error due to missing values
    Initialise A-V check table
    Get set of objects that match with the object being investigated
    While there are more objects in the set
      read an object
      update A-V check table with a found flag for each non-missing value
    End-While
    Build one or more new objects from the original object using the AV check table to cover those values not covered by other objects.
Update SYS with new objects, modified objects and remove objects in error.

Figure 4.13: An algorithm of the verification process using rough sets. There are three main phases each contained within a while loop. Procedure Fix Objects is expanded to provide more detail. The word rule refers to the original and output rule sets. The word object refers to the cases being used by the RSL which represent the rules.

4.7.1.1 Preparation of the Rules

The rules in table 4.7 needed further conversion into an attribute-value (A-V) matrix so that it could be used by the rough set library (RSL). This meant removing not conditions and exploding the decision table to include one rule for each situation. For example, if the rule had a condition NOT(DiseaseRisk = nil) then two rules were created from the original rule one with the value high and another with the value low for that attribute. This resulted in the growth of the original decision table from 18 rules to an inflated table with 59 objects. This approach to negation is in keeping with Pawlak's (1991, p.87 axiom 3) where the property is considered to have all but the negated value. The RSL routines were used repeatedly at each of the three phases. An algorithm is presented in Figure 4.13, with more detail provided in the following sections.

4.7.1.2 Phase One: Removal of Redundancies

Through the generation of reducts, redundancy in the 59 objects created above was removed and resulted in 40 reduced rules. Surprisingly from a domain perpective, the reducts generated did not include the attribute *spray active.* This indicates that the attribute can be dropped without loss of classificatory power or introduction of inconsistency. It seemed reasonable that all attributes would be used for at least one rule when we are using rules for input. In fact one would expect the *spray active* attribute to be significant when the conclusion being pursued is the *spray need.* On further investigation it was found that the original structure of the knowledge contained overlapping values for *growth stages*. It was necessary to change the model of *growth stages* being used to remove the subsumption errors that were occurring. This resulted in 75 objects being created to represent the modified rule base. One assumes that one of the best outcomes of finding verification problems is to improve the KBS in other ways and the detection of a faulty *growth stage* model is an example of such an outcome.

4.7.1.3 Phase Two: Removal of Inconsistencies

The 75 objects output from phase one were processed again on the rough set routines. In the second phase 57 rules were generated and the *spray active* attribute now had the second highest significance coefficient. However, we now had a dependency coefficient of 0.42667. It was therefore apparent that at least one rule was inconsistent. The system identified Rule 17 to be in conflict with rules 3-5 and 8-12. Although there was only three objects representing the knowledge in rule 17 that were causing the problem, the dependency coefficient indicated that there were only 28 consistent objects out of the 75 and therefore 47 inconsistencies. A further indicator of the inconsistency of the knowledge was the large variation, from 46% to 96% depending on the inferencing strategy used, in the "number of good decisions" reported when the reduced rules were used to classify the test set (in this case the training set was used as the test set) .

The problem that emerged was the original KBS had depended on rule ordering to control the inference. The cause of the low dependency coefficient was the improper use of "don't care" values in rules that were meant to act as defaults. This is yet another example of Clancey's (1992) contention that separation of inference engine and knowledge base is a fallacy and that control knowledge is often embedded in the domain model. After fixing the objects that covered Rule 17 (using the Fix Objects procedure in Figure 4.13), the final A-V matrix had 74 objects which reduced to 57 rules and the "number of good decisions" was 100% for all inferencing methods.

4.7.1.4 Phase Three: Detection of Deficiencies

The final step was to find any deficiencies. The completeness of knowledge can be measured by the ability to classify objects. When this value is computed using the 57 reduced rules we get a classification coefficient of 1. The method for computing the classification coefficient is given in formula two in Section 2. For interest all possible cases (720) were generated and the 57 reduced rules were used to classify each object, again confirming that the knowledge was complete.

4.7.1.5 Further Research

A few issues are discussed below that were not covered in this study and require further investigation. Some suggestions are offered for each issue.

Finding circularity anomalies.

A more comprehensive approach would be to expand the rule base into a larger table which included a column for each attribute used in the rule base covering both the spray need and disease risk outcomes[29]. By doing this we would also address the problem of circularity, because all pathways would clarified.

Note that if the rule base being converted was an RDR KBS then it may have been possible to keep the number of cases generated to the same size as the rule base by using the cornerstone case stored with each rule as input into the RSL. However, cases are more specific than rules and the use of cases alone would probably result in an overfit of

---

[29] Colomb (1993) has shown that knowledge in the form of decision tables or decision trees can be viewed as propositional systems and can be converted from one representation into another. He provides algorithms for such conversions using the concept of equivalence with respect to the number of cases.

the data and higher error rates and a verification program would be needed to ensure accuracy levels were maintained. A combination of rules and cases such as described earlier in this chapter and in the work of Colomb and Chung (1995) may be appropriate.

Dependencies between attributes.

An important factor that affects the correctness of values is the existence of dependencies between attributes. The identification of functional dependencies can help us with system verification through the minimisation of redundancies, validation of new data and the inference of missing data (Keen and Rajasekar 1993). The AusVit rules did not contain dependencies between attributes. This is probably atypical of most domains. This issue is better addressed by the work using FCA discussed in Chapter Five were we are able to build a subsumption hierarchy from the rules which shows which A-V pairs tend to occur together.

4.7.1.6 Discussion of RST for Verification

It had originally been hoped that RST would be able to provide useful significance coefficients when a rule base is used as input. However, after running the rough set routines on various KBS the results were inconclusive. It is now painfully obvious that the number of times a case is seen is going to impact on the significance coefficient. Depending on how the rules are structured and the domain, some rule bases may give results that appear reasonable. It appears that if we want to compute the significance of an attribute so that the knowledge may be later understood for reuse it is necessary to keep some kind of statistics about rule usage and accuracy. Such statistics are not commonly kept. However, Edwards' (1996) *credentials* system (described in Section 3.1.3) stored statistics on the number of times a rule fires and is deemed correct or false. It is anticipated that these statistics could be used directly or in conjunction with another algorithm as a measure of importance.

While the significance coefficient was altered by the number of cases seen, the rules built were not affected by the removal of repetition since only one of each example is used in determining reducts (rules). This is not the case with most ML techniques such as C4.5 or Induct. The data were also randomised with no change in results using RST. The order in which cases are seen tends to affect ML algorithms that use some sort of covering algorithm to generate rules. The strength of RST lies in how it partitions knowledge and uses these partitions to develop rules of classification with incomplete or inconsistent data. The weaknesses of RST are its excessive memory requirements and running time.

The major contribution of this study is in demonstrating the usefulness of employing rough sets as a structured way of verifying a KB. Redundancies are identified through calculation of value reducts. Using the dependency coefficient and checking the positive region we have a means of determining when and where inconsistencies have been introduced. Using the classification coefficient we could determine the completeness or deficiency of the knowledge. The major drawback of the approach is the increase in rule base size, but size is seen as a problem due to the side-effects that are introduced when we try to maintain a large KBS. The rules that are output may be used directly in table form, which has been shown to give an order of magnitude increase in speed over propositional ES (Colomb 1990), or the rules identified as being erroneous can be modified in the original rule base. The latter approach should result in a new set of rules which is *similar* to the original set of rules and may be more comprehensible to the expert than a large decision table. A third alternative is to convert the table back into propositional form. Further work needs to be performed on larger decision tables of rules to see what problems arise with repeated conversions between representations.

4.7.1.8 Comparison with other V&V Anomaly Detection Techniques.

The approach used by RST is comparable to other verification tools. RST's Matrix A (the attribute-value table) is similar to the tables used in Expert System Checker (ESC) (Cragun and Steudel 1987) and Rule Checker Program (RCP) (Suwa, Scott and Shortliffe 1982). RCP checks for redundancy and ambivalence by comparing each rule against the others, which can be performed in n(n-1)/2 rule comparisons. Testing for deficiency using RCP requires the testing of each possible combination. The method is criticised for its high computational overheads and its limited checking features which are mainly undetected because inference chains are not examined. While RST also suffers from the high overheads associated with pair-wise checking, RST's use of the positive region in the classification coefficient is a less computationally expensive approach to determining deficiency.

The CHECK program (Perkins, Laffey and Pecora 1989) builds on the work begun by RCP and is able to detect redundant rules, conflicting rules, subsumed rules, circular rules, unreferenced attribute values, illegal attribute values/dead end goals and unreachable conclusions. CHECK creates the "if-if", "then-then" and "then-if" tables with a row and column in each table for each rule. Comparisons are made in the "if-if" and "then-then" table to see if the intersection of any rules are the "same", "superset", "subset" or "conflict". The "then-if" table in CHECK is used to detect when the antecedent of one rule is the consequent of another and in this way circular inference chains can be detected. The range of anomalies detected by CHECK is still limited and there is little empirical evidence of CHECK's usefulness. The use of "if-if" and "then-

then" tables is similar to the use of the discernability matrix, Matrix D, in RST. Matrix D performs comparisons quicker than Matrix A but the generation of Matrix D can be time consuming. For the purposes of verification of a knowledge base lengthy batch processing may not be a major problem if it is assumed that most knowledge acquisition will occur at the beginning of a system's life. Once the system was in use verification could be performed after each maintenance episode.

Search space is a problem for many verification techniques. This is particularly so if dynamic verification is performed which takes into account the inference chain, such as in COVER (Preece, Shinghal and Batarekh 1992). To reduce the search space, the generation of environments in COVER is guided by two heuristics: the first uses the existing rules to restrict the possible combinations of attributes and the second heuristic orders the environments by size so that environments that contain the ones already found are not generated. RST could employ similar techniques to reduce the large search space and processing time that is needed for pair-wise comparisons. In particular, the second heuristic bears some resemblance in RST to the use of a reduced discernability Matrix X, which subsumes all oversets and repetitions in Matrix D. Matrix X can be created for selected objects and/or attributes to find cores and reducts. The efficiency of many of the rough set algorithms have been improved but these newer versions have not yet been implemented in the RSL (Sienkiewicz 1996, personal communication).

4.7.1.7 Summary of the General Verification Technique

This section has demonstrated the utility of using RST for verification of KBS. While RST is well known in the data mining area, applying RST to KBS, rather than datasets, had not previously been performed (Sienkiewicz and Pawlak, personal communication) with the exception of Colomb and Sienkiewicz (1995). The ability of RST to remove redundant and inconsistent data and its handling of uncertain and missing values appeared to make it eminently suitable for dealing with KBS that have typically many missing values compared to cases. The case study presented using the AusVit KBS shows that the technique is applicable to KBS that can be mapped into a decision table.

While much of what RST offers is achievable by other verification tools, and some also handle inference errors without the need for conversion, RST is a technique that deserves further attention due to the simplicity of the concepts and the easy availability and customisation of the routines supplied in the Rough Set Library (RSL). Pawlak (personal communication) has expressed interest in this work and its further development. There are also many possible uses of RST that go beyond the anomalies detected in this study. For example, it is possible to pass a particular conclusion to the rough set routines and generate the rules that derive that conclusion. This could be used to test that all possible conclusions were reachable.

As noted the technique described here applies to knowledge in decision table format and the example given was a standard rule base of If-Then rules being used by Level 5 Object and included intermediate rules and conclusions. The technique was also tested on the rule bases for Chess, Tic-Tac-Toe and Garvan (which were already available in decision table format) and was found to be a reliable verification technique for the detection of redundancy, imcompleteness or inconsistencies. The approach does not particularly consider single classification RDR systems due once again to the ambiguity of the false branches and the 'inflation' problem that occurs when the false branches are expanded to handle negation. However, the approach did seem suitable for MCRDR KBS but with some refinement. In the next section some modifications to the technique described above are given to extend the approach to handle MCRDR KBS. The benefit of using this approach for MCRDR is that we can evaluate that certain groups of classifications are consistent and we can remove redundancy in RDR.

4.7.2 The Off-Line Verification of MCRDR KBS.

The previous section considered how RST could be used for the verification of conventional rule-based systems. From the lessons learnt in the compaction studies it was apparent that single classification RDR would not map easily to the decision table format required by the rough set library due to the ambiguity of the false branches. MCRDR KBS, however, do map directly to a decision table format as the KBS can be viewed as pathways of exceptions which correspond to rows in a table. It therefore seemed possible to extend the techniques described for the verification of conventional rule-bases to MCRDR KBS. An experiment on a small MCRDR knowledge base is described in this section together with a description of the modifications needed to handle multiple classifications. This extension is described in Section 4.7.2.3.

4.7.2.1 Preparation of the Rules

As already described a necessary prerequisite for applying RST was the conversion of the rule base into a decision table structure. For rulebases that use intermediate conclusions this process is achievable by exploding each rule into its a lower level form where intermediate conclusions are replaced with the primitive conditions. In an MCRDR KBS the conversion process is much simpler. Each rule represents a rule pathway which may be treated as a row in a decision table with the rule conditions along the way forming the attribute value pairs in the table. Once the rules are in the rough set format (decision table) the iterative process described below and summarized in Figure 4.13 is performed. The process is described using a small (42 rules, 11 attributes and 31 conclusions) MCRDR KBS developed for the SISYPHUS III geology domain. Thus we begin with a decision table with 42 rows and 12 columns, the twelfth column contains the conclusion which is known as the decision attribute in rough set terminology.

4.7.2.2 Phase One: Removal of Redundancies

Through the generation of reducts, redundancy in the 42 objects in our decision table was removed and resulted in 40 reduced rules. These two rules were removed either because they were originally duplicates (with conditions occurring in different sequence but giving the same conclusion) or they became duplicates after dispensable conditions had been removed. The latter situation was the reason for the rules being dropped for the SISYPHUS III KBS. While only two rules were removed we had a substantial decrease in size of the rule base in terms of the number of conditions from a mean number of 4.0 conditions per rule to a mean number of 2.2 conditions per rule. The reduced KB was then evaluated against the cornerstone cases to ensure that each case gave the same set of original conclusions (remember we are dealing with multiple classification RDR). It was found that five of the fifteen cornerstone cases gave one or two extra conclusions. By making the rules more general there were more cases covered by a rule. Assuming that our original classifications were right, it was necessary to make some rules more specific to preclude the misclassified cases by adding the A-V pair (rule condition) that had been removed. If more than one condition had been removed then by looking at the case it could be determined which condition/s needed to be added. It is initially alarming that one third of cases were overclassified, but if we take into account that the cases had an average of 6.2 conclusions each and that for a total of 93 original conclusions we obtained 99 conclusions, we achieved 94% accuracy. To achieve 100% accuracy on the cornerstone cases four rules out of the 40 needed one extra condition added.

Before going on to look at the other anomalies, it is useful to consider what benefits or drawbacks may result from this process. On the positive side, three conditions (attribute-value pairs) had been removed from the KBS and the number of concepts derived from the rules using formal concept analysis reduced from 78 to 56, signifying a 29% drop in rule-base complexity. By generalizing the rules, that is making them less specific by removing conditions, we should also develop higher concepts than we get from rules designed just to classify one example, or rock in the case of the geology domain. On the possibly negative side, this removal of conditions may result in more unseen cases being misclassified compared to the original rule-base which contained more specific rules. The strategy of keeping all cases seen, mentioned before in the removal of redundancy, and using them for evaluation before and after is seen as a way of reducing the problem of creating rules that are too general or specific.

For the purposes of making the knowledge more understandable and useful for explanation, the decision of whether to minimize the rule-base depends on our goal. If the purpose of our KBS is to provide a tutoring system that gives a complete description of a rock then removal of *dispensable* conditions is not desirable. If our goal is to perform inferencing and to understand what are the key features for classifying a rock, then minimization is appropriate. From the point of view of verification, minimization addresses the issue of redundancy but may not pursue the validation goals of the system.

The significance of an attribute.

The significance coefficient is also interesting in giving some insight into the knowledge base and can be treated as a measure of relative redundancy. Two examples are given below. The significance coefficient μ, is calculated by determining the number of objects positively classified minus the number of objects positively classified once that attribute has been dropped, divided by the number of objects in the universe of discourse. Formula 4.6 returns the significance of *attr* in the set of attributes P in the active information system.

$$\mu_{P}\{attr\} = \text{card}\ (POS_P(R) - POS_{P-\{attr\}}(R))\ /\ \text{card}\ (U) \qquad (4.6)$$

where: R={all attributes);card = cardinal number of, POS = positive region, U = universe of objects.

Significance Coefficients for rules concluding Volcanic (%VC000) or Plutonic (%PL000) are:
SignifCoef of attr 0 (GRAIN_SIZE) = 1.000000
SignifCoef of attr 3 (SILICA) = 0.600000

Sig. Coeff. for rules concluding granodiorite %GR000, granite %GR001 and microgranite %GR002:
SignifCoef of attr 0 (GRAIN_SIZE) = 0.666667
SignifCoef of attr 2 (QUARTZ) = 0.666667

Figure 4.14: Significance coefficients produced by rough set theory for two different sets of mutually exclusive conclusions.

In Figure 4.14, the first example for the conclusions %VC000 -Volcanic and %PL000 -Plutonic tells us that the attribute GRAIN_SIZE is very important in deciding on whether a rock should be classified as volcanic or plutonic, in fact all rules for these conclusions used this attribute. However, in some situations, the attribute SILICA will also affect the conclusion. This interpretation is also confirmed by the work using FCA and can be seen in the labelled line diagram in figure 5.15. The second example for the %GR family of conclusions tells us that GRAIN_SIZE and QUARTZ are equally important attributes which help us to decide if a rock should be classified as granodiorite %GR000, granite %GR001 or microgranite %GR002.

4.7.2.3 Phase Two: Removal of Inconsistencies

The 40 rules (objects) output from phase one were processed again to determine whether inconsistencies existed. Use of the dependency coefficient (formula 4.4) on all the rules in an MCRDR KBS is not appropriate because there are a number of conclusions valid for each case so that certain combinations of attribute-value pairs will result in multiple classifications, not just one, being assigned. When we determined κ for the whole SISYPHUS III KBS we got a dependency coefficient of 0 because the positive region was empty. It was therefore necessary to treat each classification individually and compute the dependency coefficient for each class to ensure that within each classification there was no inconsistency. However, it is desirable that rules which give mutually exclusive conclusions be grouped together and be verified for consistency. To achieve this in the SISYPHUS III KBS the rules for those conclusion known to be related but mutually exclusive were put into individual decision tables (files) and the dependency coefficient of each table determined. This was done for the conclusion groups: Plutonic (%PL000) and Volcanic (%VC000); Basic rock (%BS000) and acidic rock (%AC000); granodiorite %GR000, granite %GR001 and microgranite %GR002. Each group scored a dependency coefficient of 1 stating that for that type of decision the rules were consistent. A further indicator of the consistency of the knowledge is the "number of good decisions" reported after each classification. Accuracy was 100% for each individual classification and each classification group.

4.7.2.4 Phase Three: Detection of Deficiencies

When we compute the classification coefficient (formula 4.5) using the 40 SISYPHUS III reduced rules we get a classification coefficient of 1. This computation only provides a partial measure of the completeness of KBS and the discussion in 3.2.3.1 should also be taken into account.

4.7.2.5 Conclusions Regarding the Verification of MCRDR KBS using RST.

In this section we have seen that the approach to detection and correction of verification anomalies developed and described in Section 4.7 could be applied to MCRDR KBS. The only real change involved the detection of inconsistencies which required the dependency coefficient to be computed for individual or identified groups of classifications instead of the whole KBS since multiple conclusions were possible for the one case. Due to the use of cases to develop and validate rules in the RDR KA approach inconsistencies should not be possible anyway. The main benefit of the technique would be the removal of repetition and the simplification of the rules, which requires the completeness and consistency of the resultant KBS to be evaluated. It may also be desirable to sort the list of rules into an MCRDR (see Section 4.5.1).

This returns us to the main and original goal of the work reported in this chapter which was the removal of repetition. In the small example given here there was no loss of accuracy, a small reduction in the KBS size and a substantial decrease in the number of conditions per rule. The generalisation of the MCRDR knowledge and the reduction in complexity is seen to be a useful outcome. However, as mentioned, the information loss may not affect the system's ability to provide the correct classification but it is unclear whether the explanations provided by the reduced KBS will be more or less useful and in what way.

4.8 Chapter Summary

Sections 4.7.1 and 4.7.2 have described how RST can be utilised to assist the verification of conventional and MCRDR rule bases, respectively. Both sections have shown that automated attempts to perform verification generally require some sort of human intervention to determine whether the anomalies detected by the system actually require fixing and those fixes will include a mixture of automatic and manual procedures. This cooperative effort is even more noticeable in Section 7.3.1 on critiquing where an approach to computer-assisted validation is described. In the approach the computer becomes an assistant in helping the user determine where errors in the knowledge may exist.

Sections 4.1 to 4.4 were concerned specifically with the issue of removing repetition from single classification RDR KBS using RST. In Sections 4.1 to 4.4 the experiments performed by Chellen (1995) using Induct were also discussed and compared to the RST results. RST appeared better suited to the task of removing repetition but the usefulness of the compacted KBS for explanation has not been answered. The compaction process involved the reorganisation of the KBS. It is assumed that the repetition due to local patching is removed through the reorganisation process but the method is not based on the detection of repetition and the removal of identified repeated knowledge. In this way it is unclear to what extent the compaction removes repetition but since the size of a KBS is affected by the amount of repetition, a reduction in size is seen to be a reduction in repetition. Section 4.5 offered a proposal for the removal of repetition from MCRDR KBS and describes a sorting list algorithm for MCRDRs.

As stated previously, the original and primary interest was in how we can understand what knowledge we have so that it can be reused. Rough sets have been useful in analysing an existing knowledge base and improving the consistency of that knowledge which will be a better starting point when the knowledge is later put to alternative uses. It had been hoped that RST would provide the links between concepts in the knowledge that were seen as important in facilitating multiple uses of the same knowledge for different activities. It became apparent however that RST does not provide the necessary structure or abstractions. As an alternative solution, the abstraction hierarchy that FCA develops seemed a plausible answer to this problem which led to the work described in the next chapter.

Chapter 5

5 Building Conceptual Hierarchies from MCRDR[1] KBS

Section 1.2 provided a situated view of knowledge which was used to support the argument that models change and are difficult to capture. The preferred solution was to capture simple models based on an experts behaviour that only involved the capture of primitive notions using combinations of attribute-value pairs. This primitive knowledge supported the reflexive modes of KA, maintenance and inferencing in RDR. However as pointed out in Section 3.4.2 the lack of higher-level modelling features in RDR is a limitation due to the benefits that abstraction hierarchies can provide particularly for explanation and instruction. It was desirable to perform reflective reasoning on our reflexive, behavioural knowledge. Shastri notes that:

> *"when semantic networks are realised as massively parallel networks, they may provide an appropriate framework for modelling reflexive reasoning - reasoning that can be performed rapidly, effortlessly, and without conscious effort" (Shastri 1991, p.3)*

The use of a semantic network representation appeared to offer the type of functionality that was necessary. The line diagram or concept lattice developed by FCA is a semantic network that shows terminological subsumption and thus provides the hierarchical structure and the relationships between concepts that were required. Subsumption is defined by Wood as: "a formal relationship between pairs of conceptual descriptions that allow them to be ordered into a taxonomy on the basis of generality" (Woods 1991, p. 67). Woods defines five types of subsumption: extensional, structural, recorded, axiomatic and deduced subsumption. The concept matrix supports the first four types of subsumption. Deduced subsumption involves use of something like a general-purpose deduction component in conjunction with the subsumption hierarchy and is not a feature of MCRDR/FCA. The RDR KBS provided a classification system but not a subsumption system. Woods (1991) describes classification as an operation on a system of concepts whereas subsumption is an operation on a pair of concepts. By combining MCRDR and FCA classification and subsumption can be supported.

[1] Note the approach developed in this chapter is only applicable to MCRDR KBS because its exception structure only contains true branches which may readily be converted into a flat structure of rule pathways that maps directly into the crosstables used by FCA. The false branches in single classification are ambiguous and more difficult to convert into a flat structure.

In addition to abstraction, context is a major factor which will affect the appropriateness of the expert's model, particularly for reuse. The constantly changing nature of contexts (Rosenfield 1988) means that it is necessary to provide "continuity" or applicability of a context to a new situation by consideration of the relations involved (Lave 1988, p.20). Agre (1993) also identifies relationships between categories as a critical issue in the use of categories for locating things. When applied to a KBS these categories can be groups of rules or conclusions. Each rule can be seen as a low level concept that specifies the conditions under which a conclusion applies. It was the goal of this study to use the primitive concepts in the form of individual rules to find the higher level concepts and the relationships between them from which a conceptual hierarchy could be developed. To this end, this chapter describes how rules captured into an assertional MCRDR KBS can be used to develop a terminological KBS. The mechanism for this transformation is formal concept analysis which is discussed next. The screen dumps shown in this chapter, and most of this thesis, are from my implementation, called MCRDR/FCA, which is an enhancement of the current MCRDR for Windows system.

5.1 Formal Concept Analysis[2]

Formal Concept Analysis, first developed by Wille (1982), is a mathematically based method of finding, ordering and displaying formal concepts (Wille 1992). FCA is:

> *"based on the philosophical understanding of a concept as a unit of thought consisting of two parts: the extension and intension (comprehension); the extension covers all objects (entities) belonging to the concept while the intension comprises all attributes (or properties) valid for all those objects" (Wille 1992, p. 493).*

In keeping with this, a concept in FCA is comprised of a set of objects and the set of attributes associated with those objects. The set of objects forms the extension of the concept while the set of attributes forms the intension of the concept. Knowledge is seen as applying in a context and can be formally defined as a crosstable as in Figure 5.1 below. The rows are objects and the columns are attributes. An X indicates that a particular object has the corresponding attribute. This crosstable is used to find formal concepts. The following description of FCA follows Wille (1982).

A formal context ($\mathbb{K}$) has a set of objects G (for *Gegenstande* in German) and set of attributes M (for *Merkmale* in German) which are linked by a binary relation I which indicates that the object g (from the set G) has the attribute m (from the set M) and is

[2] Much of section 5.1 follows the paper "Knowledge Acquisition First, Modelling Later, Enric Plaza and Richard Benjamins (Eds), *Knowledge Acquisition, Modeling and Management*. 10th European Workshop, EKAW'97, Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 237-252."

defined as: $\mathbb{K}$ = (G,M,I). Thus in figure 5.1 we have the context $\mathbb{K}$ of animals with G = {bird, reptile, amphibian, mammal and fish} and M = {has feathers, suckles young, warm-blooded, cold-blooded, breeds in water, breeds on land, has scales}. The crosses show where the relation I exists, thus I = {(bird, has feathers), (bird, warm-blooded),…,(fish, has scales)}.

	Has feathers	suckles young	warm-blooded	cold-blooded	breeds in water	breeds on land	has scales
Bird	X		X			X	
Reptile				X		X	X
Amphibian				X	X		
Mammal		X	X			X	
Fish				X	X		X

Figure 5.1: Formal Context for "Vertebrates of the Animal Kingdom"

A formal concept is a pair (X,Y) where X is the *extent*, the set of objects, and Y is the *intent*, the set of attributes, for the concept. The derivation operators:

$$X \subseteq G: \ X \mapsto X' := \{m \in M \mid gIm \text{ for all } g \in X\} \tag{5.1}$$
$$Y \subseteq M: \ Y \mapsto Y' := \{g \in G \mid gIm \text{ for all } m \in Y\} \tag{5.2}$$

are used to construct all formal concepts of a formal context, by finding the pairs (X″,X′) for all X ⊆ G or (Y′,Y″) for all Y ⊆ M. We can obtain all extents X′ by determining all row-intents {g}′ with g ∈ G and then finding all their intersections (formula 5.3). Alternatively Y′ can be obtained by determining all column-extents {m}′ with m ∈ M and then finding all their intersection (formula 5.4) . This is specified as:

$$X' = \bigcap_{g \in X} \{g\}' \tag{5.3}$$
$$Y' = \bigcap_{m \in Y} \{m\}' \tag{5.4}$$

Less formally, we take the set of objects, G, to form the initial extent X which also represents our largest concept. We then process each attribute sequentially in the set M, finding the intersections of the extent for that attribute with all previous extents. Once the extents have been found for all attributes, the intents X′ for each extent X may be found by taking the intersection of the intents for each object within the set. Thereby we determine all formal concepts of the context $\mathbb{K}$ by finding the pairs (X,X′). Figures 5.2 (a) and (b) demonstrate the steps performed in finding the formal concepts from the formal context in Figure 5.1. Where the intersection of two extents results in a duplicate it is not repeated. Figure 5.2 (a) shows the first pass of picking up the concept's extent. Figure 5.2 (b) is the second pass where the attributes which have the set of objects in the extent are added to the original intent. The end result is shown in Figure 5.3 which shows the eleven formal concepts that were derived for the formal context in Figure 5.1

by finding the intersection of sets of attributes for each object and then finding the sets of objects that have those intents. Note that the concepts have been ordered so that they do not appear in the same order as in Figure 5.2 (b). Note that the 11th concept in figure 5.2 (b) has added the concept that contains all attributes which has resulted in the set of objects being empty. This is done to ensure that we form a complete matrix with concepts 1 and 11 being the topmost and bottommost concepts, respectively.

To present a visualisation of our ordered set of concepts as a line diagram it is necessary to compute the predecessors and successors of each concept. Predecessors are found by finding the largest subconcept of the intents for each concept. Successors are found by finding the smallest superconcept of the intents. A superconcept is a set that has all of the members of another set and additional members. A subconcept is a set that has fewer members than another set but all the members it has are contained in the other set. We only concentrate on finding sub or super concepts of the intents or extents because they are inversely related and using either set will give the same result. It is interesting to note that the larger the intent the smaller the extent and visa versa. This means that the more attributes we use to describe an object reduces the number of objects that can be described in that manner. Conversely, the more objects in the extension the less attributes can be found in the intension which describe those objects. This approach is similar to the process of *realisation* in KL-ONE (Woods and Schmolze 1991) and the predecessor is like Woods (1991) most specific subsumer (MSS) and the successor is like the most general subsumee (MGS) which also require a concept first to be classified and then the MSS and MSG concepts to be determined.

Figure 5.4 shows the list of successors and predecessors for the formal context in Figure 5.1. In MCRDR/FCA the successor list was used to identify concepts higher in the diagram, the parents, and the predecessor list identified concepts lower in the diagram, the children. This notion of parents and children is consistent with the view of an MCRDR KBS being comprised of parent and child nodes. The number of levels of parents and children are used to layout the line diagram and the algorithm from Richards and Compton (1997b) is given in Figure 5.5. However, just as users have different views of their knowledge, there is not one fixed way of drawing line diagrams and often a number of different layouts should be used (Wille 1992). The user may also reposition any of the nodes to their satisfaction providing a node is not moved higher than any of its parents or lower than any of its children.

Step	Intent	Extent
1		{1,2,3,4,5)
2	a	{1}
3	b	{4}
4	c	{1,4}
5	d	{2,3,5}
6	e	{3,5}
7	f	{1,2,4}
8	g	{2,5} {5} {2}

5(a)

Step	Intent	Extent
1	{ }	{1,2,3,4,5)
2	{a}	{1}
3	{b}	{4}
4	{c,f}	{1,4}
5	{d}	{2,3,5}
6	{d,e}	{3,5}
7	{f} {d,f,g}	{1,2,4} {2}
8	{d,g} {d,e,g}	{2,5} {5}
9	{a,b,c,d,e,f,g}	{ }

5(b)

Figure 5.2 The process of finding formal concepts from the formal context.

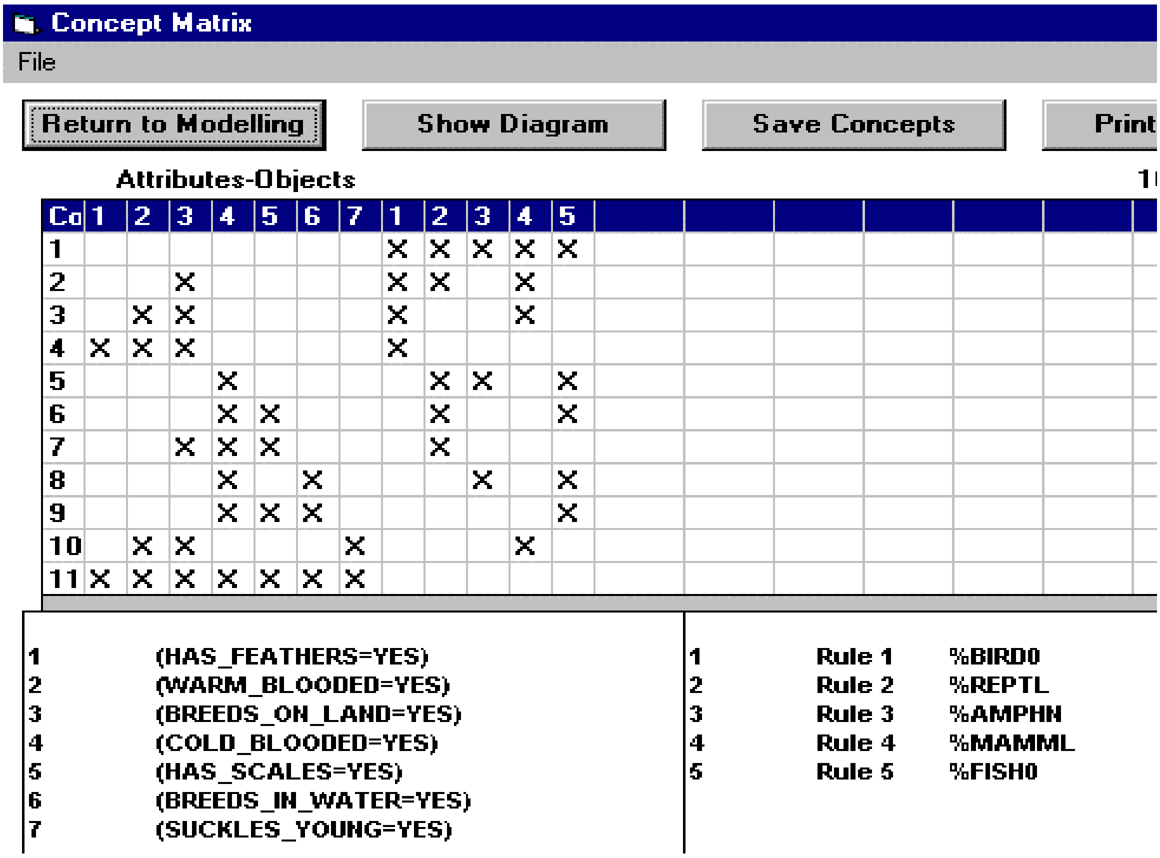


Figure 5.3: The concept matrix screen from MCRDR/FCA.

Eleven (11) concepts have been found. Each row represents a concept. The columns show the attributes, which are listed first, followed by the objects. As was shown in the formal context in Figure 5.1, there are seven attributes and five objects, here labelled 1-7 and 1-5 respectively. The labels have been converted to numbers to allow the relationships between concepts and the possible patterns (particularly useful for comparison of models) to be more readily seen. Full labelling can be obtained by using the pop-up windows as shown in this figure or by clicking on the attribute, object or concept number. The concepts have been ordered to show the subsumption relations that exist. The top and bottom concept, concepts 1 and 11, show the concepts which includes all objects and all attributes, respectively.

More formally, we use the subsumption relation ≤ on the set of all concepts formed such that $(X_1,Y_1) \leq (X_2,Y_2)$ iff $X_1 \subseteq X_2$. For a family (Xi,Yi) of formal concepts of $\mathbb{K}$ the greatest subconcept, the join (5.5), and the smallest superconcept, the meet (5.6), are respectively given by:

$$\bigvee_{i\in I}(X_i,B_i) := ((\bigcup_{i\in I} A_i)'', \bigcap_{i\in I} B_i) \tag{5.5}$$

$$\bigwedge_{i\in I}(X_i,B_i) := (\bigcap_{i\in I} A_i, (\bigcup_{i\in I} B_i)'') \tag{5.6}$$

```
For each concept
     Find parent concepts from successor list
     Find children concepts from predecessor list
End For
Locate top concept in fixed position (top and centre)
Get x-factor - find how many branchings in the predecessor list in the second column
Get y-factor - find how many branchings in the successor list in the second column
For all children of the top concept
    position one y-factor lower than the parent  and one x-factor to the right of the previous sibling  remove
     any object labelling from the parent that is found in the child
End For
For all children of the top concept
     Add-children(child)
End For
Add the last concept in fixed position
Reduce attribute labelling

Add-children(parent)   (note the child passed to this procedure is now the parent)
Find the children (parent)
For all children
     If      child is the last concept or has already been located
     Then  process next child
     Else   find out how many parents it has
              If      only one parent
             Then  If      child is an only child (its parent has one child only)
                       Then  locate the child directly below the parent
                       Else   x-coord = (Parent x-coord - ((NoChildren of Parent * x-factor)/2)
                                           + ((number of child being processed-1)*x-factor)
                       End if
             Else   locate the child midway between left and right most parents on the x-axis
                       and one y-factor lower than the lowest parent
             End If
             If the parent contains some of the children's objects
             Then  remove
             End If
             If      coordinates used
             Then   until a free position is found change x-coord moving first left and then right by half x-
                       factor
             End If
             Flag as located
             Add_children (child)
     End If
```

Figure 5.5: The algorithm used to find the concept coordinates for the Hasse diagram

From Lattice Theory, the ordered concept set can be used to form a complete lattice, called a concept lattice and denoted B(𝕂). The concept lattice is a semantic net which provides “hierarchical conceptual clustering of the objects (via the extents) …. and a representation of all implications between the attributes (via its intents)” (Wille 1992, 497). As noted in section 3.3.1.3 there are limitations to the extensional definition of a concept and only the intensional definition of a concept is used for exploring new concepts in MCRDR/FCA. However, for the development of the concept matrix (Figure 5.3) and the concept lattice (Figure 5.6) Wille’s definition of a concept is used.

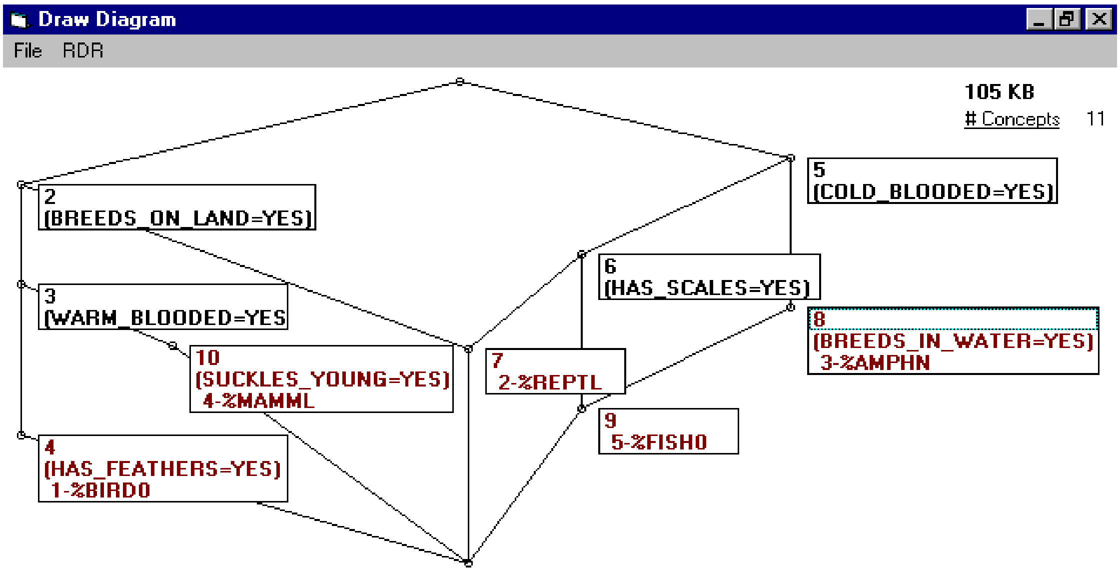


Figure 5.6: The diagram screen in MCRDR/FCA for the formal context “Vertebrates of the animal kingdom” given in Figure 5.3.

Each circle represents a concept. The attributes that belong to a concept are reached by ascending paths and objects are reached by descending paths. The rule conditions are the attributes. The objects are labelled using the rule number and the conclusion code. The conclusion codes have been made as meaningful as possible in the size limitation of five characters but a longer description could be substituted on the line diagram if desired for greater understanding.

In Figures 5.6 and 5.10 the concepts are shown as small circles and the sub/superconcept relations as lines. Each concept has various intents and extents associated with it. The labelling has been reduced for clarity. All intents of a concept β are reached by ascending paths from β and all extents are reached by descending paths from the concept β. In MCRDR\FCA it is possible to display the concept, attribute/s or object/s belonging to each node or all three dimensions can be displayed concurrently, as in Figures 5.6 and 5.10. Although showing all dimensions increases the amount of information being presented the extra information is important in understanding the diagram presented. Alternatively, the user can click on an individual node to see the concept number and all of its extents and intents. The ability to express all relationships between attributes, such as which attributes occur together, and the ability to describe each object in terms of the concepts it contains and the relationship of those concepts to others is a major strength of the lattice structure (Cole and Eklund 1996).

Clancey points out that semantic networks can embody a cognitive model that exhibits patterns of human behaviour. However, since they are limited to words they constitute a "grammatical model of cognition" (Clancey 1991b, p.251) and do not capture non-verbal conceptualisations or model the perceptual-conceptual learning that occurs when humans attach meanings and interpretations to the words. These limitations are acknowledged together with the fact that to some extent concepts are being treated as "things" rather than "processes of perceiving and processes of behaving" (Clancey 1991b, p.252). Again it is stressed that while RDR systems can exhibit behaviour similar to a human expert there is no claim that the system developed matches the way that the human mind works. A major criticism of programs that use a grammatical model is that because they are bounded by the terminology used they are unable to learn at the knowledge level. In MCRDR/FCA a grammatical model is being used to uncover concepts that may not be so easy for an expert to articulate. This occurs because in many cases an expert will perform tasks at a subconscious level and may have difficulty explaining why they have acted thus. As explained by Clancey (1988) concerning the process of extracting conceptual and procedural abstractions from MYCIN into NEOMYCIN, the most famous reuse of knowledge, "we are stating a model that goes well beyond what experts state without our help" (Clancey 1991b, p.261). Next we look at how the FCA theory described has been incorporated into MCRDR.

5.2 Combining MCRDR and FCA

As noted in Section 3.3.1, RDR and FCA share a number of views including the beliefs that knowledge applies in a context and that KA is a task that is best performed directly by experts. In both approaches KA is reduced to the task of classifying objects (cases) and the identification of the salient features. In FCA, KA begins with the elicitation of a crosstable from which the concepts derived can be used to generate implications. The implications generated are shown to the user who is asked to say whether they agree or disagree with the implication. If the user does not agree they are asked to offer a counterexample. This study starts from the opposite direction by using the rules in the MCRDR KBS as the input into a formal context. The reason for this is twofold. Firstly the purpose of using FCA was to uncover higher models in rules that had already been acquired using MCRDR. Secondly, it was felt that the RDR approach to KA was probably less demanding for experts than the development of crosstables, the analysis of the generated implications and the offering of counterexamples which is required by the FCA approach to KA .

The first step was conversion of the MCRDR KBS into a flat structure by sequentially traversing the KB for each rule picking up the conditions from the parent rule until the top node with the default rule was reached. From this flattened pathway of rules the user chose either the whole KB or a more narrow focus of attention from which to derive a formal context. When the whole KB was chosen the rules (which were identified by the rule number and conclusion code) and rule conditions formed the extents and intents, respectively. However, such a global view is only feasible for small, if not very small, KBS. As with any graphical representation, as the number of rules being modeled grew, the line diagram became too cluttered to be comprehensible. Therefore, to limit the concepts to a manageable size that could be viewed in a matrix or a line diagram the user was asked to narrow their focus of attention to a particular rule or conclusion. The decomposition of a concept lattice into smaller parts is a strategy that has previously been found useful (Wille 1989a) and is similar to the approach proposed by Ganter (1988) where the context is shortened to find subcontexts and subrelations. The technique described here is a little different in that it is not the formal context that is shortened but the rules which are put into a formal context are restricted according to the user's selection. These options are introduced below and described in more detail in Section 5.2.1.

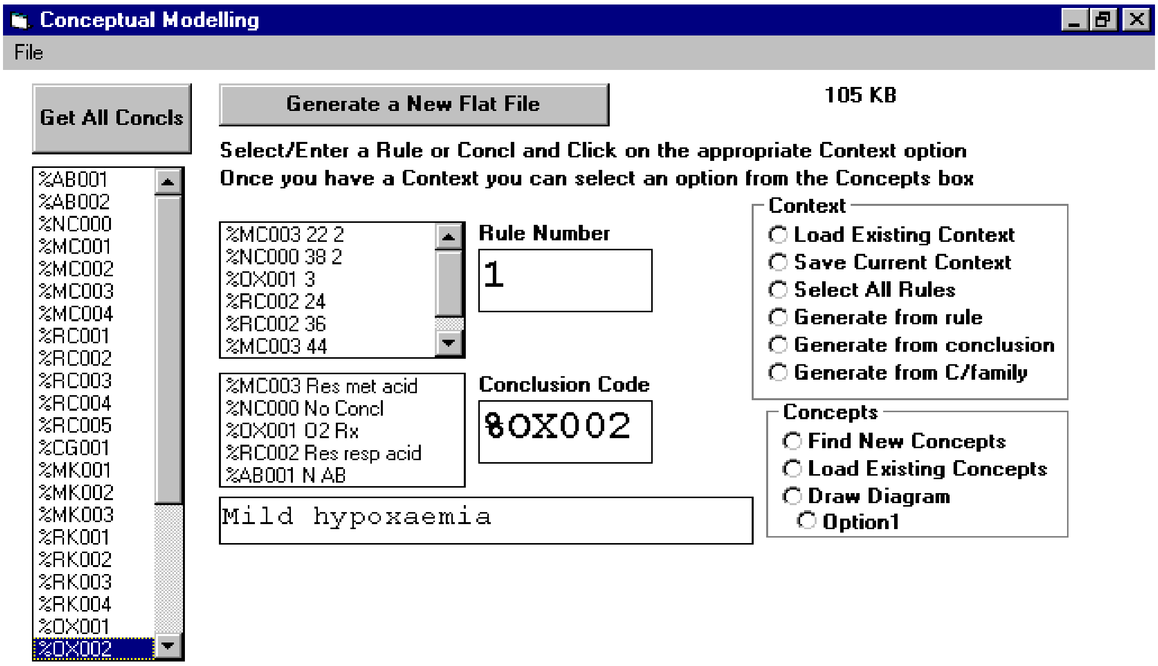


Figure 5.7: The first selection screen in MCRDR/FCA which allows the user to select rules based on a specified case, rule, conclusion code or conclusion code family. This selection will be used to build a formal context from which concepts will be generated.

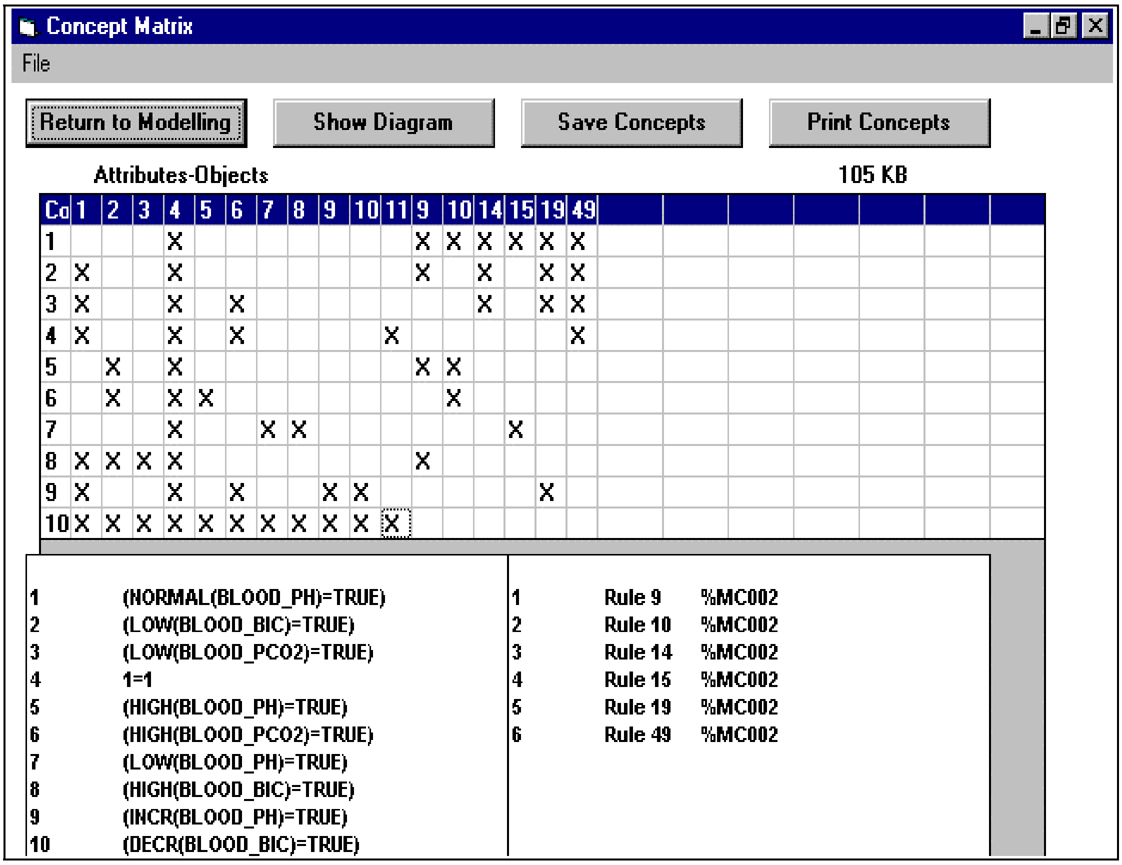
It seemed likely that a user may want to carry out modelling in connection with a particular case. As shown in Figure 5.7, the user could select a case and the rules. The conclusions associated with that case were then presented as the defaults from which to select. The user could also click the GET ALL CONCLS button to see a list of all conclusion codes from which to pick. If the “Generate from conclusion” radiobutton was clicked, all rules using the specified conclusion were selected and added as objects to the set G, forming the extension of the context. As each object was added the conditions of the rules were added to the set M of attributes to form the intension of the context, first checking to see if any attributes had already been added by previous rules. Where the relation I held, that is object g had attribute m, a cross was marked in the appropriate row and column. If the user chose a particular rule then that rule was added as the first object with the rule conditions as the initial intension. Every condition in each rule in the flattened RDR rule base was searched for a match on the initial set of attributes. If a match was found, that rule was added to the extension and all new attributes (conditions) found in the matching rule were also added to the intension. The result was a formal context $\mathbb{K}$ comprised of a set of objects G and attributes M connected by the binary relation I. An alternative selection screen, described further in Section 5.2.1 has also been developed that provides the user with views based on conditions, attributes, rules or conclusions with numerous options at various levels of abstraction to assist the user in selecting the aspects of the knowledge base of interest to them. An example based on selecting a conclusion follows.

	Normal Blood PH	Low Blood BIC	Low Blood PC02	1=l	High Blood PH	High Blood PC02	Low Blood PH	High Blood BIC	Incr Blood PH	Decr Blood BIC	Curr Blood PH≤7.36
9-%MC002	X	X	X	X							
10-%MC002		X		X	X						
14-%MC002	X			X		X					
15-%MC002				X			X	X			
19-%MC002	X			X		X			X	X	
49-%MC002	X			X		X					X

Figure 5.8: A formal context for the MCRDR rules which conclude %MC002- “Metabolic compensation.2” in the blood gases domain.

In figure 5.8 we have selected the conclusion %MC002 - *“Metabolic compensation.2”* from the blood gases domain. In Figure 5.8 the set of objects G = {9-%MC002, 10-%MC002, 14-%MC002, 15-%MC002, 19-%MC002, 49-%MC002}, where the object is referred to by the rule number and the conclusion. The set of attributes M = {Normal (Blood_Ph), Low(Blood_Bic), Low(Blood_PC02), 1=1[3], High(Blood_Ph), High(Blood_ PC02), Low(Blood_Ph), High(Blood_Bic), Incr(Blood_Ph), Decr(Blood_Bic), Curr(Blood_Ph) ≤ 7.36}. The set of relations between G and M is I = {(9-%MC002, Normal(Blood_Ph), (9-%MC002, Low(Blood_Bic), (9-%MC002, Low(Blood_ PC02), ...,(49-%MC002, Curr(Blood_Ph) ≤ 7.36) }.

[3] 1=1 is the default condition for the default rule which is inherited by all rules.



Co	1	2	3	4	5	6	7	8	9	10	11	9	10	14	15	19	49
1				X								X	X	X	X	X	X
2	X			X								X		X		X	X
3	X			X		X								X		X	X
4	X			X		X					X						X
5		X		X								X	X				
6		X		X	X								X				
7				X			X	X							X		
8	X	X	X	X								X					
9	X			X		X			X	X						X	
10	X	X	X	X	X	X	X	X	X	X	X						

Figure 5.9: The concept matrix in MCRDR/FCA for Windows for the conclusion %MC002- "Metabolic compensation.2" in the Blood Gases domain. Ten (10) concepts have been found. Each row represents a concept. The columns show the eleven attributes, which are listed first, followed by the six objects as shown in the formal context in Figure 5.8. The attribute labels have been converted to sequential numbers and the object labels correspond to the rule number to allow the relationships between concepts and the possible patterns to be more readily seen. Full labelling can be obtained by using the pop-up windows as shown in this figure or by clicking on the attribute, object or concept number. The concepts have been ordered to show the subsumption relations that exist. The extent of the top concept, No 1, includes all objects. The intent of the bottom concept, No 10, includes all attributes.

Our treatment of each rule condition, which is actually an attribute-value pair, as an attribute is similar to the technique known as *conceptual scaling* (Ganter and Wille 1989) which has been used to interpret a many-valued context into a (binary) formal context. A many-valued context, such as that represented in an MCRDR KBS, is a quadruple (G,M,W,I) where I is a ternary relation between the set of objects G, the set of attributes M and the set of attribute values W (merkmalsWerte in German). Essentially, each attribute is treated as a separate formal context with the values as attributes associated with each of the original objects. A scale is chosen, such as a nominal scale (=) or an ordinal scale (≥), to order these attributes. From the many contexts, one for each attribute, the concepts are derived.

Once the formal context has been generated from the users selection, concepts are found, ordered using the subsumption relation ≤, predecessors and successors are computed and a line diagram drawn according the process described in section 5.1. Appropriate ordering of concepts is difficult as a given concept may be a subconcept of different superconcepts. This can be seen in the concept matrix in Figure 5.9 where we can see five groupings of concepts in concepts 1-4, 5-6, 7, 8 and 9-10. In the ordering shown in Figure 5.9, concept 5 is a superconcept of concept 6. An alternate ordering could have shown concept 5 as a superconcept of concept 8. If the context is ordered before derivation of the concepts, a more aesthetic ordering of concepts may be produced but does not affect the calculation of predecessor and successors or the graph layout. The concept lattice for the conclusion %MC002 - *"Metabolic compensation.2"* is shown in Figure 5.10.

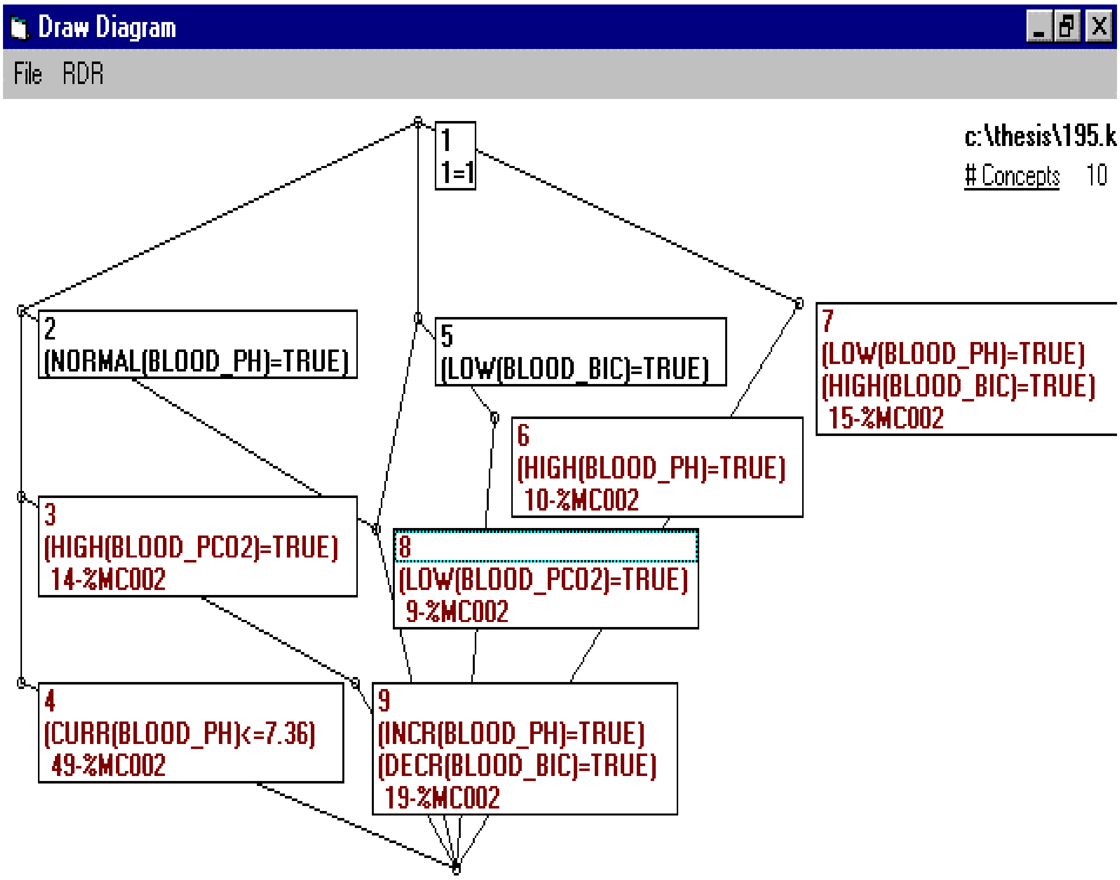


Figure 5.10: The Concept Lattice in MCRDR/FCA for the conclusion %MC002- "Metabolic compensation.2".

The example given in this section showed the results when the %MC002- "Metabolic compensation.2" conclusion was chosen. As stated, this is only one of a number of ways of selecting what rules should be included in a formal context. We look at the range of possible views in the next section.

5.2.1 The Possible Views using MCRDR/FCA

The selection screen described above and shown in Figure 5.7 was the first selection screen developed in MCRDR/FCA and only offered limited options from which the user could define a focus of attention. Restricting the aspects of the KBS to be viewed is important for keeping the information presented to the user to a manageable size. Below a total of nine different options, as shown in Figure 5.11, are described. Within some of these options there were further choices that the user could make. The purpose of each view is given. In keeping with the situated view, the idea was to present the user with the range of views that may be useful but the choice of which one to use was to be left to the user who would best know their current situation. The major problem with such an approach is that the more choices a person is given the harder it is to understand and remember the differences between choices and when they are applicable. As can be seen below in Figure 5.11 there are 11 radio buttons in the Context frame. The first two options involve file handling with the "Load Existing Context" button for loading a previously created context file and the "Save Current Context" button for writing the last context formed into the context file. The nine options that follow concern the different selection criteria that can be used in deciding which rules should be included in a formal context. We look at these options now.

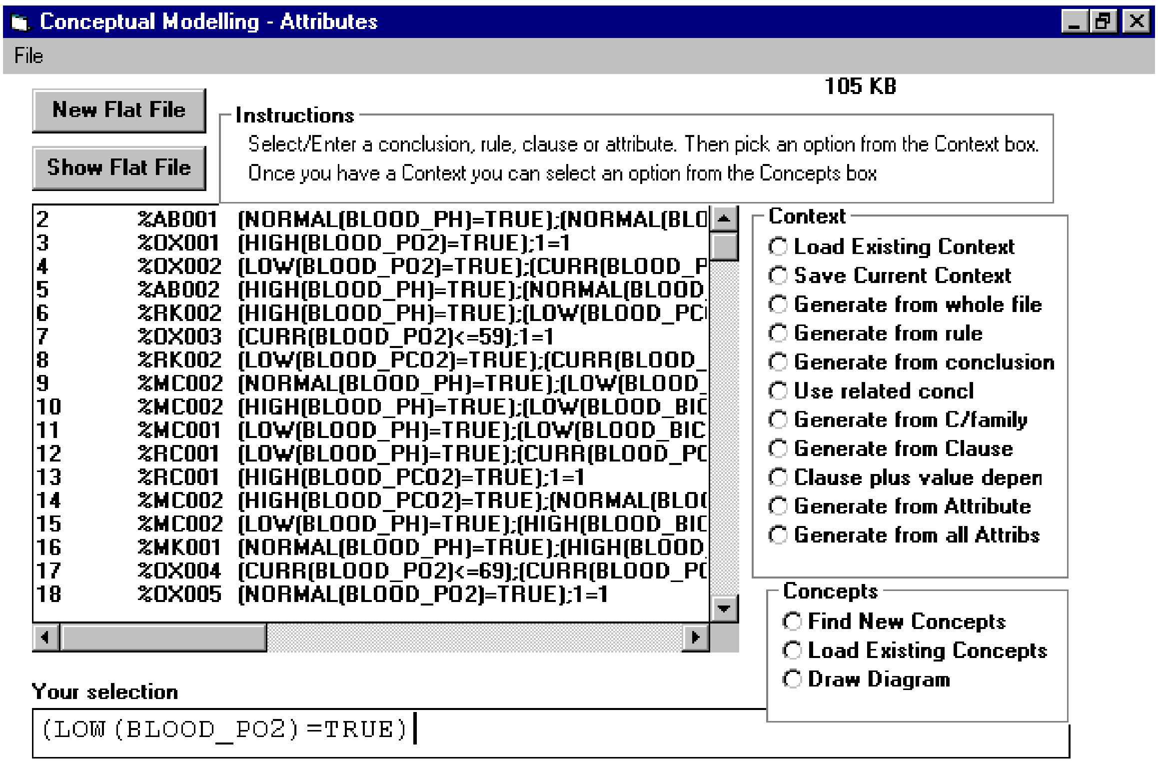


Figure 5.11: The Second Selection Screen in MCRDR/FCA.

5.2.1.1. Generate from whole file

This option uses all the rules in the flat file to generate a context. This is the simplest technique. Each rule corresponds to a row in the crosstable. The object name is a combination of the rule number and the rule's conclusion. The rule conditions form the attributes/columns.

5.2.1.2. Generate from rule

This option uses the specified rule to form the first object in the crosstable, with the conclusion being the object and the rule conditions forming the attributes. Then each record in the flat file, which is a pathway in the KB, is searched for a match on any of the specified rule's conditions. If a match is found the whole record is added. This means that any new attributes introduced by the matching rule are added as a new column in the context. An 'X' is marked for all attributes found in this rule. Only exact matches are picked up. Neighbouring values are not picked up. Neigbouring or borderline values are discussed further under option 5.2.1.7.

5.2.1.3. Generate from conclusion

Each record in the flat file is read and any rules with the specified conclusions are added to the formal context. Each condition is checked against existing conditions (attributes) in the formal context and added if it does not already exist. An 'X' is marked in the corresponding column.

5.2.1.4. Use related conclusion.

This option allows the user to generate concepts for all conclusions related to and including the specified conclusion. The notion of *related to a conclusion* is simply that if a rule shares any conditions with rules that give the specified conclusion there is some sort of relationship between them. This option is a combination of the rule and conclusion options. Thus the range of possible indicators that the two conclusions are related has been broadened. Each rule with the specified conclusion is used to generate a list of conditions. Then all rules which use any of these conditions are added to the formal context. So instead of using one rule as the basis for comparison the set of rules with the specified conclusion are used in the same manner as reported in 5.2.1.2. This will typically result in a much larger number of selected rules as each condition in every rule in the set is used as the selection criteria of other rules. This takes a different approach to using conclusion families, option 5.2.1.5, because it does not rely on the expert to identify a relationship but uses the content of the rule as the basis for a possible relationship.

This approach will include some rules whose conclusions are not really related to the selected conclusion. This overgeneralisation is a drawback but is not seen as a major problem. The aim of these selection options is to narrow the focus and remove at least some of the irrelevant concepts. It should be apparent from the diagram if a rule/conclusion is not really related. An extension to the current diagram screen would be to allow the user to drop concepts that are not of interest. Further suggested improvements are described in Section 5.2.2.

5.2.1.5. Generate from conclusion family

A conclusion family refers to a naming convention used to identify that certain conclusions are related to others. The PEIRS conventions for defining a conclusion has been used. The conclusion codes are in the form %XXNNN where X is an alphabetical character and N is numeric, such as %CL001 may stand for wear jumper and %CL002 may mean wear a t-shirt. Both conclusions are related to the which clothes (CL) should be worn. Other rules that concern which clothes to wear should use the same %CL prefix with the next sequential number allocated at the end. Generating a context from a conclusion family is very similar to option 5.2.1.3 that uses the whole conclusion, except only the first three characters are compared. If another conclusion naming convention was used then the current code would need to be modified, but as the system is a prototype a better way that lets the user specify the convention is not currently planned.

5.2.1.6. Generate from clause

In this option the user is able to specify/select a condition which is used for comparison with the attributes used in the rule conditons. If the condition is found then the rule is added as a new object with the rule conditions as the attributes. All conditions are checked if they are already a part of the formal context before being added, so there are no duplicates. An ‘X’ is marked in the corresponding column for that attribute/condition. The user does not need to enter the full condition but can use any string for matching. For example the attribute name (BLOOD_PC02) or even partial name (PC02) can be entered.

The user may combine conditions using AND (&) so that all joined conditions must be found to be a match. OR (|) has not been implemented because it can be achieved by rerunning the option and taking ADD to add the new context to the existing context which will pick up OR conditions.

When the user takes this option they are asked the question "Do you wish to pick up attributes associated with your selection ?". If the reply is NO only the conditions that match those in the selection criteria are added as attributes to the formal context. By replying NO and using the “&” operator only the attribute information for specified conditions is added to the formal context. This avoids adding other conditions that are not of primary interest and which may obscure the picture. Such an approach would be used when the expert knows which attributes/conditions they are interested in.

5.2.1.7. Clause plus value dependency

This is similar to the option above but instead of only picking up exact matches, conditions that use the same attribute as the one in the specified condition are evaluated against an attribute file to see if the specifed value has neighbouring values. For example, the condition specified may be TEMP=HOT. If the possible values for TEMP are HOT, WARM and COLD then WARM can be seen to be a bordering value. All rules that used TEMP=HOT or TEMP=WARM are added to the formal context. This is a kind of fuzzy treatment of values and is aimed at overcoming differences in perception between individuals and over time.

Two possible approaches to handling borderline values when developing formal contexts have been considered. The first is to use a border category, such as BORDER-HIGH-MEDIUM, so that if the specified value is HIGH and a rule is found with this value, an ‘X’ is marked in the HIGH and BORDER-HIGH-MEDIUM attribute column for that rule. If another rule uses the value MEDIUM it would be picked up as a borderline value and an ‘X’ marked in the MEDIUM and BORDER-HIGH-MEDIUM attribute column for the second rule. This will result in both rules intersecting on the BORDER-HIGH-MEDIUM attribute showing a relationship between the two rules that may otherwise have not been revealed in the line diagram. The other approach is to find the value/s next to the value in question, which could be higher and/or lower values, and mark an ‘X’ in the value specified and those values adjacent to it. This would result in the two rules above having an ‘X’ marked in the both the attribute-value HIGH and MEDIUM columns. But this approach does not show the actual value assigned to the rule condition and does not show which value is the borderline value.

The first option using boundary categories has been implemented. In Figure 5.12 the attribute file for the SISYPHUS III geology domain is shown. This file has been updated to include bordering value categories and the last field holds the code “D” which indicates the attribute has dependant values. Currently, to maintain the attribute file the user goes into the attribute screen shown in Figure 5.13 and specifies which values are related to others. It is envisaged that this task would be done at system setup time when the user is specifes the relevant functions and appropriate preprocessing of the data.

```
Attribute       Strings                                     Dependency Code
GRAIN_SIZE.     COARSE BORD_COARSE_MED MEDIUM BORD-MED-FINE FINE    D
COLOUR          GREEN   BORD_GREEN_GREY   GREY   BORD_GREY_L_GREEN_WHITE
                L_GREEN_WHITE  D
QUARTZ          OVERSATURATED          BORD_OVERSAT_SAT          SATURATED
                BORD_SAT_UNDERSAT  UNDERSATURATED       D
SILICA          VERY_HIGH          BORD_VERY_HIGH_INT          INTERMEDIATE
                BORD_INT_LOWISH LOWISH BORD_LOWISH_V_LITTLE V_LITTLE
                   D
OLIVINE         ALWAYS POSSIBLY NEVER
PYROXENE        RICH POOR
FELDSPAR        GT2_3PLAGIOCLASE             BORD_GT2_3PLAG_1_3TO2_3PLAG
                1_3TO2_3PLAGIOCLASE GT2_3ORTHOCLASE ABSENT    D
DARK_MINERALS   GT90  BORD_GT90_56TO90  56TO90  BORD  56TO90_30TO56  30TO56
                BORD_30TO56_LT30 LT30       D
CALCIUM         HIGH BORD_HIGH_MED MEDIUM BORD_MED_LOW LOW    D
POTASSIUM       HIGH BORD_HIGH_MED MEDIUM BORD_MED_LOW LOW  NO      D
IRON            RICH BORD_RICH_INT INTERMEDIATE BORD_INT_LOW LOW    D
```

Figure 5.12: The Attribute file for the SISYPHUS III Geology Domain. Bordering value have been included in attributes identified as containing values that may overlap.

This option also lets the user just focus on the selected attributes but does not cater for the & or | operators as in option 5.2.1.6. To handle an OR situation the user can repeatedly use this option for as many conditions as they are interested in, taking the ADD option when asked if they want to create a new context or add to an existing context. The ability to select using conjunctions of attribute using bordering values is left to future work. Further discussion on bordering values and their treatment in the nearest neighbour algorithm was given in Section 3.3.2.1.

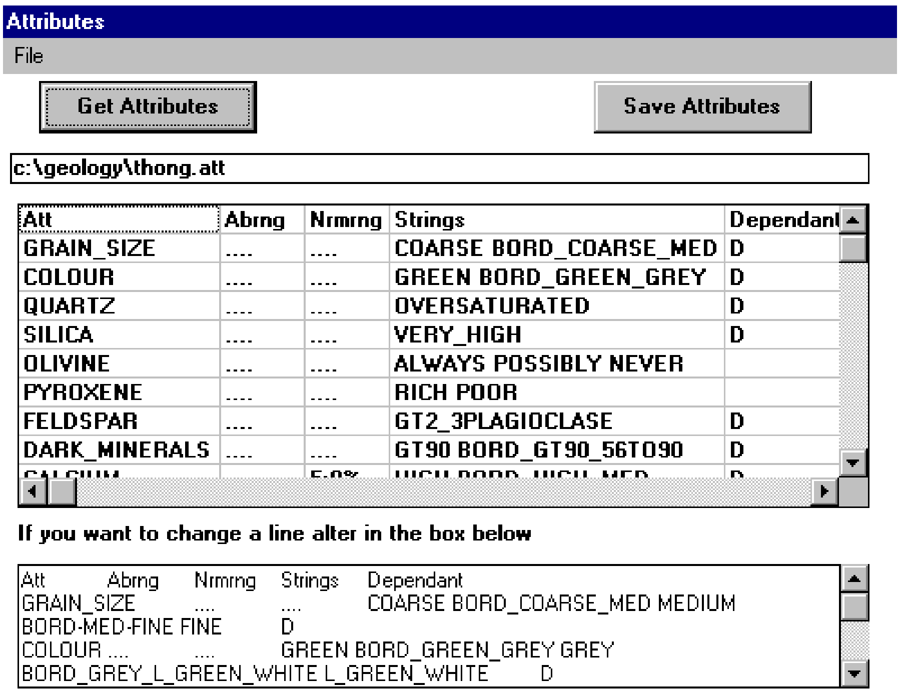


Figure 5.13: The Attributes screen in MCRDR/FCA where the user can define bordering value categories for any attributes.

5.2.1.8. Generate from attribute

This option is appropriate when the user wants to generate concepts for a specified attribute without regard for the value and also display the attribute without the value. Option 5.2.1.6 allows the user to select by attribute alone but when the concept is displayed the whole rule condition is shown in the diagram. Disregarding the value in the condition both in finding other similar objects as well as in the labelling results in abstraction of the concepts to a higher level.

5.2.1.9. Generate from all attributes

This option is similar to the previous option 5.2.1.8 but the user does not have to specify which attribute to abstract as all attributes are abstracted. This option saves repeating every attribute and substantially reduces the number of columns in the crosstable to cover a domain. This option is seen as a good first insight into a KBS at a higher level to determine what other areas may be of interest. Figure 5.14 shows the line diagram produced when this option was taken for the blood gases KBS. The diagram shows which rules use which attributes and gives some idea of relationships between attributes and which ones are likely to occur together.

5.2.1.10 Discussion of the possible views in MCRDR/FCA

The options offered above for selecting which rules to include in a formal context may be combined. It is envisaged that different users and applications will have particular requirements which may only require a subset of the options offered here and may require the addition of options not considered in this list. The purpose of these options was to:

- minimise the number of concepts generated to improve the comprehensibility of the line diagram,
- let the user focus on what is of greatest interest without extraneous concepts to cloud the result,
- provide sufficient choices to give the user control and flexibility to suit a wide range of situations.

While, the views of the knowledge provided through this selection screen meets these criteria it is not claimed that these options are optimal or comprehensive. A key issue with the options for selecting views described above is how to present them to the user so that they know when and how they can use them. As more options are made available the decision when to use them and how to combine them becomes difficult. It is suggested that the particular options made available, which may include options not explored in this thesis, should be tailored to the needs of the users for each particular application. The options that allow selection by rule, conclusion or conclusion family should be standard inclusions but others will depend on the nature of the domain and the needs of the user. We go on to look at the shortcomings of MCRDR/FCA and possible enhancements.

5.2.2 Limitations of MCRDR/FCA and Proposed Improvements

The system described in this chapter is a prototype that has been used as a proof-of-concept that knowledge in an MCRDR KBS could be reused as the basis of developing an abstraction hierarchy to support activities that require greater understanding of the concepts and their structure than was provided in the RDR assertional KBS. The system thus does not offer the full range of features that may be wanted in a deployed system. There is no apology for this as it is strongly believed that any system that solves a significant problem will need tailoring to the environment in which it will be used. This is seen as one of the benefits of RDR since the KBS is user-developed and tailoring of the system by the KE to different domains and applications is relatively simple using the Visual Basic or Hypercard software. There are, however, a number of enhancements that would improve the usability and usefulness of MCRDR/FCA regardless of the application or domain. Of course, the particular application or domain will affect how important it is to include these enhancements. We look now at some features that would improve the MCRDR/FCA system.

There are many changes which can be made to the user interface. Many of these changes relate to the limitations of the amount of information that can be displayed comprehensibly at one time. Important enhancements include the ability to drop nodes from the lattice or to zoom in and out using selections made via the concept lattice to determine what new contexts and concept lattices should be developed.

The limitations of visual representations and the need to capture and show related values were major driving factors in the development of TOSCANA (Vogt and Wille 1995). In TOSCANA a scale for each attribute is developed which is a simple line diagram for that attribute showing the relationships between the possible values. When more than one attribute is considered at a time then the line diagrams become nested according to the attribute sequence selected so that at a node for one attribute-value combination a smaller line diagram for the next level of attribute is shown within that node. TOSCANA is a commercial product and a number of large organisations are making use of, or are at least interested in, this tool. However, the number of levels of nesting possible are obviously once again limited by the amount of space on the screen and there is also the problem that the order of attributes will affect the interpretation of the line diagram.

The major problems with extension of the MCRDR/FCA tool to support zooming and the handling of multiple contexts concurrently are the limitations of Visual Basic and the amount of processing required. Visual Basic has been satisfactory for the development of a prototype but many restrictions on the size of such things as strings and grids (tables) has meant that KBS over a few hundred rules can not be handled without most of the code being rewritten in another language such as C. Essentially all processing needs to be performed externally and Visual Basic used simply for the interface, just as the RDR Engine part of MCRDR/FCA was written in C by Phil Preston.

An additional problem which goes beyond the limitations of Visual Basic is the fan out problem that occurs as the size of the knowledge base grows. As described in Section 5.1, concepts are formed by taking the intersection of sets of attributes and the set of objects which share those attributes. Each time a new row in the formal context is processed the new concept is intersected with every existing concept and results in the exponential growth of the set of concepts. In the worst case this is an NP-Hard problem that can not be remedied with the addition of more computer resources as computer resources can only assist in linear time problems. As is common with NP-hards problems in AI some sort of pruning of the solution space is offered in this thesis by restricting the size of the contexts through the various selections offered in Section 5.2.1.

In Section 7.3.1 it is shown how a proposed rule can be critiqued against existing pathways in the KBS. A nice and not too hard extension to this would be the ability to view this proposed rule as part of a concept lattice. This would also be a form of 'what-if' analysis that allows the user to explore what the outcome of the rule will be on the existing model before they make the commitment to add the rule.

Section 5.3.2 and the next chapter describe the use of MCRDR/FCA for the comparison of different KBS. Comparison of the concept matrices can be assisted by ordering the rule conditions (attributes in the formal context). Then some form of visual pattern recognition, manual or automatic, could be performed to produce subsets of the concept matrices showing matches and differences.

Extensions have been made to MCRDR/FCA which allow the user to develop a number (currently up to four) concept lattices and then to be shown a concept lattice of the similarities and differences between the four models. The user may also load the four KBS so that they can view the case associated with a particular rule to be used as a counterexample in possible discussions between owners of the different KBS. This feature was preliminary work to the RE work described in Chapter Six which combines the different KBS at the beginning into one flat file and then allows different views of the combined KBS to be taken.

In Section 5.2.1.7 a description was given on how boundary-related values could be handled. This issue is important because relevant rules may not be included in the formal context developed if the right selection criteria is not given. It is also important because without consideration of boundary-related values the actual relationships between rule conditions and conclusions will not be evident in the diagram. The line diagram should show that two rule conditions that refer to the same attribute one with the value HIGH and the other with the value MEDIUM are closer than if the value for that attribute in the second rule condition was LOW. The solution described in Section 5.2.1.7 has been implemented but not explored or evaluated in great depth and could possibly be improved by the technique known as conceptual scaling where different operators are used for comparison. In MCRDR/FCA the = operator is used when determining the intersection of sets but it should be possible to implement a range of operators such as >= or <= which can be combined. This would allow more precision in the specification of what type of boundary-values are to be included. For example, the user may be interested in all rules where BLOOD-PH = HIGH and BLOOD_BIC = NORMAL. To ensure cases with borderline values for these attributes are selected they could specify the borderline value for BLOOD_PH which would pick up in addition any conditions where BLOOD_PH=NORMAL and only choose to pick up borderline values greater than the specified value for BLOOD_BIC so that conditions with BLOOD_BIC=HIGH would be included but not BLOOD_BIC=LOW. When we consider the nature of the Blood Gases domain where each rule condition is made up of a function, attribute, operator and value and the large number of possible variations for each, particularly the value, it may be essential to include conditions that satisfy a number of criteria.

Despite the limitations and need for further development beyond a prototype, MCRDR/FCA has been applied in a number of domains where it has been used to gain some understanding of the KBS that was not as easy through other methods such as rule traces or browsing the KBS. Four Domains are described. The first three involve classification problems and the fourth is a configuration problem. As is noted in the description of the case studies which follow the classification domains provided a much richer source of information than the configuration domain where there was virtually no abstractions to be found. Since it is early days for MCRDR for configuration it is not clear whether this result is due to the nature of the domain or the way in which it has been developed. Some conjectures have been made in Section 8.2.2.4 regarding the results in this domain.

5.3 Case Studies using MCRDR/FCA

The MCRDR/FCA system has been used in a number of domains including pathology, agriculture, geology and chemistry. Each of these are discussed briefly below with a demonstration of a different aspect of how the tool can be used.

5.3.1 Blood Gases Domain

The first domain that was looked at with MCRDR/FCA was a 60-rule Blood Gases KBS known as 105, that had been developed from the cornerstone cases associated with the 2000+ PEIRS rules. The line diagram in Figure 5.14 has been generated from a context based on attributes only without regard for the values, option 11 on the second selection screen in Figure 5.11 and described in Section 5.2.1.9. This diagram is interesting as an abstraction of the knowledge base. It can be seen that conclusions from the same conclusion family are generally grouped together. For example, the attribute BLOOD_BIC is used by all conclusions for the %MC conclusion family and BLOOD_P02 is only used by the %OX conclusion family. These groupings of conclusions into families are to be expected since the expert has already identified that there is a relationship between them. There were eight different conclusion families: AB, CG, MC, MK, NC, OX, RC and RK. As pointed out by the expert in the evaluation study performed in Section 8.2.2.1 but not known at the time by myself, the use of the letter "K" as the second letter in the code meant that BLOOD_PH=HIGH. In Figure 5:14 it can be seen that all the conclusions using "K" use the attribute BLOOD-PH but this abstracted view does not give us the information regarding the value. To find this information it is necessary to zoom further into the diagram taking into consideration all concepts that use the attributes BLOOD_PH, BLOOD_BIC and BLOOD_PC02. The view shown in Figure 5:14 was one of the first drawn and was used as a proof of concept that the models developed using FCA were consistent with the model the expert was expressing in the KBS assertions.

The blood gases KBS was used as a test domain to see how useful MCRDR/FCA could be for learning the key concepts in the domain. The results of these evaluations which involved the expert that developed the 105 KBS are described in Section 8.2.2.1.

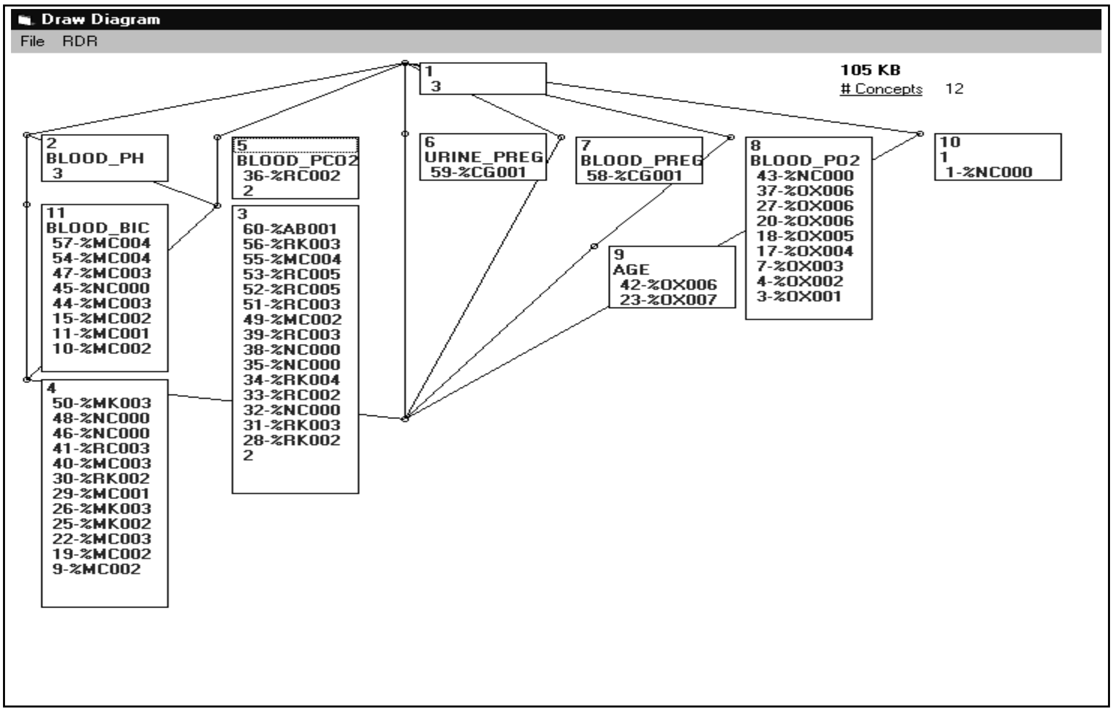


Figure 5.14: The line diagram in MCRDR/FCA for Windows using a formal context based on the attributes in the Blood Gases KBS without regard for the values of those attributes.

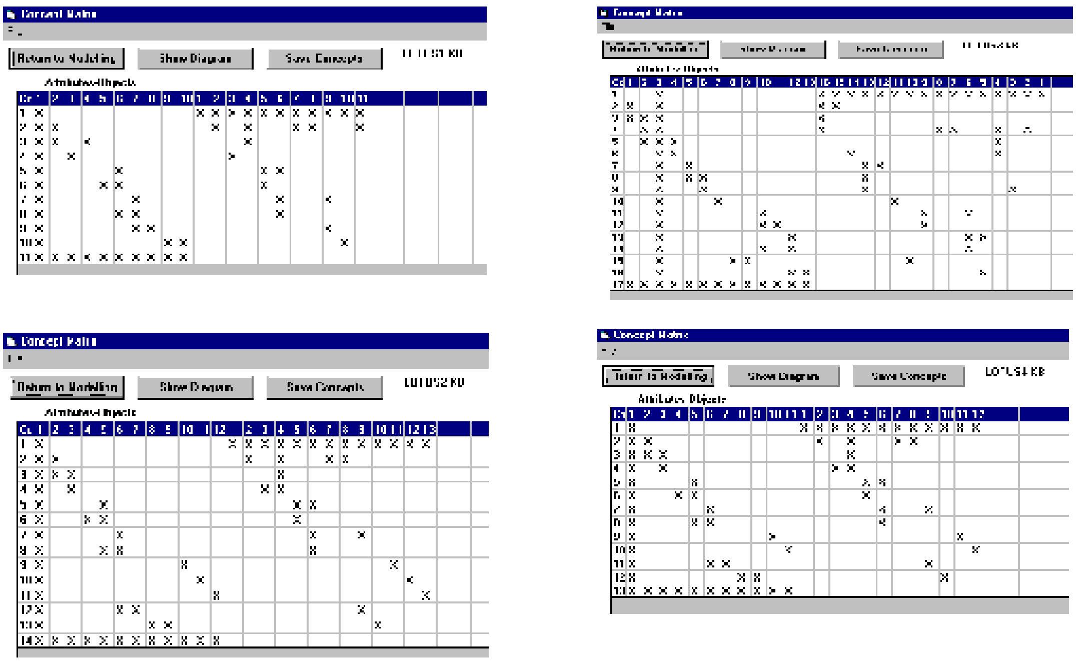


Figure 5.15: The concept matrices for the four LOTUS KBS in MCRDR/FCA

5.3.2 Lotus Domain

The second domain was known as LOTUS and concerned the adaptation and management of the *Lotus Uliginosis cv* Grasslands Maku for pastures in the Australian state of New South Wales (Hochman et al 1996). The knowledge was recorded into four

KBS by four independent agricultural advisors. These four advisors were not experts regarding the Lotus crop as knowledge in this domain was emerging and the main purpose of developing multiple KBS was to build up a corpus of knowledge for this new domain. The concept matrices and line diagrams in Figures 5.15 and 5.16 were used to compare the conceptual models of the advisors. This technique was seen as a useful way to identify the main concepts and reconcile any differences within this domain.

The concept matrices for the four LOTUS KBS are shown in Figure 5.15. It is not expected that you can properly read these four matrices since they have all been reduced to fit into the one figure but the purpose of Figure 5.15 is to show the visual similarity between the four KBS. Through the use of labelling as shown in Figures 5.3 and 5.9 it is possible to make a detailed comparison between the four models using the concept matrix. From such comparisons it can be seen that all KBS share a number of concepts, (the first nine attributes and ten objects are the same in each KBS). It can also be seen that the fifth concept in the Lotus3 KB is not shared by any of the others and that the advisor considers that when the LOTUS_RATE >=3 the conclusion of *Ryegrass* no longer holds and it should be replaced by *No Conclusion*. Concepts 12 and 13 in the Lotus3 KB represent new concepts that are not shared by any of the other KBs. The concept intent (SCARABS=YES) and the concept extent (%SCARA), which represent a rule condition and rule conclusion respectively, have been introduced by this advisor. The conclusion %SCARA is an abbreviation for “Maku OK, but persistence limited by scarabs”. By looking at the matrix the experts are able to see not only what attributes (intents) and conclusions (extents) others consider important but also the relationship between them and how it affects other conclusions.

The line diagrams for Lotus 1, 2, 3 and 4 are shown in figure 5.16. The line diagram provides a more hierarchical understanding of the sub and super relationships in the domain than the concept matrix. At a glance we can see that Lotus1 and 2 have 11 and 14 concepts, respectively, and that the three concepts that are different are concepts number 9, 10 and 11 in the Lotus2 KBS. These concepts have introduced new attributes and conclusions (objects) not used by the Lotus1 KBS. The structure of the knowledge in both KBS is very similar with four levels of concepts in both. Even though concepts 2, 3 and 4 in both KBS appear to be slightly different structurally they embody the same ideas. Due to inheritance of attributes on higher paths both advisors consider that when (LOW_PH=YES) and (RYEGRASS>=15) the conclusion should be %NC000 No Conclusion. Without enumerating all similarities and differences between all KBS, we can see that the concept matrix, and even more so the line diagram, provide succinct but powerful tools for analysing conceptual models. Comparison is facilitated by using the same attribute ordering in all formal contexts.

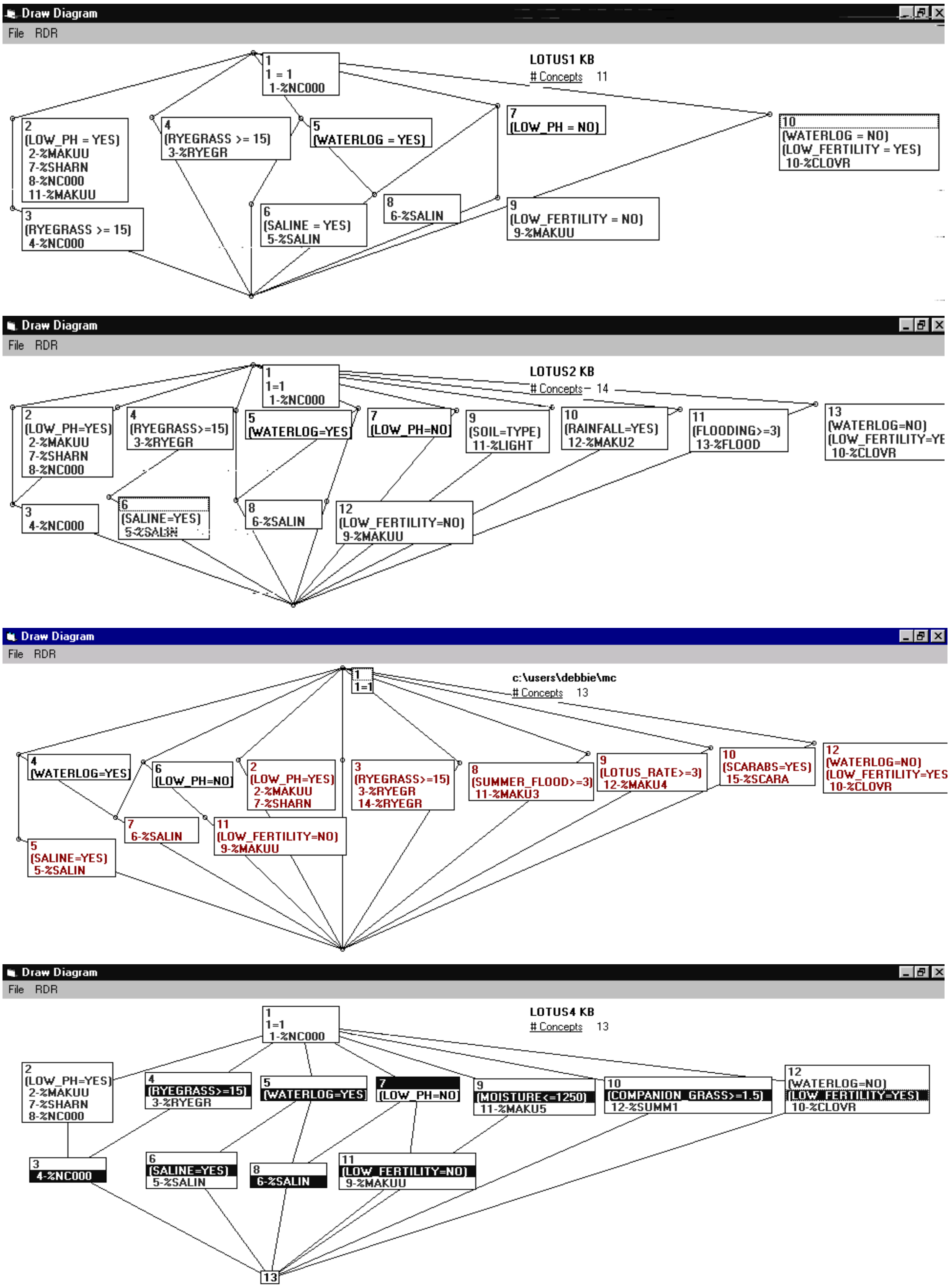


Figure 5.16: The line diagrams for Lotus 1, 2, 3 and 4 KBS in MCRDR/FCA

The examples shown used all the rules in the LOTUS KBS since the number of rules ranged from 11-18. As described earlier the focus of attention could have been narrowed by using the various options on the selection screen to reduce the number of objects to be included in the formal context. The recommended approach is to view the whole KB to identify variations and then reduce the context by selecting the rules or conclusions that differed.

An extension to the approach is given in Chapter Six where the reconciliation of expertise in multiple KBS from multiple sources is treated as a requirements engineering task and was spawned from the introductory work described in this section. In the next section we look at the third and final classification domain considered.

5.3.3 SISYPHUS III Geology Domain

The third domain was the SISYPHUS III (Shadbolt 1996) geology domain which also involved knowledge from multiple experts. Initially the tool has been used to understand the key concepts of the domain. We can see in Figure 5.17 that when GRAIN-SIZE = COURSE a rock is plutonic (%PL000) and if GRAIN-SIZE = FINE a rock is VOLCANIC (%VC000). However, when GRAIN-SIZE = MEDIUM then if SILICA = LOWISH it is a volcanic rock otherwise if SILICA = VERY-HIGH or INTERMEDIATE it is a plutonic rock. The line diagram has shown us what attribute-value pairs are the critical ones for these conclusions.

In addition to using the concept lattice and matrix for analysis of the rule base, the concept hierarchy derived using FCA is being used to assist the user in deciding whether and where a new concept/proposed-rule fits in with the existing concepts. This can be done by entering a new rule as usual in the MAKE screen in Figure 3.3 and then taking the EVALUATE RULE command button or by entering the pathway on the Test Pathways screen in Figure 3.13. This feature is described under critiquing in Section 7.3.1.

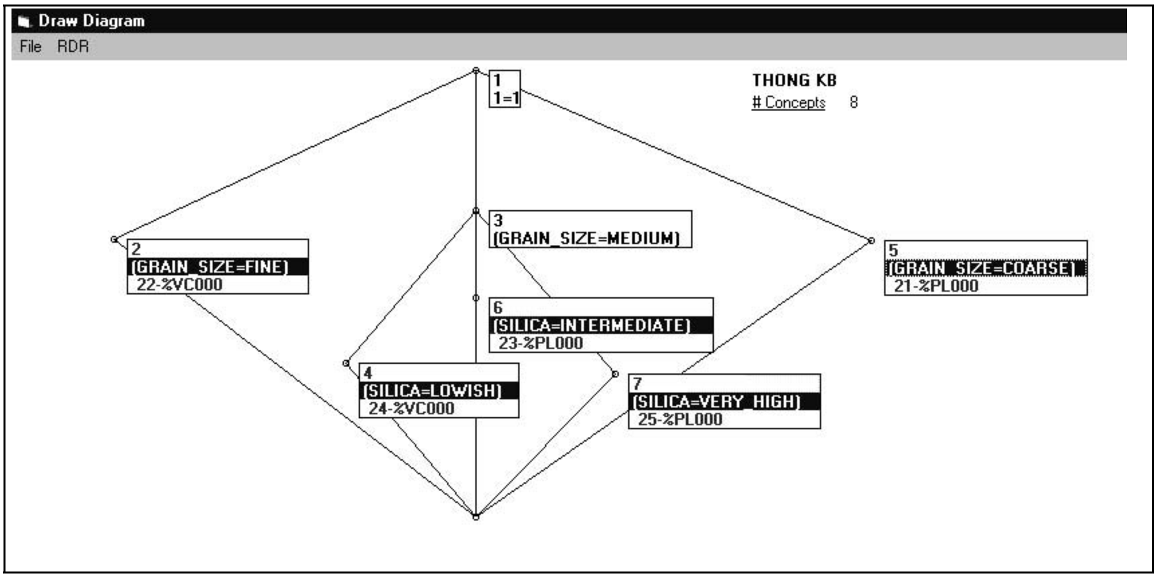


Figure 5.17: The line diagram for the SISPHYUS III domain from the context for the conclusion %PL000 - Plutonic and %VC000 - Volcanic.

This section has made use of the final combined MCRDR KBS developed as part of the RDR contribution to the Sisyphus III experiment. Interestingly one of the other submissions to Sisyphus III used FCA to build line diagrams from the card sorts (Erdmann 1998) but in the approach there was no strategy for handling conflict (which was referred to as *noise*), the next step after developing diagrams was not clear and no KBS were developed. This domain is discussed in more detail in Chapter 6 where the various sources of expertise provided in the KA material were used to test out the RE framework developed. We look next at a configuration task.

5.3.4 The Ion Chromatography Domain - a configuration task

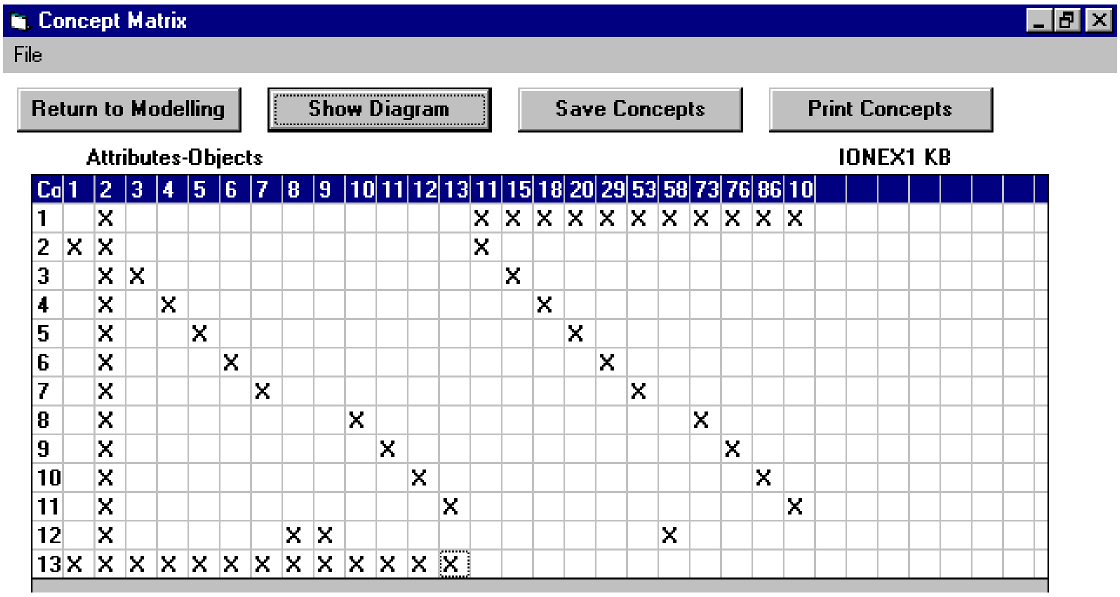
A description of the domain is taken from Ramadan et al (1997b)

"Ion exclusion is a well known ion-exchange chromatographic technique for separating strong acids as a class from weak acids using a high-capacity sulfonated ion exchange resin. The ICE mechanism of solute retention is based on the phenomenon that neutral molecules penetrate the resin while the counter ions with respect to the exchange ion are repulsed or, in other words, excluded from it. Therefore by this mechanism acidic compounds can be separated on a cation exchange resin and basic compounds on an anion exchange resin. The aim of the system is to define appropriate conditions for the separation of desired groups of acids or bases".

The ion chromatography domain is complex with 23 attributes with 2 to 10 values for each attribute. The configuration task itself is also more complex than the task of classification because in addition to parameters (attributes) and values we also need to take into consideration constraints, requirements, preferences and global cost function. For a configuration to be a solution each parameter must have a value, no constraints can be violated and all appropriate requirements must to be satisfied. Inferencing with a configuration tasks is not as simple as with classification as the inference cycle must be repeated a number of times until a solution can be found. This is because the outputs of one cycle will be the inputs of the next cycle until a solution is found (Ramadan et al 1997a).

```
1     1=1~    2-%DEA08~  3-%POS01~  4-%MOA03~  5-%COL01~  6-%MOB01~  7-
      %DEA08~ 8-%MOA05~ 9-%MOA01~ 10-%COLP1~ 11-%SUP01~ 12-%DEA01~ 13-
      %POS02~  14-%DEA05~  15-%SUP02~  16-%DEA03~  17-%POS02~  18-%SUP02~
      19-%DEA02~   20-%SUP02~   21-%PRE02~   22-%POS02~   23-%COL01~   24-
      %COL03~ 25-%COL05
2     (FORMCOM=MOLYBD)~1=1~    2-%DEA08~ 3-%POS01~ 4-%MOA03~
3     (FORMCOM=MOLYBD)~1=1~(DETECTOR=DIR_SPEC)~  3-%POS01~
4     1=1~(DETECTOR=DIR_SPEC)~       3-%POS01~ 100-%SUP02~ 101-%ELB04~
      106-%MOA02~ 108-%POS01~ 109-%COL01~ 113-%MOA02~
5     1=1~(DETECTOR=DIR_SPEC)~(MOBILEPHASE=H2SO4)~     109-%COL01~
6     1=1~(MOBILEPHASE=HNO3)~  6-%MOB01~
7     1=1~(ANY(FORMCOM)=MANNIT)~     7-%DEA08~
8     1=1~(MANNIT(FORMCOM)=TRUE)~    8-%MOA05~ 110-%DEA08~
9
            1=1~(MANNIT(FORMCOM)=TRUE)~(YES(CONDCTANCE)=TRUE)~(MOLYBD(
      FORMCOM)=TRUE)~    110-%DEA08~
10    1=1~(YES(CONDCTANCE)=TRUE)~     9-%MOA01~  57-%DEA01~  80-%DEA06~
      110-%DEA08~
```

Figure 5.18: Partial listing of the concepts found for the Ion Chromatography KBS. The concept number is shown first, then the set of attributes (rule conditions) separated by "~". The set of objects (rule number and conclusion) separated by "~" are shown last.



Co	1	2	3	4	5	6	7	8	9	10	11	12	13	11	15	18	20	29	53	58	73	76	86	10
1		X												X	X	X	X	X	X	X	X	X	X	X
2	X	X												X										
3		X	X												X									
4		X		X												X								
5		X			X												X							
6		X				X												X						
7		X					X												X					
8		X								X											X			
9		X									X											X		
10		X										X											X	
11		X											X											X
12		X						X	X											X				
13	X	X	X	X	X	X	X	X	X	X	X	X	X											

Figure 5:19: The concept matrix for the conclusion family %SUP in the Ion Chromatography KBS.

No changes were necessary to MCRDR/FCA to allow the 154 rule Ion Chromatography KBS to be modelled using the line diagrams since the structure of the knowledge is no different to any other MCRDR KBS. The only difference is the addition of an inference cycle. However, it appears that theories such as rough sets and FCA are more aimed at classification type tasks and the taxonomy FCA develops in the form of a line diagram is in fact a classification tool. This may account for why building explanations of the ion chromatography domain using the line diagrams were not easy. Another more obvious reason for the poor results is that many of the rules only had one condition and thus the number of objects that share an intersection of attributes is often one as shown in the partial listing of the concepts determined for the whole domain in Figure 5.18. We can see this in the example in Figure 5.19 where I chose to look at all conclusions that start with %SUP. Note that attribute number 2 is the default condition 1=1 and is shared by all rules.

To try to find some interesting concepts or higher level abstractions that could be explored further, I had developed a general technique for analysing a KBS that involved generating all concepts for the whole KBS. This full set of concepts was generally too large to look at on the screen so I had the list printed, like the partial list in Figure 5.18. From the printout I would try to find concepts which had a large number of attributes shared by a large number of concepts. Unfortunately in this domain there were virtually no concepts that met that criteria. An exception is shown in Figure 5.20 where some intersecting concepts could be found which could represent abstractions.

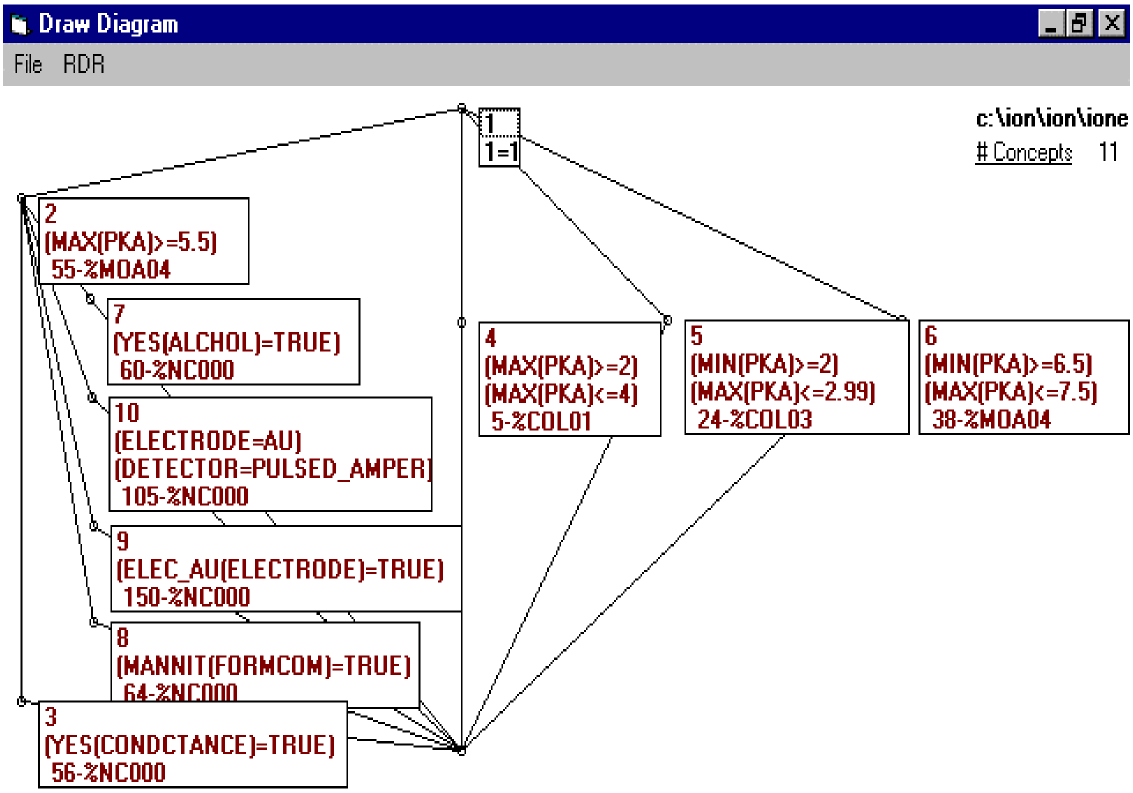


Figure 5:20: The line diagram for the formal context using the rules which have the partial condition MAX(PKA) in the ion chromatography domain.

In Figure 5.20 we can see that the conclusion %MOA04-“MOBILEPHASE = H_2O” should be given if MIN(PKA) >= 6.5 and MAX(PKA) <= 7.5 (concept 6) or MAX(PKA) >=5.5 (concept 2) except if ALCOHOL is TRUE (concept 7) or
ELECTRODE = AU and DETECTOR = PULSED_AMPER (concept 10) or
ELECT_AU(ELECTRODE) is TRUE (concept 9) or
MANNIT(FORMCOM) is TRUE (concept 8) or
CONDUCTANCE is TRUE (concept 3).
As an aside it appears that the condition ELECTRODE = AU is the same as ELECT_AU(ELECTRODE) is TRUE.
MAX(PKA) is also used to determine if column 1 - %COL01 or column 3 - %COL03 should be selected.

On the whole the results from the ion chromatography domain were disappointing and it was difficult to learn anything interesting about the domain.

5.3.5 Summary of the Analyses from Each Domain

The line diagrams from each of the four domains show that MCRDR/FCA can be used on a range of domains. The blood gases KBS was the first domain considered and a number of interesting views of the knowledge were possible. The agricultural and geological KBS also provided sources of multiple expertise and prompted further investigation of the tool for reconciliation of different viewpoints. The combined SISYPHUS III KBS has also been used in the verification studies considered in Section 4.7.2. Generally speaking the MCRDR/FCA tool seems to be a useful method of gaining greater understanding of a domain. Based on the use of MCRDR/FCA in these domains, it appears that classification tasks use more structured knowledge than configuration tasks. At this stage it is hard to know whether the tendency to build rules with one condition was a feature of the expert used in the ion chromatography domain or whether this was a feature of knowledge and KA in that domain. It is also not known whether this domain is representative of configuration tasks in general. The answer to these questions will involve more configuration tasks to be solved using MCRDR and further comparison. However, some reasons why this may have happened are given in Section 8.2.2.4. To assist the reader in determining the usefulness, particularly for instruction, of the various models built in each of these domains using MCRDR/FCA each domain has been analysed using the line diagrams and that analysis is evaluated by domain experts in Section 8.2.2.

5.4 Chapter Summary

This chapter has described how an assertional KBS in the form of an RDR KBS may be interpreted as a formal context which is used by FCA to derive a terminological KBS in the form of an abstraction hierarchy. This finding is significant as it makes possible a whole new way of viewing the development life cycle of KBS and addresses some of the problems associated with developing complex models and domain ontologies.

RDR has been shown to be a successful way of capturing assertions but has not offered a higher level view of a domain. Early investigations of the type of knowledge needed to support multiple uses showed that assertional knowledge alone was not enough but that the abstractions, relationships and structure provided by terminological knowledge was also needed. The concept lattice of FCA provides these features and in a way that allows the existing RDR knowledge to be used without alteration. By retrospectively uncovering the higher models we avoid the bottleneck problems associated with mainstream approaches to building such models plus we have a more reliable means of validating the knowledge we are capturing since the assertional knowledge is based on real cases and observed expert behaviour.

The different selection criteria that could be used to specify which primitive concepts to include in a formal context also provide a powerful knowledge base browsing tool and the ability to see different views of the knowledge. These views can involve a particular area of the domain knowledge which includes all related concepts or can be a view of the knowledge at various levels of abstraction.

The ability to show different views is extended in the next chapter to cover viewpoints from different parties which has enabled the activity of requirements engineering to be added to the range of activities that RDR can support. In Chapter Seven we review how the incorporation of FCA into RDR has facilitated a number of other activities such as critiquing and tutoring. In Chapter Eight the FCA line diagram is evaluated along a number of dimensions. The work reported in the current chapter is the major contribution of this thesis.

ERATA: Please insert at page 153.

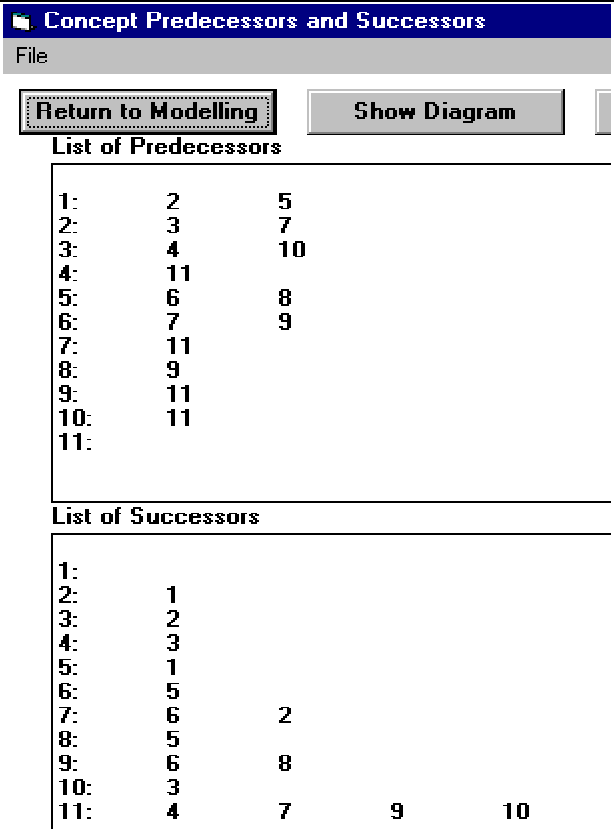


Figure 5.4: The MCRDR/FCA screen which shows the list of predecessors and successors for concepts generated for the vertebrate domain shown in figure 5.1.

Chapter 6

6 Requirements Engineering[1]

The work reported in the previous chapter opened up a new range of activities beyond those that were originally envisaged in this thesis. The concept matrices and lattices developed by MCRDR/FCA can be viewed as conceptual models that were implicitly represented in the simple RDR KBS comprised of conclusions and A-V pairs. By comparing conceptual models we are able to handle multiple sources of expertise and the types of activities that this involves. The work described in Section 5.3.2, which compared the models developed for the four Lotus MCRDR KBS, was the starting point for the work described in this chapter. However, this chapter is concerned with more than the development of multiple models that can then be manually compared. Capturing and using knowledge from multiple sources changes the nature of the questions users may ask and requires a different type of *activity-reuse* compared with the type of activities needed in an individual KBS. A major factor when handling multiple sources of expertise is how to handle the inevitable differences between the individual viewpoints that each KBS represents.

This chapter is concerned with the detection, identification and management of conflict between viewpoints to provide a combined KBS that reflects each individual viewpoint. The management of conflict is essentially a requirements engineering task and thus the framework and enhancement of the MCRDR/FCA tool described here extends the knowledge engineering task to an RE task.

This chapter is an example of increased interest in synergy between KA and RE research. RE is becoming recognised as an important concern in KA research. This chapter provides a comprehensive approach to RE that is implemented using FCA and MCRDR but the framework offered is applicable beyond MCRDR and is applicable to any KR that can be converted to a decision/cross table.

There is much similarity between the needs of knowledge engineering using multiple sources of expertise and those of requirements engineering (RE) from multiple perspectives. To a large extent in KE, the knowledge engineer decides which viewpoint

---

[1] With the exception of section 6.3.3, this chapter follows closely the paper "Richards, D. and Menzies, T. (1998) Extending the SISYPHUS III Experiment from a Knowledge Engineering to a Requirements Engineering Task *11th Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:SIS-6".

to adopt. The RE approach, by way of contrast, is to negotiate a resolution between the owners of the different perspectives. However, is requirements engineering (RE) a complicated addition to current knowledge engineering? Will the process of rationalising multiple conflicting viewpoints slow down the production of an expert system? If the costs of RE are prohibitive, then RE will rarely be applied.

The RE strategy described in this chapter is a simple extension to the knowledge engineering techniques described in Chapter Five. Given assertions in rulebases (the A-boxes) from different stakeholders, a concept hierarchy (the T-box) is generated and critiqued. Conflicts recognised in the T-box can be used to drive negotiation strategies amongst the different stakeholders. A general framework for this approach, that applies beyond MCRDR KBS, is described together with an implementation using FCA and RDR. This work addresses the situation where we are able to reduce requirements to a set of A-V pairs and thus represents a subset of the total available RE approaches.

This chapter is organised as follows. Section 6.1 introduces RE. Section 6.2 describes the RE framework and an implementation is found in Section 6.3. A case study is presented in Section 6.4 using seven knowledge bases built from the data for the SISYPHUS III (Shadbolt 1996) project. The evaluation technique is described in Section 6.3.6 and is used in Section 6.4.6 to show how the resolution operators have halved the initial degree of conflict. Related work, future work, further discussion and chapter summary are presented in Sections 6.5, 6.6, 6.7 and 6.8 respectively.

6.1 Requirements Engineering: A Review

Requirements engineering (RE) can be defined as "the elicitation and formulation of requirements to produce a specification " Easterbrook (1991, p.8). Current requirements engineering focuses on the maintenance of multiple concurrent viewpoints from different stakeholders (e.g. Easterbrook or Finklestein et al 1994). Here, a subtle re-expression of RE is offered: "RE is the application of negotiation operators to reduce the differences between stake holder viewpoints". The advantage of this re-expression is that it can be mapped into an evaluation program for RE, which are discussed more in 6.3.6 and demonstrated in 6.4.6. While RE also involves other issues such as how to gather initial specifications and formalisation of the end specification the RE issue addressed in this chapter is one of the most difficult issues.

There are a number of reasons why it is important to capture these multiple viewpoints rather than taking the approach that the different viewpoints must be captured into one specification (Easterbrook 1991, Ramesh and Dhar 1992). Tracking multiple perspectives is needed because:

1. Specification errors are often the cause of a poor choice between alternatives during the specification phase. By tracking multiple perspectives a history of the design rationale is provided so that when modifications are necessary they can be made more quickly and based on the background that formulated them in the first place.
2. Many people are involved in projects requiring information to be passed between groups and phases. Individuals will forget and may change over time, subgroups have different roles and viewpoints.
3. Design changes can be replayed and retraced. This has implications for reuse of specifications because if it is understood in what circumstances a certain path should be taken then these steps can be reapplied.
4. A more representative specification can be developed and a better framework for conflict resolution can be provided. The specification acts as both a contract and a communication channel.
5. Ownership is an important issue and by allowing multiple perspectives owned by the originator of that perspective we are more likely to motivate the user to participate in the resolution process.

The concept of a viewpoint corresponds to Finkelstein et al's (1989) formalisation of a viewpoint which includes: a style, area of concern, a specification, a work plan and a work record. This allows for an individual to hold a number of viewpoints and removes some of the problems of equating a particular viewpoint with an individual. Each owner of a viewpoint is called a stakeholder. When managing different viewpoints, conflict between different stakeholders must be handled.

6.2. The RE Framework

The viewpoint management framework has the following five steps that are repeated until all stakeholders are satisfied.

1. Requirements acquisition - Capture each viewpoint in a working knowledge based system (KBS). The KBS is an assertional knowledge base (A-box), which is called the performance system, and a terminological knowledge base (T-box), which is called the explanation system, plus a set of cases. These cases can be divided into historical cases representing true observations from the domain; and hypothetical cases representing some desired functionality. Note that historical cases cannot be doubted while hypothetical cases can possibly be ignored since an imagined case may never occur in the real world.
2. Requirements integration - convert all KBS into a common format.

3. Concept generation - In this phase we add to the T-box for each individual KBS.
4. Concept comparison and conflict detection - Compare the T-boxes of each KBS and detect conflicts.
5. Negotiation - Employ a resolution strategy based on the type of conflict detected in Phase Four. Output of this phase is fed back into Phase One where modifications are performed.

6.3. An Implementation

The general framework in Section 6.2 has not committed to any particular implementation choices. I continue now looking at this framework but within the context of an instantiation. As a result a number of restrictions on the generality of the implementation are imposed and are described in the following subsections.

6.3.1 Phase One: Requirements Acquisition using a Knowledge-Based Approach

Easterbrook (1991) argues that specification can be viewed as a knowledge acquisition process and that many KA tools can be adapted for requirements engineering. One such system, Requirements Apprentice (RA), has been developed by Reubenstein and Waters (1989) which uses a KBS to provide analysis of conclusions and inconsistencies and can also be used for documentation. The knowledge developed can be used further by other tools. The main criticism Easterbrook makes of systems developed to support requirements engineering is that they tend to support single viewpoints only. If we assume that inconsistency and conflict is the norm then this limitation is great. Nii's (1986) solution was to keep the knowledge from each expert separate and treat each KBS as a separate reasoning system.

Like the studies mentioned above, this study takes a KBS approach to requirements acquisition. This section focuses on the situation where the inputs are multiple A-boxes provided from multiple experts, the T-boxes are empty and the set of cases for each A-box is not empty, that is $A \neq \emptyset$, $T = \emptyset$ and $X \neq \emptyset$, where A, T and X denote an assertional KBS, terminological KBS and set of cases, respectively. A further restriction requires that the A-box must be convertible into a decision table. The work in Chapter Four and Five has shown how MCRDR systems can easily be converted into decision tables that do not require mapping of intermediate conclusions into primitive rules (objects). Conversion to a decision table is also suitable for production rule-based systems and has been applied to a number of CLIPS KBS as described in Chapter Eight. In this chapter RDR systems are distinguished from propositional rulebases. The latter are referred to as "standard rules".

This phase is also the maintenance phase for once one cycle is completed it is vital to ensure that the changes made to the explanation system (T-box) output from Phase Five are reflected in the appropriate individual and shared performance systems (A-boxes). The ease of KA, maintenance and mapping to a decision table with MCRDR makes it the preferred KBS approach. Also, as Herlea (1996) points out user involvement is a critical issue in making RE work and system development using RDR has been designed to be performed by the user.

As noted in the general framework, the input to Phase One also includes a set of cases. The importance of the set of cases is twofold. Firstly, on a general level, the cases are used in the negotiations as counterexamples for discussion. Secondly, using the MCRDR approach as described in Chapter Three, or other case-based technique, the cases are also used for initial KA and for modification of other views. For example, as one of the modification strategies outlined below (see Figure 6.4), when a concept is found to be in conflict, the case or cases associated with that concept are passed to the other stakeholders for KA. This should either resolve the conflict or at least ensure that all parties have given their views given the same set of criteria. In Section 6.5 on Related Work it is suggested how cases can be obtained.

6.3.2 Phase Two: Requirements integration

Requirements integration is the process of ensuring that all viewpoints are in formats that can be compared. Adopting the general framework, it may be that viewpoints have been captured using different KR. To avoid the requirement of mapping from all KR's used into all other KR's, that is N^2 mapping schemes, all KRs are converted into one format so that only 2N mapping schemes are needed. As mentioned in Phase One, in the current implementation any representation that maps into a decision table can be used so that a common approach to subsequent phases can be taken. It will become apparent in the next section why the decision table format restriction exists.

6.3.3 Phase Three: Concept Generation

In the general framework, an explanation system could already exist (that is, $T \neq \emptyset$ in Phase One). Alternatively, it could be supplemented or built in this phase. In the current work, we restrict ourselves to the case where $T = \emptyset$. The approach chosen is to begin with a performance system and later derive the explanation system. We start with a set of privately owned and defended A-boxes ($A_1..A_i$) written by some experts ($V_1..V_i$). The knowledge base also includes some data structures generated from previous cycles of the five phases. These structures are:

- One circumvent table for each A-box. This table identifies which rules not to include in future RE sessions.
- One subsumes table for the entire system. This table stores mappings of different terms to a common term.
- One CircumDelayIgnore table for the entire system. This table tags which T-box conflicts have been marked as “circumvented”, “ignored” or “delayed” in the previous cycle.

For more on these tables, see Section 6.3.5.

Easterbrook (1991) points out that one of the reasons why systems fail to meet the user’s needs is that the original mental model of the user has not been captured in the final design model. By letting the user have direct involvement with system development and using a simple KA approach based on the user’s behaviour, the capture and validation of assertions seems more feasible. Using FCA on the KBS rules as described in Chapter Five the leap from A-box to T-box is achieved which is demonstrated in an example below based on the Library World (Finkelstein et al 1994).

In the Library World there is a hierarchy of agents made up of borrowers, staff (which is further subdivided into librarians and clerks), library and catalogue. In this example, only the viewpoints of the borrower, librarian and clerk agents are considered as they can be considered “direct” agents which not only supply or receive information but they also process information (Finkelstein et al 1994). The action table for the “borrower” agent in Finkelstein et al 1994, which is reproduced in Figure 6.1 below, is interpreted as a formal context as shown in Figure 6.2.

SOURCE	INPUT	ACTION	OUTPUT	DESTINATION
Borrower	book	check-in	book	Clerk
	card		card	
Library	book	check-out	book	Borrower

Figure 6.1: An action table elaborating the agent “Borrower” From Finkelstein et al 1994, p. 7.

In Figure 6.2 we have the formal context “Library from the Borrower Viewpoint” with the set of objects G = {Borrower check-in, Borrower check-out, Library check-out} and set of attributes M = {source borrower, source library, input book, input card, action check-in, action check-out, output book, output card, destination borrower, destination clerk}. The crosses show where a relation between the object and attribute exists, thus the set of relations I = {(Borrower check-in, source borrower), (Borrower check-in, input book),…,(Library check-out, destination borrower)}. As described in the previous chapter, each row in the crosstable represents a concept. Using FCA we can form a complete lattice as shown in Figure 6.3.

	source borrower	source library	input book	input card	action check-in	action check-out	output book	output card	dest. borrower	dest. clerk
Borrower-check-in	X		X		X					X
Borrower-check-out	X			X		X	X	X	X	
Library-check-out		X	X			X	X	X	X	

Figure 6.2: Context of “Library from Borrower Viewpoint”

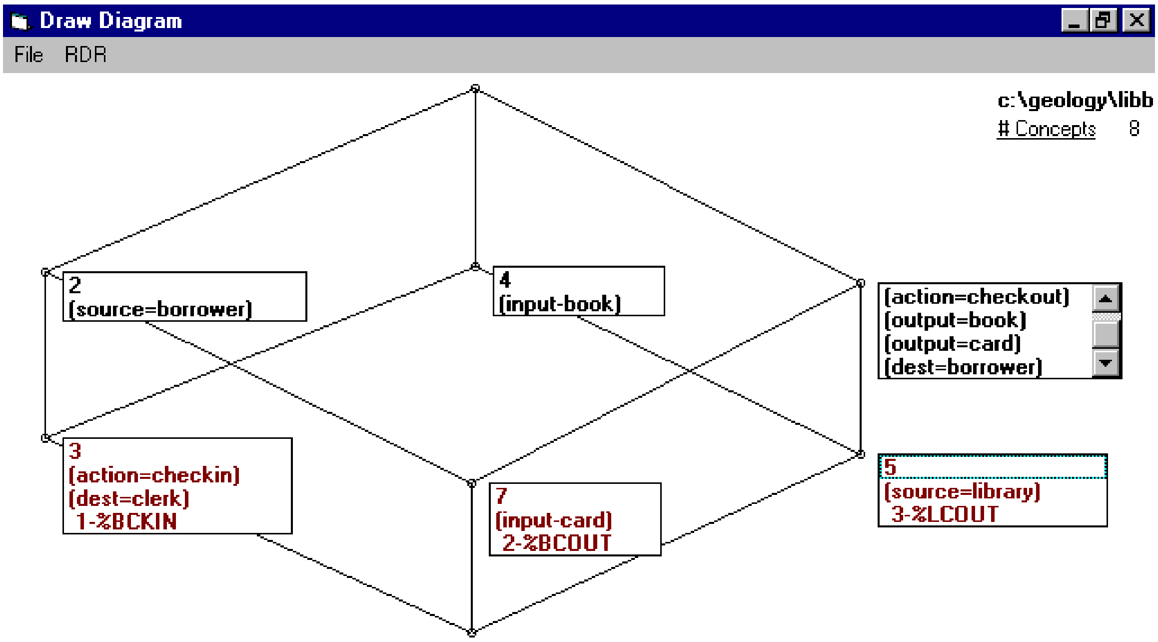


Figure 6.3:The Line Diagram for the Formal Context “Library from Borrower Viewpoint”.

A formal context and line diagram may be developed for each of the viewpoints in the Library World. The line diagrams can then be compared to detect differences between viewpoints. This process is considered next.

6.3.4 Phase Four: Concept Comparison and Conflict Detection

A number of researchers offer different sets of conflict types (e.g. Easterbrook 1991 and Schwanke and Kaiser 1988). This study has chosen to use the four quadrant model of comparison between experts developed by Gaines and Shaw (1989) which is used to classify two conceptual models as being in one of four states:

- Consensus is the situation where experts describe the same concepts using the same terminology.

- Correspondence occurs where experts describe the same concepts but use different terminology.
- Conflict is where different concepts are being described but the same terms are used.
- Contrast is where there is no similarity between concepts or the terminology used.

In other sections a broader view of conflict is taken and encompasses inconsistencies that include the states of contrast, correspondence and conflict. Gaines and Shaw's model, however, does offer greater precision in describing the nature of the conflict which is important in deciding how it can be handled. More formally the states of consensus and contrast are defined according to the FCA notion of a concept as a related set of attributes and objects. V denotes a View, P denotes a concept, B denotes a set of attributes and O denotes a set of objects. For an example of each of the four states as they may appear on a concept lattice see section 6.4.4 and Figure 6.9.

Consensus $\{V_1.P_{i.}.B_j\} = \{V_2.P_{k.}.B_L\}$ where $B_j = B_L$ and (6.1)
$\{V_1.P_{i.}.O_j\} = \{V_2.P_{k.}.O_L\}$ where $O_j = O_L$

Contrast $\{V_1.P_{i.}.B_j \cap V_2.P_{k.}.B_L\} = \varnothing$ and (6.2)
$\{V_1.P_{i.}.O_j \cap V_2.P_{k.}.O_L\} = \varnothing$

As shown in (6.1), consensus exists where *all* of the attributes and objects of a concept in one viewpoint match with a concept in another viewpoint. The search for a match is sequential and terminates when a match is found or when all the concepts in the other viewpoint have been compared. Thus in the worst case we need N x N-1 comparisons. Discovering the consensus between conceptual models is important for establishing common grounds from which differences can be viewed. Contrast (6.2) can be viewed as the opposite to consensus and exists when *none* of the attributes or objects in a concept in one viewpoint can be found in any of the concepts in another viewpoint. The search for a contrasting concept terminates when a match on any attribute or object is found or when all concepts in the other viewpoint have been processed.

A concept not in a state of consensus (match found in another viewpoint) or contrast (completely different to all concepts in another viewpoint) is then either in a state of correspondence or conflict. The key to deciding which state it belongs to depends on the terminology. In the approach presented it would be up to the stakeholders to decide whether the terminology used for an attribute or object was the cause for two concepts not appearing at the same node. If more assistance for the user is desired, Gaines and Shaw (1989) have shown that the repertory grid technique, which the crosstable could possibly be mapped into, can be used to identify where terminology is the cause of inconsistency. Having detected various conflicts let us consider how to resolve them.

6.3.5 Phase Five: Conflict Negotiation

Before it can be decided how to fix a detected inconsistency it is necessary to provide a conflict resolution strategy. There are a number of resolution methods which include negotiation, arbitration, coercion and education (Strauss 1978). Negotiation is the most appropriate within the assumed context of parties of equal status and ability. As Easterbrook (1991) points out, a good solution will require creativity and creativity is not something that can be automated. However, since automation is a fundamental goal of requirements engineering research the approach is extended beyond a general, genial chat by offering as much automated assistance for this step as possible.

Each RE researcher appears to use a different set of resolution strategies (e.g. Easterbrook 1991, Thomas 1976). Easterbrook and Nuseibeh (1996) offer five categories that cover the actions that have been found necessary. These are:

- Resolve, correct any errors;
- Ignore, no action is performed;
- Delay, identify the existence of the inconsistency but defer action until a later date;
- Circumvent, identify the existence of the inconsistency so it can be avoided;
- Ameliorate, reduce the degree of inconsistency. This action requires analysis and reasoning.

Resolving conflict will involve performing modifications. If the cause of disagreement is differences in terminology (correspondence in the Gaines and Shaw four state model) then one technique is to up-date all views to conform to an agreed upon set of terminology. This option is probably not satisfactory to the various stakeholders and also means that the history of changes is lost or altered. A simple and more appropriate solution is to use a subsumes table which maps terms from individual views into a shared terminology which is then used for comparison. The use of individual, rather than shared, terminology is a key feature of this work and as is shown in the SISYPHUS III case study later it is important that the terminology used in each source is not altered or reinterpreted. Naturally terminology will need to be reconciled to achieve a reduction in conflict but the choice of what can be treated as the same or similar is left to the stakeholders and is updated in the shared view only. Where a stakeholder feels that their own terminology is superior or more precise but still related to a term in another viewpoint/s it is possible to introduce an intermediate term, which may represent an abstracted concept from the original, to all viewpoints sharing the higher level concept so that the combined viewpoint reflects consensus at the higher but not at the lower level. The original terms in the individual viewpoints are not changed or lost.

Another way in which conflict may be resolved is through the addition or deletion of attributes or objects. The conflict may be due to the set of attributes or objects being only partially shared by another concept. To bring these concepts into a state of consensus it may be decided to drop or add attributes or objects. As mentioned in Phase One, part of the automated support for negotiation is the ability to produce a case associated with the object (rule) that is in question. The cases associated with all objects that can be reached by downward paths are also relevant to the discussion. The closer, distance measured in number of objects separating the two nodes, the object is to the concept in question the more relevant the case should be considered. New attributes or objects could also be added by showing the associated case to the other party and using that case for KA. Alternatively, if a hypothetical case is shown to be impossible, then the rules based on this case should be dropped. The decision of what action to take is made in this last phase and performed in Phase One. In Figures 6.4(a) and (b) a summary of the strategies applicable to standard rules and MCRDR is provided. Note that:

- In each strategy for handling attributes or objects from standard rules, the requirement has been added that some sort of checking should be done after each change to ensure that there are no unknown side effects elsewhere in the rulebase. With MCRDR this checking is given for free because no previously correctly classified cases can become misclassified with the RDR approach to KA and the exception structure.
- RDR has a very tight link between rules and cases. This link is not defined for standard rule bases. Hence, handling case addition/deletion is not defined in standard rule bases (see Figure 6.4(b)).
-

	Add Attribute	Delete Attribute	Add Object	Delete Object
Standard Rules	Add attribute to rule. Check effect on other rules.	Remove attribute from rule. Check effect on other rules.	Add new rule. Check effect on other rules.	Remove rule. Check effect on other rules.
MCRDR	Defined - use existing KA approach	Add stopping rule to rule in error. Add new rule at top level with the old rule minus the attribute to be removed.	Perform KA using the case associated with the concept which has the desired object.	Add a stopping rule.

Figure 6.4(a) The resolution strategy for handling attribute and objects.

	Adding Real Cases	Deleting Real Cases	Adding Hypothetical Cases	Deleting Hypothetical Cases
Standard Rules	N/A	N/A	N/A	N/A
MCRDR	Show to all	ILLEGAL	Show to all	Drop related rules - check refinement rules if they should be dropped or stopped and a new rule, without the dropped rule conditions, added.

Figure 6.4(b) The resolution strategy for handling cases.

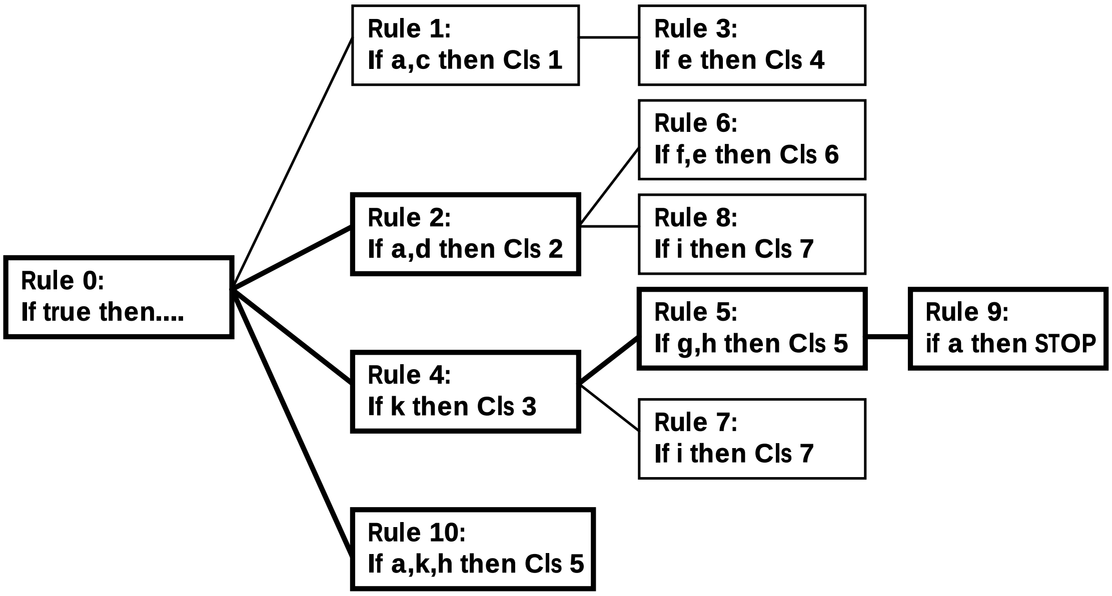


Figure 6.5. An MCRDR KBS with stopping rules.

This KBS is a modified version of Figure 3.2 which demonstrates the use of the stopping rule. The highlighted boxes represent rules that are satisfied for the case {a,d,g,h,k}. We can see that there are two independent conclusions for this case, Class 2 (Rule 2) and Class 5 (Rule 10). Rule 5 had been the cause of a conflict between viewpoints. To resolve this conflict it was decided that attribute g should be dropped. As described in Figure 6.4, the STOPPING RULE is used to say that this pathway should not fire, so even though the case satisfies Rule 5 that rule is stopped from being reported. We can see that Rule 10 now replaces the rule pathway for Rule 5 dropping the attribute g.

An example of how one attribute may be dropped and another added using MCRDR and the resolution operators is shown in Figure 6.5 which shows the important role of stopping rules to facilitate these changes. In Figure 6.5 the user has decided to add the attribute “a” and delete the attribute “g” to the concept associated with rule 5. The original rule 5 has been stopped and is replaced by rule 10.

The negotiation strategies can be strengthened by offering filtering rules which guide the dialogue between the stakeholder and the system. One filter is the use of preference criteria to guide the stakeholders in deciding what part of the view should be considered first as a candidate for change. Taking the concept lattice structure into consideration (where objects belonging to a concept are reached by descending paths and attributes are reached by ascending paths) it can be said that if an attribute at the bottom of a pathway or an object at the top of a pathway is to be removed then no other concepts will be affected and this act can be performed without further investigation. This strategy can be useful for example, if it had been agreed upon that only one of two attributes were necessary and one was at the bottom of a pathway and the other higher up, then it would be advisable to remove the lower attribute. Other templates or KA scripts (Gil and Tallis 1997) can be used to guide the user with revising their KBS and ensuring that each of the rules related to the change are modified and tested.

The last four resolution strategies are relevant for situations in which a complete resolution can not be negotiated and each one has its appropriate usage. For example, ignoring is a useful strategy where the issue is not that important or pursuing it is not worth the effort or harm it may cause to the end solution. These approaches can be termed as living with inconsistency or 'lazy' consistency (Narayanasway and Goldman 1992) and can be compared to fault-tolerant systems that continue to function after non-critical failures occur. It has been argued that enforcing removal of all inconsistency "constrains the specification unnecessarily" and "tends to restrict the development process and stifle novelty and invention" (Finkelstein et al 1994, p.2 & 4). They see that consistency is necessary within a viewpoint but partial consistency between viewpoints is allowable.

This study also accepts that living with inconsistency will be necessary and uses tags to identify the status of the conflict. The use of tags is similar to the use of "pollution markers" (Balzer 1991) that act as a warning that code may be unstable or that the users should carefully check the output. Pollution markers can also be used to screen inconsistent data from critical paths that must have completely consistent input. If it is the concept that is being circumvented, ignored or delayed, the concept is marked in the shared T-box since there is not necessarily a one-to-one correspondence between rules and concepts. This updated T-box is used as input in the next T-box generation. When it comes to rules (the A-box) the only tagging strategy offered is circumvention. Tagging a rule as "circumvented" in the individual A-boxes is used to avoid certain unstable parts of the requirements. When the new T-boxes are generated these rules will not be included.

The resolution strategies shown in Figures 6.4(a) and (b) are also applicable to the strategy of amelioration. However, the result is not consensus but a reduction in the degree of the conflict. Amelioration results in bringing concepts closer together. If we think in terms of the concept lattice the result will be a shortening in the distance between the two concept nodes.

6.3.6 Evaluation

To evaluate that our RE operators are reducing the amount of conflict between viewpoint we need two sub-routines:

- A "distance metric" which can assess how far apart are two viewpoints.
- A set of "negotiation operators" which extract features from the viewpoints that could lead to reducing the distance metric.

These RE sub-routines can now be evaluated as follows. Different negotiation operators can be assessed via their observed effects on the distance metric: good RE operators have a big impact on this distance. Operators also have a construction cost: good operators should be easy to build. An ideal RE operator is cheap to build, cheap to use, and reduces the distance metric considerably (though, in practice, some trade-offs may be required between operator complexity and operator effectiveness).

The negotiation operators already defined are used to compare the degree of conflict before and after. By computing a score for each concept in each viewpoint compared to each other viewpoint and taking the total of these scores it can be checked that the degree of conflict after the RE process is less than at the start. A score of 0 is assigned to a concept found to be in a state of consensus with a concept in another viewpoint, since the distance between them is zero. For concepts in a state of contrast (no partial or complete match in the other viewpoint) a score of 1 is assigned, which is the same result as if the conflict measure was used since the number of attributes not shared divided by the number of attributes is equal to one. These measures assume that the two concepts being compared share the same object (conclusion). If they do not then it appears that they are not meant to represent the same concept so that comparison is not meaningful. Concepts in a state of correspondence are treated the same as concepts in a state of conflict since the reason for conflict depends on user input and the focus here is on the size of the difference not the cause. Once terminology differences are reconciled such concepts will move into one of the other states and be handled accordingly.

These measures have been used in Section 6.4.6 to determine how well the RE strategy is working on the SISYHPUS III viewpoints. Since in the next section we restrict ourselves to looking at one conclusion at a time in all viewpoints, the measures described are appropriate. Of greater difficulty is determining the degree of conflict if it is not known which concept in the other view represents a similar concept. It is necessary to either get the experts to label their concepts so that, for example, two concepts labelled “cat” would be compared or to compare the concept in one viewpoint with all others in the other viewpoint finding the one that shares the largest set of attributes on the assumption that it is the concept which most likely represents the same concept. For the purposes of the example in the next section it was not necessary to resolve this problem and further work on distance measures is part of the future RE work.

6.4 Requirements Engineering using SISYPHUS III: A Case Study

It is argued in this chapter that the needs of knowledge engineering using multiple sources of expertise and those of requirements engineering (RE) from multiple perspectives are very similar and that an RE approach to management of conflict is superior to the knowledge engineering approach that relies on the KE to resolve differences. The SISYPHUS III rock domain, in particular, is a good example of a domain where there is little consensus of opinion in the actual classification of individual rocks. Many rocks are classified differently, with some experts arguing that certain rock classifications do not exist but are simply different examples of another rock classification whose appearance has been altered due to the formation process they have undergone. The disagreement amongst experts and problems that this causes is evident in a number of approaches to solving the SISYPHUS III problem. For example, the KADS (Jansen, Schreiber and Wielinga 1998) solution was to abandon the data provided and to start again using other sources such as books and web-based tutorials. However, there are also differences between these sources which they have endeavoured to reconcile by becoming experts in the domain and making their own decision as to which viewpoint to adopt. As an example, Jansen, Schreiber and Wielinga (1998) point out that the rock Adamellite was missing from three key sources but that eventually they managed to find “an exact description”. So they have decided to accept the viewpoint of the source containing a description but it appears that if such an exact description were well accepted it would not have been so hard to find. It is interesting to note that this happened to be the rock chosen for many of the examples that follow. With such inconsistency within the domain, the data provided by the SISYPHUS III appeared to provide a very suitable testbed for the RE approach presented in Sections 6.1-6.3. We look first at the system requirements of the SISYPHUS III experiment.

The problem statement and scope of SISYPHUS III is given as (Shadbolt 1996):

> *During the Apollo moon program it came to be recognised that many of the primary scientific objectives of the missions could not be achieved without the astronauts acquiring a certain level of geological competence. This included the ability to undertake the collection and documentation of rock samples from the lunar surface. All of the moonwalkers except one - Harrison Schmidt - were taught their geology in snatched excursions with field geologists to selected sites in the USA, and via a limited number of class based lectures. We can expect that future manned missions to the moon and other planets will also contain non-specialists who will be expected to carry out similar tasks.*
>
> *In Sisyphus III we are building a geological expert system for rock sample characterisation. This is intended to act as a tutorial aid and diagnostic decision support system for the trainee astronauts. The final run-time system will be expected to run on a high end colour laptop computer. The initial requirement is that a system be ready for preliminary trials within four months. It must be capable of identifying major types of igneous rocks. Igneous rocks are materials that have solidified from*

molten or partially molten material. These were probably the first formed portions of the earth's surface. They have also provided most of the components of the other two rock types - sedimentary and metamorphic rocks. Igneous rocks are normally first studied in the field and then in the laboratory. However, in this case it has been stipulated that the users should be able to use the system in conjunction only with a hand specimen of the rock, a hand lens and a prepared thin section of the rock that can be viewed using a geological microscope. The aim is to support the description and discrimination of the major types of rock along their most salient and pertinent characteristics.

The problem description is extended a little further for the purposes of this case study. The KA material supplied with the first phase includes: 5 card sorts, 4 laddered grids, 5 structured interviews, 4 self reports and 4 repertory grids. The card sorts and repertory grids provide useful frameworks that require the least amount of data extraction compared to the other techniques that use natural language. The problem description is extended beyond the development of a system that classifies a set of igneous rocks correctly and offers an explanation of the conclusion to the question of how to use data from multiple and conflicting sources of expertise to build this system. These multiple sources of expertise impose a further requirement not explicitly stated in the problem statement above, which is to develop a strategy for handling any differences.

The RDR SISYPHUS III submission produced a number of KBS from a combination of the above sources. The systems were built by a geological and KE novice in the same manner that has been used in this study which is described in section 6.4.1. Differences in terminology, measurement and inconsistencies between the sources were reconciled using the rule of thumb “choose the option that most experts agreed on”, where each KA source was treated as an expert. This is a reasonable heuristic but one that can not always be applied since in some cases it requires interpretation to decide that two concepts match. Thus, to some extent the main final system is a product of what the novice KE thought was correct. The approach outlined in this chapter for reconciling conflict in multiple viewpoints appeared to provide a better way of arriving at a final though not complete model that took into account each perspective without the bias of the KE as the method of resolution. It is proposed that even without the involvement of the stakeholders in the resolution process the RE framework offered is a more methodical approach to conflict resolution than trying to handle conflict when and if it is detected.

In the next subsections the five phases already presented are used to show how the proposed framework can be utilised. Note that the resolution process is conducted on a small scale, that is, the conclusions are compared one at a time rather than comparing all the complete models in one go. The main reason for this is the more information being dealt with, in particular represented visually, the more incomprehensible the diagram becomes. While this approach may be seen as time consuming it is not necessarily the most inefficient because problems can often be better identified and resolved when broken down into smaller problems. An expected benefit of this incremental approach is that as conflicts are resolved, for example differences in terminology are reconciled through the subsumes table or concepts are tagged to be ignored, there will be progressively less errors or differences.

6.4.1 Phase One - The Data and Knowledge Acquisition

```
#1 10-03-1997 09:16:57
GRAIN_SIZE
  COARSELY_CRYSTALLINE
SILICA           VERY_HIGH
COLOUR LIGHT
OLIVINE UNLIKELY
MICA             LIKELY
#2 10-03-1997 09:16:57
GRAIN_SIZE       FINE
SILICA
  INTERMEDIATE
COLOUR FAIRLY_LIGHT
OLIVINE          UNLIKELY
MICA             UNLIKELY
#3 10-03-1997 09:16:57
GRAIN_SIZE       FINE
SILICA           BASIC
COLOUR DARK
OLIVINE LIKELY
MICA             UNLIKELY
```

Figure 6.6: Extract from Card Sort 5 Case File.

To reconcile the viewpoints contained in each of the data sources mentioned above, The KE novice developed six individual KBS using MCRDR for each of the card sorts and laddered grids. For the purposes of this case study the other material was not used even though supporting or extra information had originally been gleaned from them and added to the final KBS. It was important that the models should only represent the data directly specified as related to them and avoid inclusion of the KE's own interpretation[2]. The card sorts contained differences in the number of attributes used, the attributes chosen, the values assigned to attributes and the terminology of attributes and objects. It was these types of differences I wished to reconcile. One alternative is to force a common set of terminology and scales to be used. However, it is important to stakeholders/experts that they be allowed to express their models in their own words. This is supported by the work on repertory grids, where subjects preferred to use their own constructs over ones supplied to them (Shaw 1980).

The data in the card sorts were used directly to develop cases. In Figure 6.6 the first three records in the Card5 case file can be seen. For the restricted study performed in this chapter, a case file was developed only for Card Sorts 1, 2, 3 and 5 and Laddered

[2] As stated here it was the aim of this study to avoid interpretation by the KE. Unfortunately, it was found later that the KE had made certain decisions regarding the data. For example, in Card Sort 5 Adamellite, Dacite, Granodiorite and Granite had been described by the expert using identical features. The KE had decided to drop Dacite and Granite and to change a feature of Granodiorite. This was the type of situation I had wanted to avoid. If all four rocks had been classified using the same features they would have appeared at the same node and have shown that either an error had been made and/or additional condition/s were needed to differentiate between the four rocks. In this way our modelling technique would have been useful for identifying not only conflict between but also conflict within a viewpoint.

Grids 1 and 3[3], and will be referred to as C1, C2, C3, C5, L1 and L2, respectively. Using the appropriate case file, KBS were developed by running an inference on each case. If the conclusion given by the system was incorrect, a new conclusion was assigned and the attribute value pairs which justified that conclusion were selected as the rule conditions. The conditions were selected based on intuition and the *difference list* that shows the differences between the current case and the case that gave the misclassification.

```
0
11
12
1 0 11 0  %NC000 : 1 = 1
2 1 0 0   %AD000 : (GRAIN_SIZE = COARSELY_CRYSTALLINE) & (SILICA = VERY_HIGH) &
                   (COLOUR = LIGHT)
3 1 0 2   %AN000 : (GRAIN_SIZE = FINE) & (SILICA = INTERMEDIATE)
4 1 0 3   %BA000 : (GRAIN_SIZE = FINE) & (SILICA = BASIC)
5 1 0 4   %DI000  :(GRAIN_SIZE=COARSELY_CRYSTALLINE) &(SILICA=INTERMEDIATE) &
                   (COLOUR=FAIRLY_LIGHT
6 1 0 5   %DO000 : (GRAIN_SIZE = NOT_COARSE) & (SILICA = BASIC)
7 1 0 6   %DU000 : (GRAIN_SIZE = COARSELY_CRYSTALLINE) & (SILICA = ULTRABASIC) &
                   (COLOUR = DARK)
8 1 0 7   %GA000 : (GRAIN_SIZE = COARSELY_CRYSTALLINE) & (SILICA = BASIC) &
                   (COLOUR = DARK)
9 1 0 8   %KE000 : (GRAIN_SIZE = ?)
10 1 0 9  %GR002 : (GRAIN_SIZE = NOT_COARSE) & (SILICA = VERY_HIGH)
11 1 0 10 %RH000 : (GRAIN_SIZE = FINE) & (COLOUR = LIGHT)
```

Figure 6.7: The rule file for Card Sort 5.

The first rule number, the total number of rules and the number of cases associated with the rules are given in the first three lines. The default rule is on line four. The structure of each rule is rule number, parent rule number, child rule number, sibling rule number, conclusion code and the conjunction of conditions appear after the colon.

The MCRDR rule file developed from the Card5 case file is shown in Figure 6.7. For the rest of this case study, each KBS is treated as a different expert viewpoint which is in fact the origin of each card sort and laddered grid (Shadbolt, personal communication).

4.2 Phase Two: Requirements Integration

As described in the framework, each KBS is converted into a common format, specifically a crosstable or decision table. Figure 6.8 shows the crosstable, which is a formal context, for the first four MCRDR rules for Card Sort 5.

[3] To make the examples using the conclusion %AD000- Adamellite in Figures 6.9, 6.11 and 6.13 more complete, I developed a partial KBS for Card Sort 4 to cover that rock so that seven viewpoints are actually compared in the example.

	1=1	grain_size= coarsely_crystalline	silica= very_high	colour= light	grain_size= fine	silica= intermediate	silica= basic
1-%NC000	X						
2-%AD000	X	X	X	X			
3-%AN000	X				X	X	
4-%BA000	X				X		X

Figure 6.8: Crosstable of the first four MCRDR rules for Card Sort 5

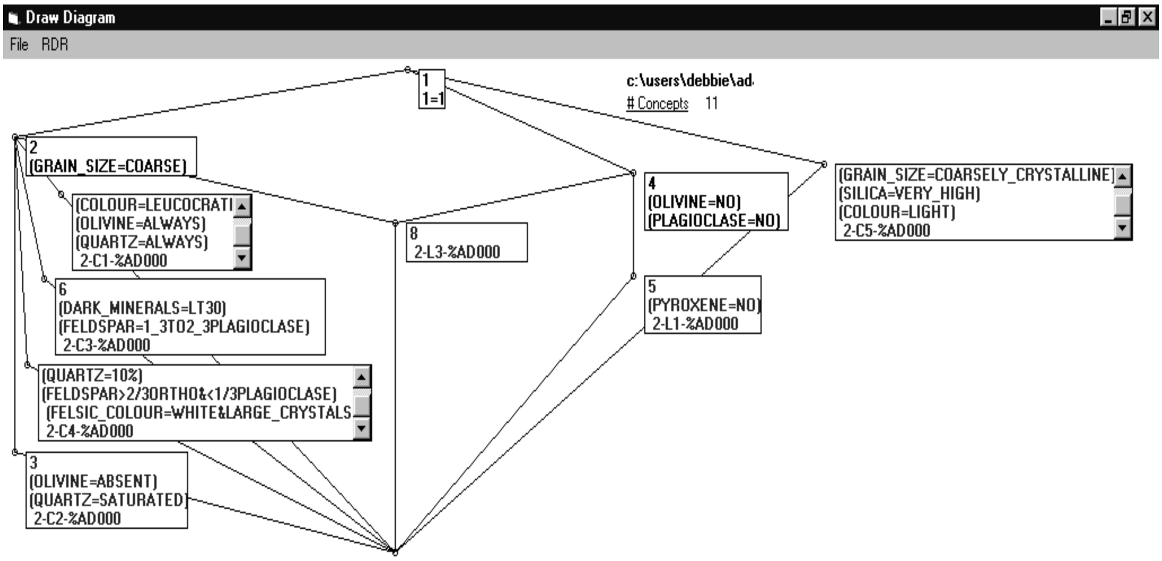


Figure 6.9: The Concept Lattice for the Conclusion %AD000- Adamellite based on seven KBS.

6.4.3 Phase Three: Concept Generation

In this phase the crosstable is used to derive the ordered set of formal concepts which can be shown as a concept lattice. When dealing with multiple KBS there are two main alternative approaches to implementation of this phase. The crosstables or formal contexts output from Phase Two can be combined into one formal context with the source of each row annotated[4] for identification OR the concepts for each formal context can be derived and then compared. The simplest method with the existing MCRDR/FCA tool is to combine all the individual formal contexts into one formal context and perform queries or comparisons on them as a whole. However, handling such a large context has its problems. As noted before: “the diagrams for activities of any reasonable complexity become very difficult to visualise and understand” (Gaines and Shaw 1993a, p. 59). The MCRDR/FCA options described in Section 5.2.1 offered

---

[4] The approach to annotation was simply to add the source identification to the beginning of the conclusion code in column one of the crosstable so that in Figure 6.10 the first conclusion would read 2-C5-%AD000 which stood for rule 2, source Card Sort 5 and conclusion code %AD000.

13 (this number includes the choices within the options) combinable options that allow the user to narrow their focus of attention to selected part/s of interest in the knowledge base. In order to ensure that some parts of the KBS are not missed which may result in some conflict being undetected, it is important to select views that will cover all the knowledge in all the viewpoints. Since the major goal of the SISYPHUS III system is to correctly classify a particular rock it seems reasonable to explore each conclusion from the combined viewpoints to see what conflict exists. Figure 6.9 shows the concept lattice for the conclusion %AD000- Adamellite.

6.4.4 Concept Comparison and Conflict Detection

The concept lattice in figure 6.9 may be analysed and the nature of the differences between concepts determined. The case study described here involved the manual detection of conflicts from the line diagram. However, as described briefly in Section 5.2.2, a tool was also developed which allowed up to four viewpoints to be loaded, which were used to build two different line diagrams. One diagram showed all the concepts shared by the combined viewpoints and the other diagram showed the complement of the shared concepts. The idea behind this was that it was an easy and quick way of identifying common ground and what concepts/terminology needed to be negotiated. These diagrams are not used here as the aim is display the combined viewpoints with shared and unshared concepts and to show how the distance between viewpoints can be reduced.

Looking at Figure 6.9, we can see that the C5 viewpoint for Adamellite is in a state of contrast since none of its attributes are shared with any of the other viewpoints. However, some of these differences appear to be terminology related. There is consensus that GRAIN_SIZE=COARSE between C1, C2, C3 and L2 but in C5 the GRAIN_SIZE=COARSELY_CRYSTALLINE. This appears to be a correspondence type of conflict because of differences in terminology. There appear to be other correspondence errors. The attribute QUARTZ is used in C1, C2 and C4 with the values ALWAYS, SATURATED and 10%, respectively. The value of OLIVINE in L1 and L2 is NO and in C2 the value is ABSENT. In C5 the COLOUR=LIGHT and in C1 the COLOUR=LEUCRATIC. The dictionary meaning of “leuco” is white (Macquarie Dictionary), so it appears that the terms in these two concepts are compatible. It also appears that the feature DARK_MINERAL=LT30 indicates a lightness of colour. The differences in the terminology used for the values assigned to GRAIN_SIZE, QUARTZ, OLIVINE and COLOUR can be reconciled by using the subsumes table to map to a common term as shown in Figure 6.10.

The value assigned to OLIVINE in C1 is ALWAYS and represents a conflict where the terms are compatible but the concept is obviously the opposite to the concepts in L1, L2 and C2. There is consensus between L1 and L2 that PLAGIOCLASE=NO but these concepts conflict with the concepts FELDSPAR=1_3TO2_3PLAGIOCLASE for C3 and FELDSPAR>2/3ORTHO&<1/3 PLAGIOCLASE for C4. These various conflicts need to be resolved which takes us to the next stage.

6.4.5 Phase 5: Negotiation

Original Term	Synonym
GRAIN_SIZE=COARSLY_CRYSTALLINE	GRAIN_SIZE=COARSE
QUARTZ=SATURATED	QUARTZ=ALWAYS
QUARTZ=10%	QUARTZ=ALWAYS
OLIVINE=ABSENT	OLIVINE=NO
COLOUR=LEUCRATIC	COLOUR=LIGHT
FELSIC_COLOUR=WHITE&LARGE_CRYTALS	COLOUR=LIGHT
DARK_MINERAL=LT30	COLOUR=LIGHT

Figure 6.10: The Subsumes Table

Reconciliation of conflicts in terminology is a good place to start in the negotiation process. Those terms identified to be similar are put in the Subumes Table, see Figure 6.10. As noted in Section 6.3.5 individual terms may be kept and a new shared term is introduced in the situation unless there is agreement to change. With the updated Subsumes Table phases 1-5 are repeated. In Figure 6.11 the effect of the subsumes table on the original conflict in Figure 6.9 can be seen.

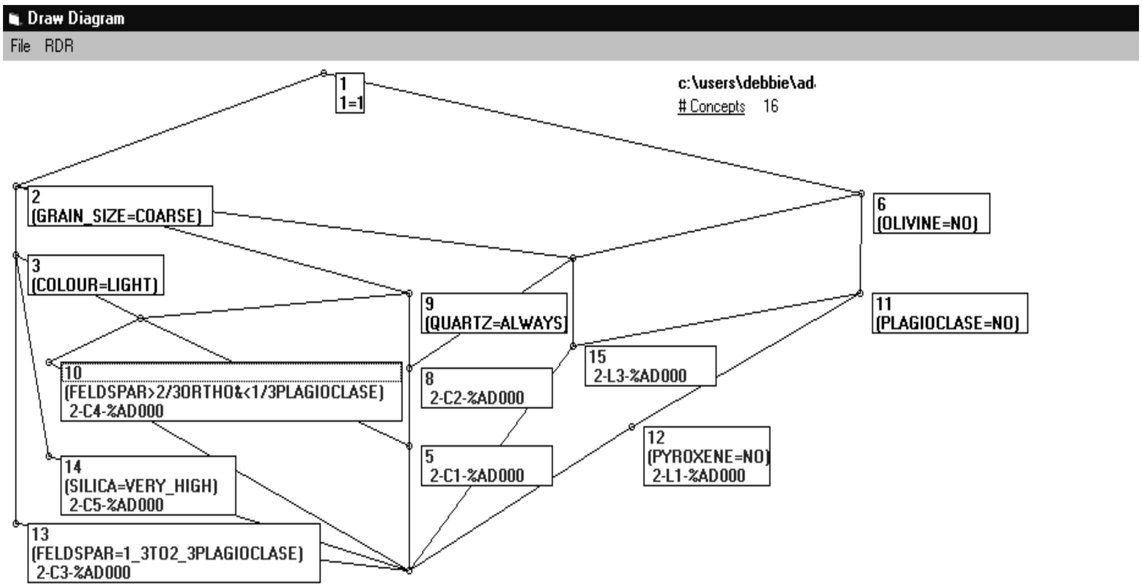


Figure 6.11: The updated Line Diagram after applying the Subsumes Table in Figure 6.10.

In addition to terminological differences there are some conceptual differences. The following scenario simulate how some of these differences can be resolved:

- Since sources C2, L1 and L2 agree that OLIVINE=NO is relevant, the expert in C1 decides that they have made an error and changes OLIVINE=ALWAYS to OLIVINE=NO. As described in Figure 6.4(a) this is done by adding a stopping rule to the incorrect rule and adding a new rule with all of the previous conditions but with the value of OLIVINE changed. This also requires amending the value of OLIVINE in the associated hypothetical case or passing a case from another viewpoint which covers this situation.
- To ignore or delay a concept the Ignore Tag Code “I” or Delay Tag Code “D”, respectively, is used as shown in Figure 6.12. This Tag Code is used to update the CircumDelayIgnore table and shows the concept in the concept lattice annotated with the word “IGNORE” or “DELAY”. Fig 6:13 shows two delayed concepts.
- The L1 expert agrees that GRAINSIZE=COARSE should be included so they amend their KBS by adding an exception rule to Rule 2, as outlined in Figure 6.4(a), where the conclusion is given as %AD000 and the condition as GRAINSIZE=COARSE.
- A feature of the approach is the ability to offer counterexamples that can be used in negotiations. In Figure 6.13 the case associated with Concept No. 8 (Rule 2 in Card Sort 5) is shown to the group to assist with reconciliation of this conflict. The user takes the Show_Case option from the RDR menu and specifies the source (which KBS) and the rule in which they are interested. The C5 and L1 experts can not be persuaded by the other experts to drop the attributes SILICA=VERY_HIGH and PYROXENE=NO, respectively, so it is decided to delay resolution of these conflicts until a later date. This is achieved by using the Ignore Tag which drops this concept and updates the CircumDelayIgnore table, see Sections 6.3.3 and 6.3.5. The user may also defer action on a conflicting concept by using the Delay Tag which updates the CircumDelayIgnore table as above but rather than dropping the concept, the control background and foreground colour is reversed and the word DELAYED is displayed as in Figure 6.13.
- To circumvent an object (rule) the related rule is flagged before a context is generated. In these examples all rules that conclude %NC000 (the no conclusion rule) have been circumvented.
- A final strategy concerns the handling of the controversy over the importance of FELDSPAR in determining if a rock is Adamellite. Expert C3 believes that the FELDSPAR content is one to two-thirds PLAGIO-CLASE. Expert C4 believes FELDSPAR is less than one-third PLAGIO-CLASE, experts L1 and L2 believe that PLAGIOCLASE = NO and experts C1 and C2 do not consider FELDSPAR or the PLAGIOCLASE content. It is thus decided to circumvent the concepts with these

attributes. This is achieved by selecting Concepts No 10, 11 and 13 in the CMATRIX screen, see Figure 6.12, and adding a “C” tag for circumvent. The concept and the tag are written to the CircumDelayIgnore Table. Once given this tag, a concept is not included in determination of the list of predecessors (parents) and list of successors (children) which are used to layout the line diagram. It can be seen in Fig 6.13 that these attributes are no longer shown. If desired, these concepts can be reinstated and shown.

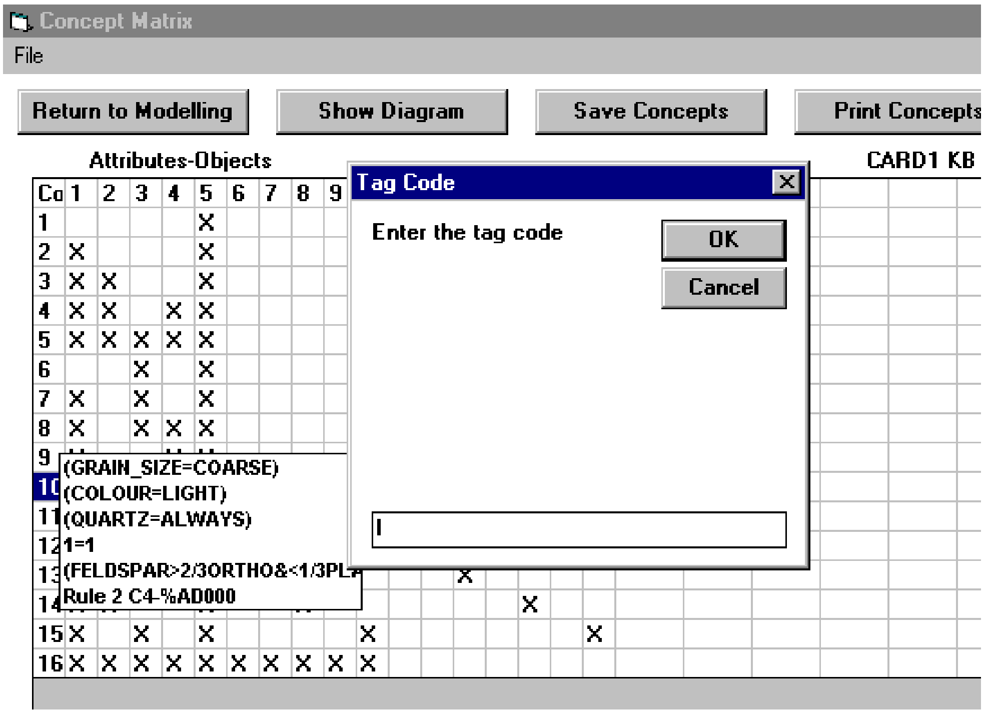


Figure 6.12: Tagging concepts to be “ignored” on the CMatrix Screen.
This is achieved by clicking on the Concept No and entering the Tag Code into the pop-up InputBox.

All of the changes mentioned above are reflected in the final diagram in Figure 6.13. Note that even though the number of concepts has only reduced by 1, the concepts are much less complex. In figure 6.9 GRAIN_SIZE= COARSE offered the most, but not total, point of agreement. Now all views agree with this and there are more attributes shared by viewpoints which previously only appeared in one viewpoint. As shown visually in Figure 6.13, the viewpoints in Card Sorts 1,2,3,4 and 5 are more similar to each other than the viewpoints in Laddered Grids 1 and 3 which are similar to each other.

As shown in the diagram in Figure 6.13 each of the viewpoints can be summarised as:
C1- GRAIN_SIZE=COARSE;COLOUR=LIGHT;OLIVINE=NO;QUARTZ=ALWAYS
C2- GRAIN_SIZE=COARSE; OLIVINE=NO;QUARTZ=ALWAYS
C3- GRAIN_SIZE=COARSE;COLOUR=LIGHT
C4 - GRAIN_SIZE=COARSE;COLOUR=LIGHT;QUARTZ=ALWAYS
C5- GRAIN_SIZE=COARSE;COLOUR=LIGHT;SILICA=VERY_HIGH(DELAYED)
L1- GRAIN_SIZE=COARSE;OLIVINE=NO;PLAGIOCLASE=NO;PYROXENE=NO(DELAYED)
L3- GRAIN_SIZE=COARSE;OLIVINE=NO;PLAGIOCLASE=NO

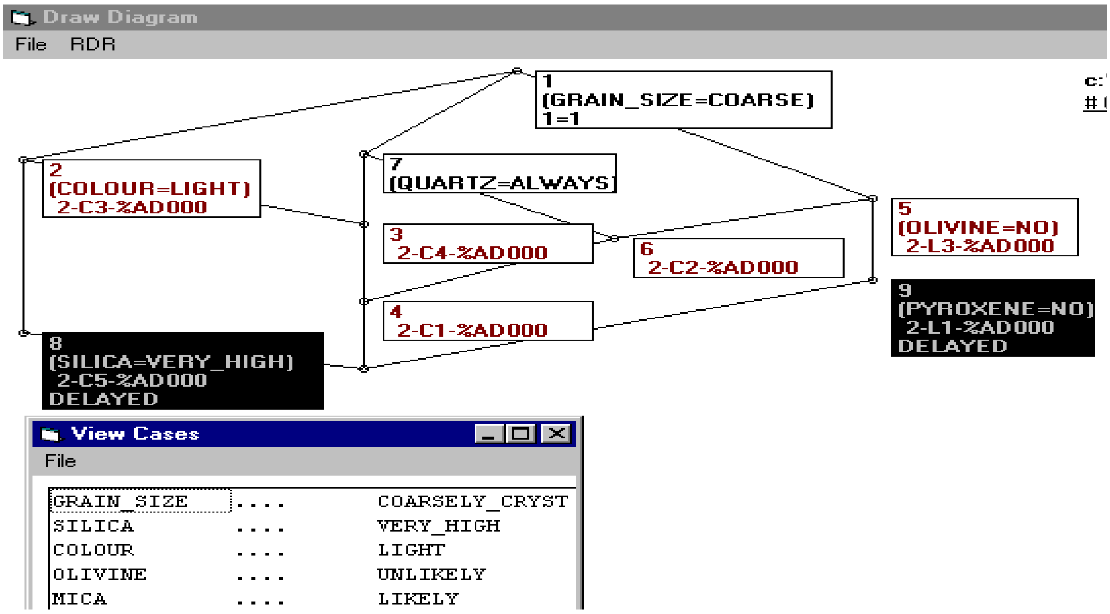


Figure 6.13: The final Line Diagram screen from this round of negotiations. This diagram includes all seven viewpoints (C1, C2, C3, C4, C5, L1 and L3) for the classification of the rock Adamellite. Some attributes have been dropped or added to views, concepts have been tagged to be circumvented (not shown) or delayed (shown). The case for concept 8 is shown to assist with negotiation. There is considerably less conflict now than in Figure 6.9.

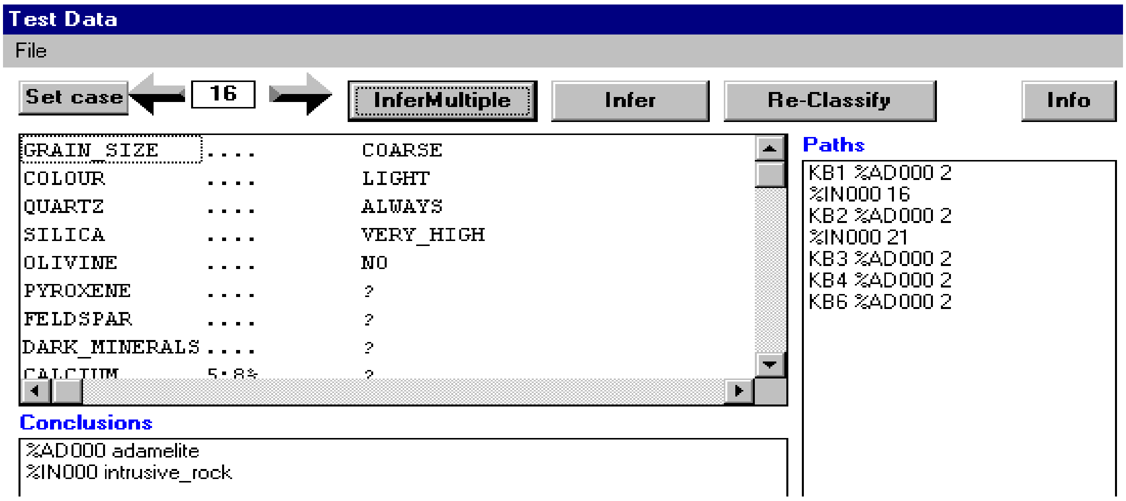


Fig 6.14: Running a Case against Multiple KBS as a Test on the Degree of Agreement

As a further test on the amount of agreement in the T-boxes, all six A-boxes can be loaded (recall only six were complete) and cases run to see the amount of agreement. In Figure 6.14 the case for Adamellite was selected and it was found that C1(KB1), C2(KB2), C3(KB3), C5(KB4) and L3(KB6) all concluded %AD000. The rule for Adamellite in L1 did not fire because it had the condition (PYROXENE=NO) which is not present in the case. If the value for PYROXENE in the case were modified to NO instead of missing then all KBS would correctly classify this rock specimen. Note that when this case was originally run before the modifications none of the KBS gave the conclusion %AD000. The reader is invited to verify this by using the case in Figure 6.14

against the concept lattice in Figure 6.9 treating the attributes that can be reached by an ascending pathway from a conclusion as the rule conditions.

Choosing a conclusion is only one of the many ways that viewpoints can be compared. It can be seen in figure 6.15, that comparisons made by attribute, in this case GRAIN_SIZE, can give us information on differences in terminology and measurement. This diagram can be used to detect possible synonyms (eg COARSE=COARSELY_CRYSTALLINE) and to compare the closeness of viewpoints. It can be seen in concepts 6, 7, and 8 that C5 uses terminology not used by the other views. If COARSELY_CRYSTALLINE were changed to COARSE and NOT_COARSE to MEDIUM then these objects would agree and move to the same concepts as in the other views. It can also be seen that experts C2 and C5 use "?" and UNSURE for the rock %KE000, Kentallenite. These two values are obviously equivalent. Since the other experts do not use the attribute GRAIN_SIZE for classifying Kentallenite, these experts may decide that it is not a useful classifier and drop these attributes (conditions) from the relevant rule. Each conclusion can also be compared to see which have been given the same value. For example, conclusions %AD000 has GRAIN_SIZE = COARSE in all views that use this attribute for that rock, but GRAIN_SIZE = FINE for %DA000-Dacite in C1 and C3, but C2 uses GRAIN_SIZE=MEDIUM for this rock. This may be due to a difference in perception or measurement and the rock may be borderline FINE_MEDIUM.

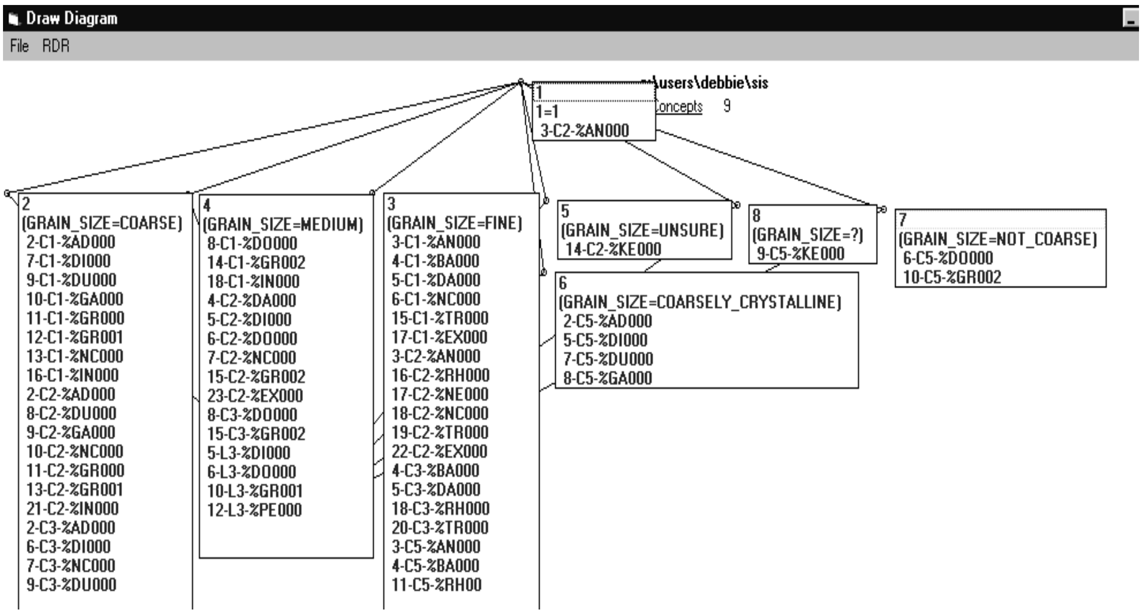


Figure 6.15: Line Diagram for the selection -"attribute = grain_size without related attributes".

6.4.6. Evaluation

Using the measures described in 6.3.6 the degree of conflict at three points was computed: at the beginning; after updating the subsumes table; and at the end. The only viewpoints/concepts in consensus are viewpoints/concepts matched against themselves. There are a number of contrasting concepts shown by the cell entries which equal 1, such as C1-C5 and C1-L1. Concepts shown as a fraction are in conflict where the numerator is the number of attributes not shared and the denominator is the total number of attributes in both concepts. The total degree of conflict for a viewpoint is shown as a decimal for better comparison.

	C1	C2	C3	C4	C5	L1	L3	TOTAL
C1	0	5/7	5/7	6/8	1	1	5/7	4.89
C2	5/7	0	4/6	5/7	1	1	4/6	4.76
C3	5/7	4/6	0	5/7	1	1	4/6	4.76
C4	6/8	5/7	5/7	0	1	1	5/7	4.89
C5	1	1	1	1	0	1	1	6.00
L1	1	1	1	1	1	0	2/6	5.33
L3	5/7	4/6	4/6	5/7	1	2/6	0	4.10
								34.74

Table 6.1: The degree of conflict between each viewpoint before RE.

In Table 6.1 it can be seen that all viewpoints are in conflict with others, with views C1, C2, C3, C4 and L3 having similar degrees of conflict. From this table it can be seen that viewpoint C5 is in complete contrast with all other views, thus it has the highest degree of conflict, followed by L1 that only shares some attributes with L3.

	C1	C2	C3	C4	C5	L1	L3	TOTAL
C1	0	3/7	3/7	2/8	1	1	5/7	3.25
C2	3/7	0	4/6	3/7	3/6	4/6	2/6	3.02
C3	3/7	4/6	0	3/7	2/6	1	4/6	3.52
C4	2/8	3/7	3/7	0	3/7	1	5/7	3.25
C5	1	3/6	2/6	3/7	0	1	4/6	3.35
L1	1	4/6	1	1	1	0	2/6	5.00
L3	5/7	2/6	4/6	5/7	4/61	2/6	0	3.48
								24.88

Table 6.2: The degree of conflict after the synonym table.

In Table 6.2, it can be seen that the total degree of conflict for each viewpoint is lower after similar terms have been reconciled using the subsumes table, even though it has not reduced the conflict in all cells. The total degree of conflict for all viewpoints has reduced from 34.74 before the resolution strategies were applied to 24.88 after the first strategy of reconciling terms was used. It is interesting to note that all views except L1 now have similar, though lower than before, degrees of conflict. This shows that much of the conflict originally in C5 was due to differences in terminology. This is not the case with L1 which is the most in conflict after this round of negotiations (Table 6.3).

	C1	C2	C3	C4	C5	L1	L3	TOTAL
C1	0	1/7	2/6	1/7	3/7	4/8	3/7	1.98
C2	1/7	0	3/5	2/6	4/6	3/7	2/6	2.50
C3	2/6	3/5	0	1/5	1/5	4/6	3/5	2.60
C4	1/7	2/6	1/5	0	2/6	5/7	4/6	2.39
C5	3/7	4/6	1/5	2/6	0	5/7	4/6	3.01
L1	4/8	3/7	4/6	5/7	5/7	0	1/7	3.17
L3	3/7	2/6	3/5	4/6	4/6	1/7	0	2.84
								18.49

Table 6.3: The degree of conflict between each viewpoint after RE.

Table 6.3 shows the degree of conflict remaining after the resolution strategies in Figure 6.4 and the various resolution Tag Codes were applied to produce the final concept lattice in Figure 6.13. As noted before not all conflict has been removed but it has reduced by 53% from 34.74 to 18.49.

6.5 *Related Work*

In the framework, cases play a critical role and it has been assumed that cases are available. In the use of the SISYPHUS III data, a source of cases has been available but this may not always be so, particularly when dealing with software specification requirements. One viable option for the purposes of RE is *use cases* since they satisfy the need for sets of attributes and outcomes and are "primarily an approach to discovering requirements from a user-centred viewpoint" (Rumbaugh 1994, p.23). They could be used in conjunction with RDR or as direct input into the decision table and maintenance would consist of modification to the use cases. So that parts of the system are not forgotten, the actors, which are external agents that require services from the system, are enumerated followed by the use cases. Specific values, not generalisations should be plugged into the cases so that thinking is grounded in precise examples. Rumbaugh suggests first building a system which contains the domain model and then the application model using use cases. The domain model and application model can be equated to the T-box and A-box, respectively. The process described in section 6.3.3 advocates the use of cases to build the A-box, the rules, from which the T-box, the concept hierarchy, is derived. The problem with use cases is that it may be hard to extract the relevant details to go into the case. There may also be too much information extracted to be included in a formal context which would make comparisons using the line diagrams difficult (Shaw, personal communication). Another possibility is the use of the repertory grid technique to acquire specifications from different viewpoints which could be used as cases for RDR. Alternatively repertory grids could be used directly as input into Phase Two and easily converted to crosstables. The process described in Phases Three to Five can then proceed as outlined in this chapter. Phase One would consist of development and maintenance of individual repertory grids.

Prior work in this area has been presented as verification or maintenance research (Compton et al 1992, Richards and Compton 1997a). Richards and Menzies (1997) claim that an RE technique could be viewed as a technique for knowledge maintenance (KM) and V&V. Menzies (1997b) argues that knowledge management techniques amount to a small number of activities processing less than a dozen types of knowledge. At different points of the software lifecycle, the emphasis is on certain activities processing some of the knowledge types:

- At later stages of the life cycle, the process of fixing inconsistencies is called "maintenance" or "validation". The essential difference between requirements engineering and maintenance or validation is that more behavioural knowledge is available (e.g. libraries of previously successful runs of the system). That is, $|X|$ is larger for (V&V and KM) than RE and $|X_{RE}| << |X_{KM}|$. With behavioural knowledge, it is possible to assess proposed KB revisions via running again all old cases.
- At the initial stages of the life cycle, the "inconsistency detection" and "fix" activities process the knowledge types of domain assertions and problem solving methods. Typically, at this early stage, there is little "behavioural knowledge" describing known or desired behaviour of the system. Requirements engineering, then, is the process of fixing inconsistencies in a system which has yet to execute. Without behavioural knowledge, it is necessary to rely on inspection tools that allow the expert to reflect over their knowledge. This chapter has assumed limited behavioural knowledge and has hence focused on inspection tools. If actual cases are available, which contain the behavioural knowledge, then we are able to avoid much of the conflict and develop a specification from these cases possibly using the RDR KA technique.

The only difference between MCRDR/FCA for requirements engineering and MCRDR/FCA for maintenance is that one more revision assessment operator can be used in the maintenance case (regression analysis over behavioural knowledge). Richards and Menzies (1998) speculate that this work is a possible first step towards a succinct toolkit to support knowledge engineering and software engineering activities right across the life cycle.

6.6. Future RE Work

A preliminary exploration of the RE framework has been defined and parts of the toolkit in operation have been demonstrated. Clearly, the next stage is full implementation of this approach. Such an implementation would have to handle a variety of interface details (e.g. appropriate displays of large concept lattices via such techniques as scrolling or zooming) and could be an extension of the existing MCDRD/FCA tool.

The approach aims to minimise the conceptual distance between A-boxes from different experts. The decision to accept that some conflict will always remain means that we need an evaluation and termination strategy that lets us test whether the framework is reducing the conflict and when enough conflict has been resolved to allow the specification to be used. As demonstrated in Sections 6.3.6 and 6.4.6 it is desirable to track over-time whether with each subsequent cycle there is a move towards consensus and these statistics should be computed when the resolution strategies for the current cycle have been applied. However, the statistics used were easily computed because it was known which concepts to use for comparison. As pointed out in Section 6.3.6, it we want to compute an overall degree of conflict within each KBS it may be harder to determine what concepts should be compared and how these scores should be used. A graph-theoretic distance measure between T-box lattices is proposed.

6.7. Further Discussion

This chapter has argued for a novel view of RE. Standard RE is an early-software lifecycle issue. The viewpoint resolution technique discussed here can be performed whenever there are some A-box (rules) and cases. Initially, cases will be hypothetical and the rules sets small (snippets of known business processes). Finally, cases will be "live" data and rule sets will be large. Large KBS can be handled in this framework by narrowing the focus of attention and reducing the context. In either case the technique can be applied. That is, the "RE Tool" can be applied right throughout the system development life cycle.

The RE framework described is a small extension to the KE approach described in Chapter Five. RDR and FCA were used as subroutines within the RE system. For the example given, the only additional code needed for RE was about 100 extra lines of Visual Basic code added to a system that uses 2,500 - 3,000 lines of C code and 1,500 - 2,000 lines of Visual Basic code for the interface. I envisage that full implementation would require no more than a total of 500 extra lines. Thus, the RE extension to a KA tool will be a mere 10% extension (about 500 lines on top of 5000 existing lines).

FCA was used to build explanatory T-boxes from performance A-boxes. This approach is applicable to any representation which can be mapped into a decision table. However, during the discussion on resolution strategies (Figure 6.4), it was noted that certain representations offered advantages. For example, when adding an attribute in a standard propositional rule base, the effects of this addition had to be checked all over the KBS. Such a check comes for free in RDR.

6.8 Chapter Summary

In this chapter it has been shown that by using the MCRDR/FCA tool it is possible to develop, compare and combine viewpoints from multiple stakeholders or sources. The RE framework provided offers more than just a comparison tool but included a number of operators which were extensions to MCRDR/FCA to handle the resolution of conflict. This has involved a change in some of the modification strategies offered by MCRDR as it describes a technique for maintaining the assertional KBS that is driven by the changes needed to the terminological KBS and also supports the removal of terminological knowledge, which is not possible in the assertional KBS.

To instantiate the RE framework the SISYPHUS III data has been used. It was not the goal of this chapter to evaluate the accuracy of the KBS developed or their usefulness for classification or instruction which was the aim of the SISYPHUS III project. However, it has been shown that the MCRDR KBSs, which include the individual as well as the combined viewpoints, are executable and it is claimed that the concept lattices could act as a tutoring tool in helping the user learn about the domain. The goal of this chapter has been to address an issue not stated explicitly in the problem statement for SISYPHUS III and that is how to manage multiple and conflicting sources of expertise. Measures have been provided that show the resolution strategies can reduce the degree of conflict between multiple stakeholders. In tackling this problem, this approach has also addressed a drawback of standard RDR. RDR systems have been shown to be useful for single expert knowledge acquisition. In such a situation, RDR offers a good performance module, but a poor explanation module. However, in the case of multiple experts, an explanation system is required since experts must trade off their competing views. FCA allows us to build an explanatory T-Box from an A-box initialised by RDR.

This chapter has opened up a new range of activities that may be supported by the MCRDR/FCA tool beyond those considered necessary for KBS based on a single source of expertise. This does not mean that only one expert was involved but that only one viewpoint, which may have used a combination of expert sources, was captured in the KBS. In the next chapter we return to consider the types of activities an individual will be interested in performing. The various requirements and features of the different activities are explained together with a description of how the MCRDR/FCA system is able to support the range of activities.

Chapter 7

7 Activity Reuses

The major goal of this thesis has been to see if it was possible to capture knowledge for one purpose and reuse that knowledge for different activities without the need to capture multiple copies of the knowledge tailored to each usage. It was found that the key to supporting different uses of the knowledge was a way of capturing or deriving abstracted knowledge, the relationships between the various levels of concepts and structuring these. Through the use of FCA it has been possible to take the simpler approach to derive, rather than capture, this sort of information from the behavioural knowledge that is relatively easily acquired using the MCRDR technique. This chapter is concerned with demonstrating how MCRDR/FCA functions as a complete system that supports a wide range of activities. It is not claimed that the range of activities included is exhaustive but that the range indicates that most common purposes to which knowledge can be put could be supported.

Firstly, the issues that affect which activity is likely to be selected by the user are briefly given followed by a review of each of the activities and how they are supported in RDR systems[1], particularly MCRDR/FCA.

7.1 The Issues Affecting the Activity Selected.

The choice of activity will depend greatly on the situation and the users involved. Factors to be considered include the intentions, expectations and pre-existing knowledge of users. Pollack, Hirschberg and Webber (1982) see interaction as a "negotiation process". Salle and Hunter (1990) make the following observations. Users want to discuss, beliefs and justifications. Novices and experts have preconceptions which affect how they perceive the system. ES may need to detect the novices motivation, goals and strategies as is done by experts. Users are not always clear in defining goals and may actively try to hide them. MCRDR/FCA does not try to detect such things as beliefs, intentions, motivation, goals and strategies but caters for these differences between individuals and groups by providing a range of tools and ways of accessing the knowledge from which the user can select. This is seen to be more in keeping with a

[1] The discussion in this chapter on RDR systems in general is an expanded version of the paper "Richards, D and Compton, C (1996) Building Knowledge Based Systems that Match the Decision Situation Using Ripple Down Rules, *Intelligent Decision Support IDS'96* 9th September, 1996, Monash University, 114-126.

situated view of knowledge and the need for ES to provide feedback and user control, than a system that controls the interaction based on detected features of the user.

The following is a list from Salle and Hunter (1990, p.10-11) of the types of questions a system should be able to answer in addition to the traditional questions "What is the fault ?" and "what is the remedy ?".

1. What would happen if ....?
2. Why would X happen.....?
3. How could X be prevented.....?
4. What are the critical factors in X.....?
5. What other solutions would work....?
6. Isn't solution X as effective.....?
7. Which is the best remedy....?
8. Is X the right remedy.....?
9. Why did remedy X work.....?
10. Why didn't remedy X work....?

An inferencing or consultation system will answer question 7. A good explanation system should handle all the "why" questions (numbers 2, 9 and 10). A causal modeling system is concerned with questions 1,2, 3 and 4. Causal modeling could also be adapted to answer questions 9 and 10 if the remedies were made part of the system or a fault and a remedy model were developed. Questions 1, 5 and 6 are addressed by a ‘what-if’ exploration system. Critiquing systems answer question 8. A teaching system should let the user seek out the knowledge needed to answer the other questions which relate to alternatives and choices and improved understanding of the domain.

In this thesis, the activities have been broadly classified into one or a combination of two main types of modes of use: reflexive and reflective which can be seen as similar to the distinctions made in Chapter Six between a performance and explanation system. Using the RDR paradigm, consultation or inferencing, knowledge acquisition, maintenance and validation form the reflexive modes of activity. Critiquing, causal modelling, explanation, tutoring, ‘what-if’ analysis, hypothesis testing and student modelling form the reflective modes of activity. In first generation ES there appears to have been greater success with the reflexive modes. First generation ES were weak in supporting the reflective activities and revealed the inability of ES to adapt well to changed usage. Since the existing RDR approach to the reflexive modes has been demonstrated in other studies, such as (Edwards et al 1993), these activities were not changed in this study although the KA and validation activities have been enhanced by being provided in reflective modes as well. Also, since progress has been slowest in the reflective modes, the focus of this thesis is on this area.

The distinction between some activities is not clear and will depend on the situation. For example an expert and student may perform the same actions but be performing the explanation or tutoring activity, respectively. Consultation, 'what-if' analysis and critiquing can be used as inferencing activities, but the latter two can also be used for explanation. Causal modeling is treated as an activity in its own right but may also be used for explanation or tutoring. Despite the possible overlap of activities the main feature of each reuse will be given below together with a synopsis of how RDR handles each activity.

7.2 Reflexive Modes

It was noted in Section 1.2 that most human, and particularly expert, action is reflexive. Rather than building or using abstractions and reflectively solving a given problem, experts tend to respond immediately with a solution. When reflection or in-depth analysis of the problem is performed it is often because the problem is stretching the expert to apply their knowledge beyond their current confidence or competence levels. In response to the reflexive nature of most human expertise, RDR were designed to provide simple and intuitive HCI which were introduced in Section 3.2.1 and are described further in this chapter in Sections 7.2.1 to 7.2.3. We look now at the reflective modes of consultation, KA, maintenance and validation.

7.2.1 Consultation/Inferencing

The most common usage of ES is to support the consultation activity, where the system poses questions to the user and produces a recommendation at the end. As Kang (1995) points out, many systems of this style failed to reach routine use because reliance on the user for input was difficult due to restrictions on the user's time and/or the user's inability or unwillingness to provide the answers. In contrast, initial RDR systems received input from information systems (IS) and output a report, which was then checked by the expert. In the main current implementations, the set of input cases may be historical cases, computer-generated cases or the user may enter hypothetical cases. However, the main role of the user is not to provide input but to evaluate the classification assigned by the system, see Figure 7.1.

An experimental technique more like the traditional consultation mode and designed to get the system to ask for more information, known as Interactive (I)RDR, was developed. IRDR used special rules with conclusions of the form "What is the value for rainfall ?" and rule conditions which had the value "unavailable". A more sophisticated system known as Error Recovery System (ERS) has been developed to support a help desk application (Kang et al 1997). This involves multiple inference cycles using a propose and revise strategy (Zdrahal and Motta 1995) and explicit rules that prompt the user for data. Kang (1995) proposed that the question rules be placed at the top level in an MCRDR tree and the questions be ranked and ordered according to whether they were satisfied by data in a previous inference cycle.

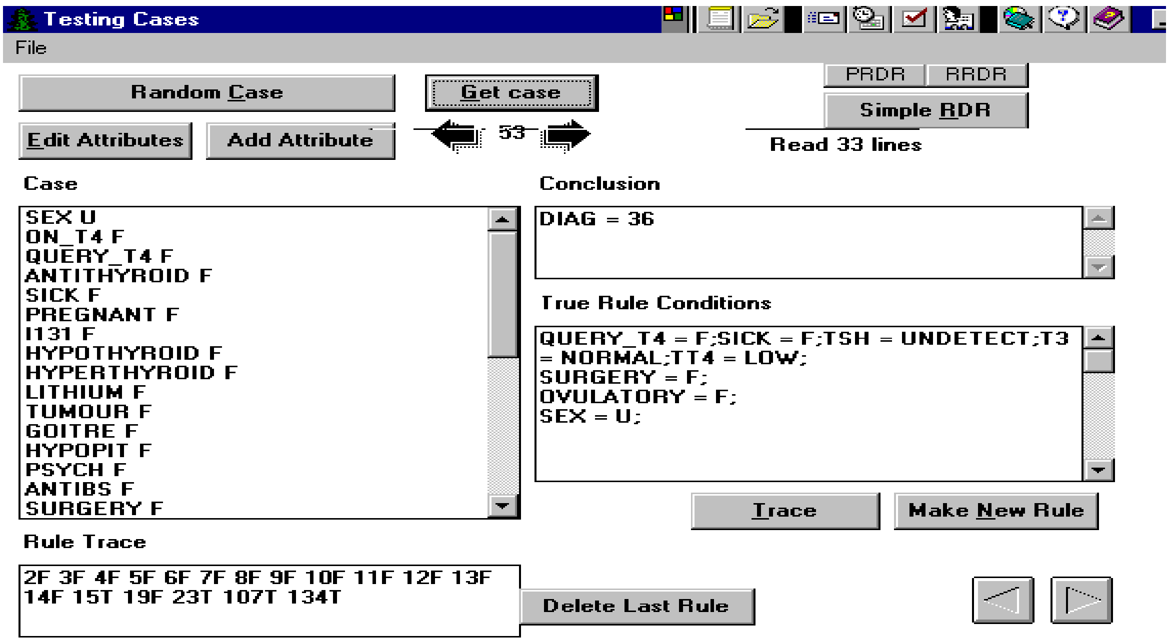


Figure 7.1. An inferencing episode using XRDR for Windows. A case is retrieved and assigned a conclusion. The user decides whether to change the conclusion by adding a new rule.

More recently a system has been developed which allows the user to specify which attributes should be prompted for when a particular attribute-value pair has been found. This system was briefly mentioned in 3.1.3, where the proposal to use a ML technique to determine in real-time which attributes were relevant was also put forward. The KADS solution to the SISYPHUS III problem (Jansen, Schreiber and Weilinga 1998) offered three possible ways of deciding which attribute to select. These methods include: random selection which may result in the user being asked the value for a large number of irrelevant attributes; ordering the attributes which is similar to the method implemented above and the use of an information-gain algorithm to determine the most informative attribute. The KADS submission employed all three option and used an information-theoretic algorithm similar to ID3 (Quinlan 1988) for the third option.

Previously the traditional consultation system-style has not been appropriate to the domains in which RDR has been used. An exception is the Lotus agricultural domain where a questionnaire screen as shown in Figure 7.2 was developed to provide a consultation style interface. However, to expand the suitability of RDR to a wider range of domains the typical "prompt user for input, output a recommendation" activity will be necessary but further work is postponed until the need becomes application driven.

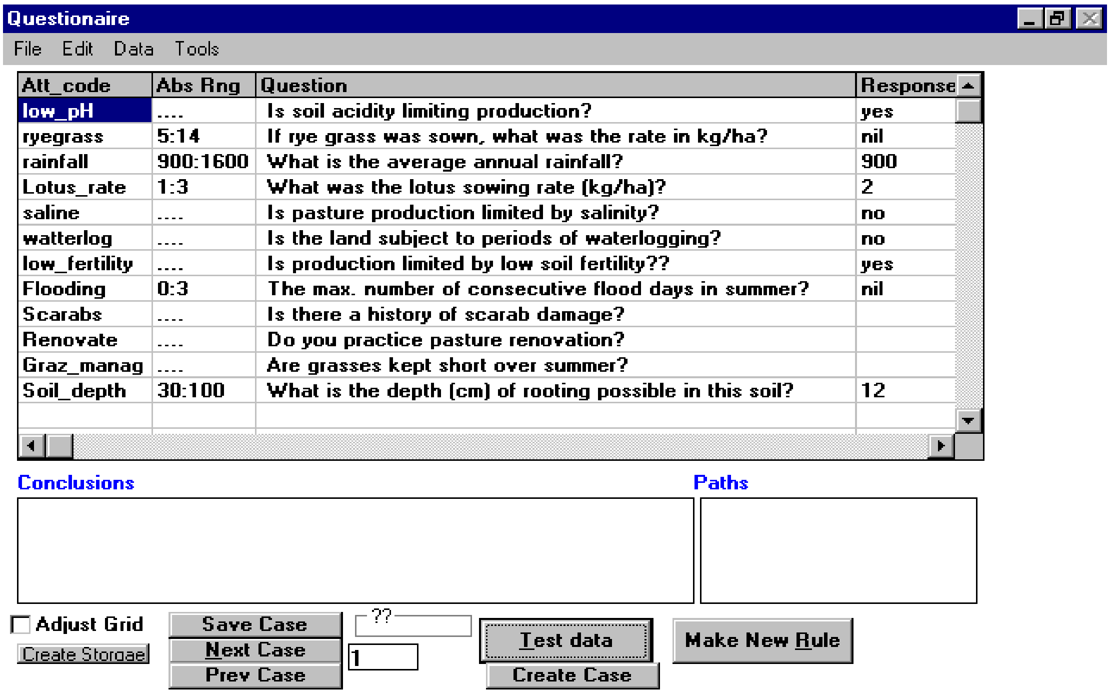


Figure 7.2: The Questionnaire Screen in MCRDR which was developed for the Lotus Agricultural Domain is an example of a consultation style system interface provided by RDR.

7.2.2 Knowledge Acquisition

In many ES, knowledge acquisition is a separate task from use of the knowledge base for inferencing. Puerta et al (1992) point out that this aspect is often forgotten and the concentration on reuse of problem solving methods has overshadowed the need for reuse of knowledge acquisition tools for the same knowledge domain. KA using RDR was described in Section 3.2.1 and is depicted in Figure 7.3. In the case of KA, there is no clear distinction between the building of the system and initial knowledge acquisition, the maintenance phase and even use of the system for reasoning. All three activities are performed as part of the KA process, although it is possible to perform inferencing as an activity on its own for the purposes of consultation as described in the previous section. As was shown in PEIRS, an RDR system can be set up quickly with minimal assistance from a knowledge engineer and then maintained completely by the expert. Since the same system is used for KA and inferencing it allows validation at KA time by performing an inference on the case which prompted the new rule.

Typically, one thinks of KA as a means to an end rather than an end in itself. It has been found that due to the ease of KA using RDR that KA itself has become the goal and has helped to capture and develop evolving knowledge in a number of applications covering agricultural (Hochman et al 1996) and medical domains (Edwards 1996, Feldman, Compton and Smythe 1989). The ability to uncover higher-level models using FCA from operational knowledge in an RDR KBS is particularly useful in domains where knowledge is emerging or in the common situation where it is difficult for experts to describe how they arrive at a conclusion. Also by viewing the knowledge in a number of ways and exploring the relationships that exist between the various concepts, the benefits of KA can be a significant source of new understanding of a field. In traditional systems where the domain and/or PSM models need to be described before KA can commence such an innovative use is not as easily possible. One of the main benefits of RDR is the ability to quickly build systems based on cases and learn about your domain without high resource requirements or commitment to get it right the first time. As noted in 3.3.1 this feature is shared with systems built on PCP such as Knowledge Support System Zero (KSS0) (Gaines and Shaw 1989) and Expertise Transfer System (ETS) (Boose et al 1989).

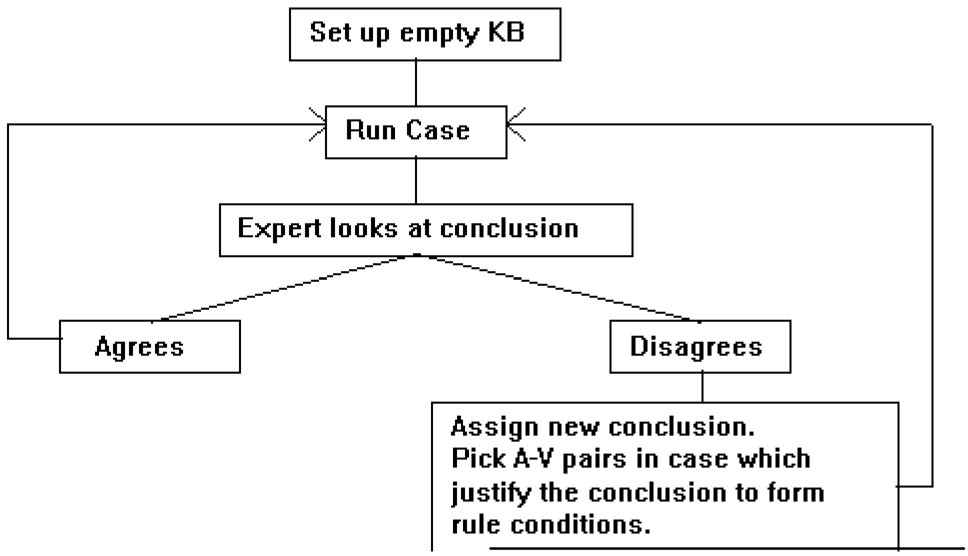


Figure 7.3: The KA process in MCRDR/FCA.

Some work has been done on providing KA as a critiquing activity that both critiques the proposed conclusion and the proposed rule. This is described in Section 7.3.1 on critiquing. The ability to perform KA in a reflective mode may result in a conflict in the goals of the system. KA using RDR is designed to be simple. Allowing multiple ways in which KA can be performed adds complexity to the KA process. This is true but in defense of offering the user KA in an alternative mode, the user can still perform KA in a reflexive manner as before but they now have the option of getting a second opinion if they wish to reflect on the conclusion and conditions for the rule they are proposing to add.

7.2.3 Maintenance and Validation

As described in Section 3.1, RDR were developed to handle the typical maintenance problems which occur with ES. RDR offer incremental maintenance and on-line validation. Maintenance and validation are not separate tasks in RDR but are part of inferencing and KA so the discussion of these activities has already been covered. Support for validation has been strengthened by allowing the user to perform KA as a critiquing activity and is described more in Section 7.3.1.

7.2.4 The Reflexive Modes in Action

To give the reader a feel for the reflexive modes of inferencing, KA and maintenance the screens that are used in MCRDR/FCA and how they are used are now given. The lack of distinction between these activities will be apparent as they are all part of the same process. These screens are virtually the same as in the current MCRDR for Windows implementation developed by Phil Preston and the MCRDR/FCA system is based on that code. I have added some extra options to some screens for the reflexive modes discussed in the following section.

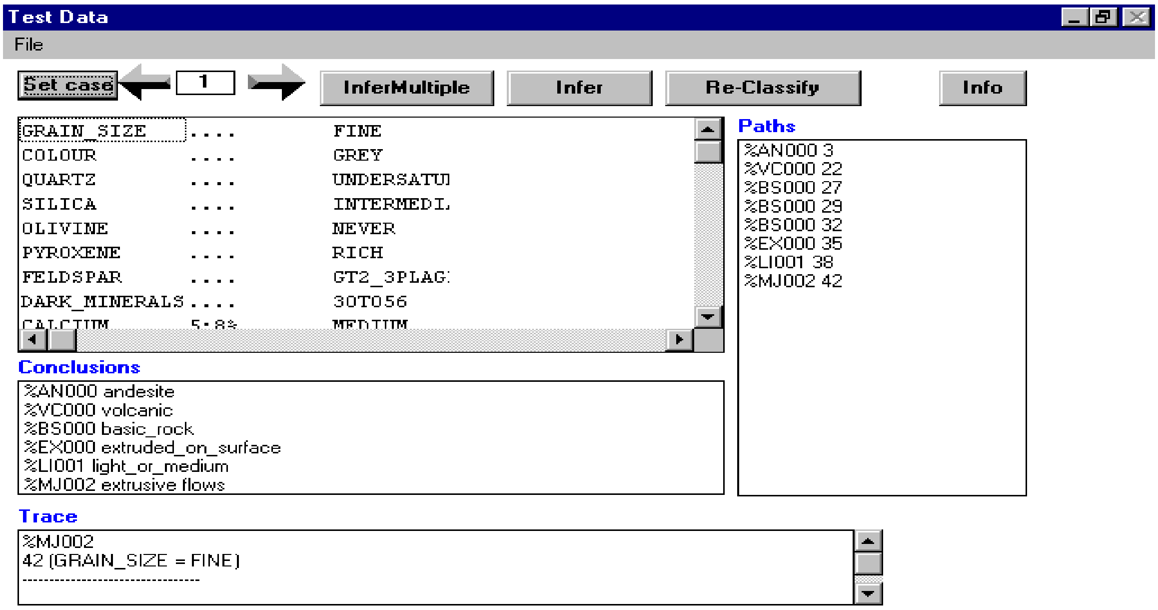


Figure 7.4: The Test Screen in MCRDR/FCA which is used for inferencing.

After the user has specified the KB and case file they wish to work with they are taken to the screen in Figure 7.4 which performs an inference on the last case accessed for this KB. The user may select another case to review if this is not the case in which they are interested. The user is shown the conclusion codes and a brief description of those codes in the conclusions list box. The codes and the numbers of the rules that fired are shown in the Paths list box. A rule trace that includes the conclusion, rule number and the rule conditions is shown in the Trace list box. The user considers the conclusions assigned to the case by the system and if they agree they select another case to review. If they do not agree they select the RECLASSIFY button, which takes them to the screen shown in Figure 7.5.

On the Reclassify Screen in Figure 7.5 the user is again shown the current case and the list of conclusions. The user can double-click on an existing conclusion code to move it into the rules to be stopped, click add to put a new conclusion into the “want” list box or specify which new conclusion should be a refinement (an exception) for a stopped rule. If the ADD command button is chosen, the user is taken to the CONCLUSION screen in Figure 7.6.

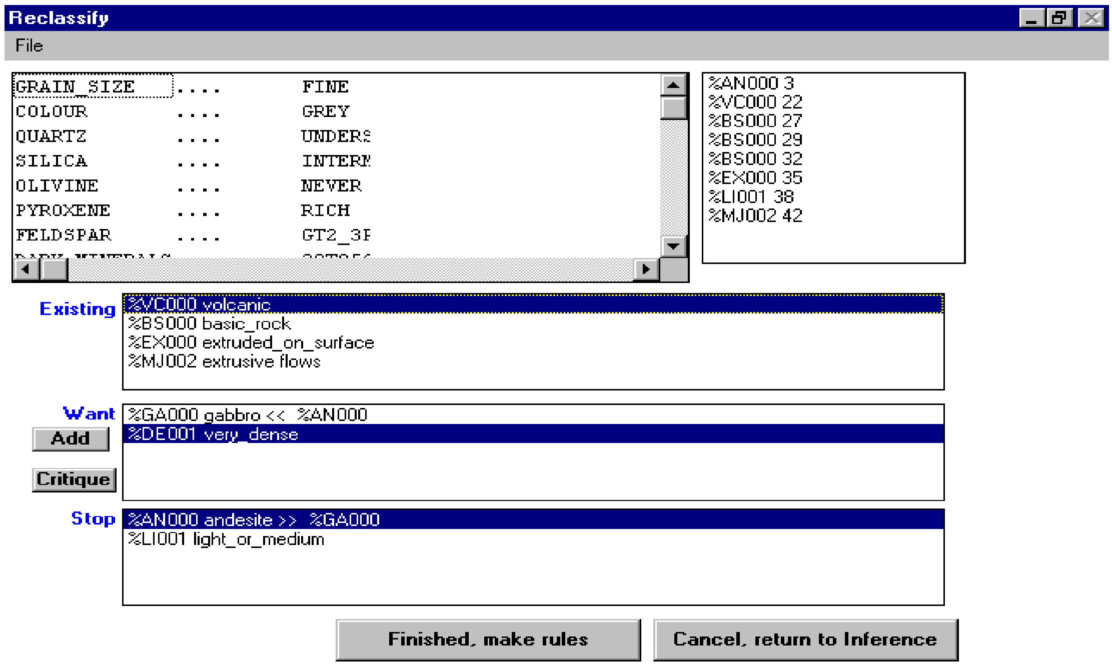


Figure 7.5: The MCRDR/FCA Reclassify screen that allows the user to specify which conclusions should be added, changed or stopped.

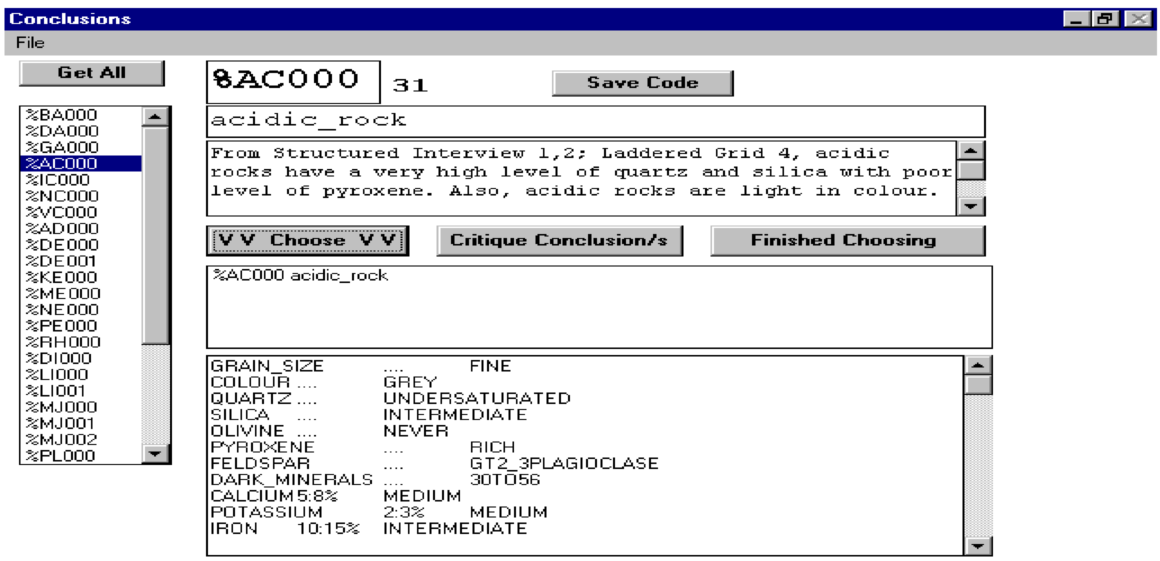


Figure 7.6: The MCRDR/FCA Conclusion screen for selecting a new conclusion.

The Conclusion Screen in Figure 7.6 is used to select which new conclusions to add for the current case. The user can pop-up a list of existing conclusion codes. When a code is selected from the list the short and long description are shown. The user can also add a new code by entering the new details into the code, short and long description sections on the screen. If a user wants to include a new or selected conclusion they click the CHOOSE button. When they have selected all the appropriate codes they take the FINISHED CHOOSING button which returns them to the Reclassify Screen in Figure 7.5.

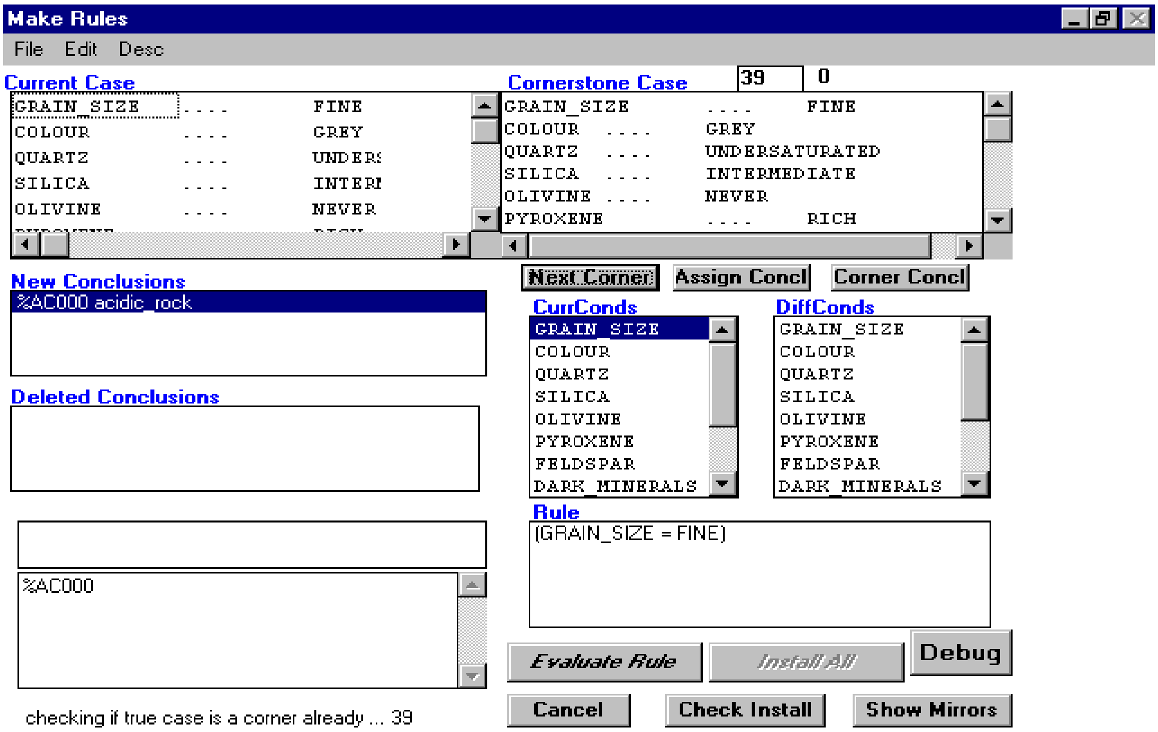


Figure 7.7: The MCRDR Make screen for entering new rules

In the Make Screen in Figure 7.7 the user first selects the conclusion for which they wish to make a new rule (remember multiple conclusions are possible). They then click on an attribute to be used in a condition for this rule. A pop-up window appears with a list of possible values for that attribute based on the differences between the current case and cornerstone cases. The cornerstone case being processed is shown in the window on the top right hand side. If the case is differentiated by the condition chosen the cornerstone list is traversed until a case is found that is not differentiated and it is put in the window. An additional condition must be added to differentiate this cornerstone case from the current case. When all cornerstone cases have been differentiated the INSTALL ALL button becomes active and the user can choose this option to add the rule. This may require patches in a number of locations. This process is repeated for each conclusion that was selected. When all conclusions have rules associated with them the user is returned to the Test screen in Figure 7.4 where they are shown the new set of conclusions for the current case. This provides immediate confirmation that the new rules have been added correctly. After checking the new list they may perform another inference on a different case and repeat the process outlined.

7.3 Reflective Modes

Just as humans need to reflect on their knowledge, there are times when the user will need to reflect on the knowledge in the KBS. In the following sections each reflective mode of use is described together with a summary of how RDR handles each activity. The reflective activities considered are critiquing, causal modelling, explanation, tutoring, ‘what-if’ analysis, hypothesis testing and student modelling.

7.3.1 Critiquing

Critiquing systems are an alternative approach to consultation style that may better suit certain domains and users. Salle and Hunter (1990) describe the critiquing system style as a system where the user can receive advice from the computer, have their own decision critiqued and be able to override the recommendation of the computer. The diagram in Figure 7.8 from Miller (1986) shows that a critiquing system differs from traditional ES that attempt to simulate an expert, in that it takes the physician’s plan as input in addition to the patient description (the current case) and outputs a critique of the plan rather than a recommendation. Miller claims that such an approach is often more suitable, especially in medical domains, where it is inappropriate for the system to tell the user what to do. Critiquing does not offer a “do this” generic solution but provides a way of supporting the expert and providing that support in a way that is based on the user’s thinking and style of practice.

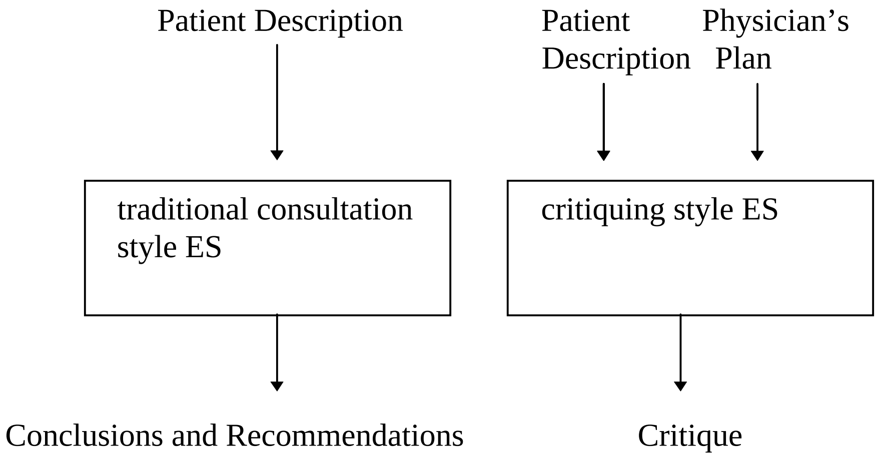


Figure 7.8: A comparison of consultation and critiquing ES
Adapted from Miller (1986)

Miller (1986) finds critiquing to be best suited to applications where there are a number of alternatives with different risks and benefits and where the knowledge is dynamic.

Woods (1987) sees the ability to provide such a facility as a means of assisting users to construct and test hypotheses by exploring the search space. This allows experimentation and adds "conceptualisation power" (Salle and Hunter 1990, p.13). Lehner and Kralj (1988) put forward the idea that the inconsistency that will result from a two expert problem solver, one human and one machine, will lead to better decision making. Miller (1986) sees critiquing as a way of helping the expert consider alternatives they may have missed and to keep up with current thinking in the field.

The benefits of critiquing style systems are not just limited to expert users. Studies by Kidd (1985) and Pollack, Hirschberg and Webber (1982) and Salle and Hunter (1990) all revealed that human to human interaction between experts and novices involved a negotiation process where the novice often proposes the solution, wants to know why something was recommended and may choose to reject the solution. Where this was not possible, interaction was found to be unsatisfactory (Coombs and Alty 1984).

Miller (1986) has implemented numerous prototype systems in a range of medical domains. Each prototype explores issues unique to its domain. The original system is ATTENDING which critiques anesthetic management plans. This system uses an Augmented Transition Network (ATN) but other prototypes use different architectures. A critiquing ES shell, E-ATTENDING, has been developed which has a data gathering component, production rule interpreter, expressive frames and a prose generator. A number of systems using this shell have been built. VA-ATTENDING critiques ventilator management plans and considers the use of strategic and tactical knowledge and how to resolve goal conflict. HV-ATTENDING looks at the pharmacological management of essential hypertension and is implemented as a *interactive paper* which allows an expert to disseminate their findings in a way that other physicians can access. PHEO-ATTENDING does not consider management but the medical area of *workup* which is more structured and constrained and looks at sequencing of tests and procedures to determine which diagnoses are correct. PHEO-ATTENDING addresses the issue of how to combine differing expert opinion. Conflict is resolved by breaking the problem into sub-problems and merging the related frame.

Another early critiquing system was ONCOCIN (Langlotz and Shortliffe 1983) which assists in the treatment and care of cancer patients. This system is of particular interest to this study because ONCOCIN is an example of how knowledge captured for one purpose may be reused for another purpose. The knowledge in ONCOCIN was not changed but a different system-style was offered to make the system more acceptable to users. The users, in this case oncologists, found it time consuming and annoying to have to perform a consultation with the system for each case. If the oncologist did not agree with the recommendation given by the system they had to override the conclusion and provide lengthy justifications for the difference, even if only minor. The adaptation to a critiquing system allowed the user to enter their own recommendation which the system would then analyse against its own recommendation using hierarchical plan analysis. Only variations of a prespecified size were reported by the system. The system became like another expert with which the expert could confer. This system differs from the system by Miller in that rather than actually critiquing the user's plans it explains it's own plan when it detects a mismatch. As emphasized by Langlotz and Shortliffe (1983) this approach demands a good explanation system to work and thus the findings are also relevant to explanation research. ONCOCIN also differs from the ATTENDING approach in the nature of the domain that was used. Oncology patient management can be handled by formalised oncology protocols. Miller classifies this domain as one that is based on objective criteria and therefore also an atypical medical domain.

More recent work that has followed on from ONCOCIN is the Therapy-Helper Project (T-Helper) (Lauderdale 1995) which has become part of the EON Project (Tu 1997). The Eon Project is concerned with protocol-based care. Online patient data and knowledge concerning protocols from a wide range of clinical information systems are used to determine when a protocol applies, to recommend a course of treatment and “to critique patient care that deviates from recommended practice patterns” (Tu 1997, p.1). These projects point out that critiquing requires reasoning and they are looking at developing new data models which contain the necessary temporal knowledge and new problem solving methods to assist in this reasoning process. T-Helper uses the Protégé-II methodology to configure the episodic skeletal-plan refinement (ESPR) method. The ESPR method is made up of three main subtasks which includes: using a *skeletal plan* to propose a *plan action*, problem identification and plan modification. Although a number of gaps in the methodolgy and tool kit are noted, T-Helper is an example of how Protégé-II can provide reusable ontologies.

Silverman (1992b) offers a survey of existing critiquing systems. Work in this area should shed light on the way decisions are made, the validity of decisions made by so-called experts and how decision making can be improved. Silverman (1992a) describes a system developed by the US army to critique decisions. A model of the human error process was created using social judgement theory and subjective decision theory. The two main causes of errors identified were missing knowledge and erroneous reasoning. He stressed the importance of judgement and substantive knowledge in the reasoning process and advocates the inclusion of deep knowledge.

To summarise the possible benefits and requirements of a critiquing system as offered in the various systems mentioned, a critiquing system:

- allows the user to enter the plan/recommendation,
- allows the user to test hypotheses and consider alternatives,
- the system critiques the user's plan or the system explains its own plan,
- the system provides a second expert opinion,
- allows the novice to negotiate a solution and
- includes deep and structured knowledge.

These points will be used to evaluate the critiquing activity offered in MCRDR/FCA which is what we look at next.

7.3.1.1 Critiquing in RDR

To initially test out the usefulness of RDR for critiquing, the screen in Figure 7.9 was added to the MCRDR for Windows interface that presented the user with a case and let the user select the appropriate conclusion/s for that case. The system then determined its own conclusions and compared the two recommendations. The user was notified whether the user's conclusions were in agreement or disagreement. If a conclusion entered by the user was not one of the system's conclusions the user was asked to enter a rule, which should represent the reasoning behind the conclusion. If there were conclusions offered by the system that the user did not agree with, the user could look at the explanation, an extended trace as discussed under explanation in 7.3.4.1, and add a new rule to modify the rule which gave the incorrect conclusion or add a stopping rule so that the conclusion would no longer be given in those circumstances. In a traditional ES such an approach would not be as easy because activities, such as KA, critiquing, maintenance or inferencing, may be in different systems or performed by different users.

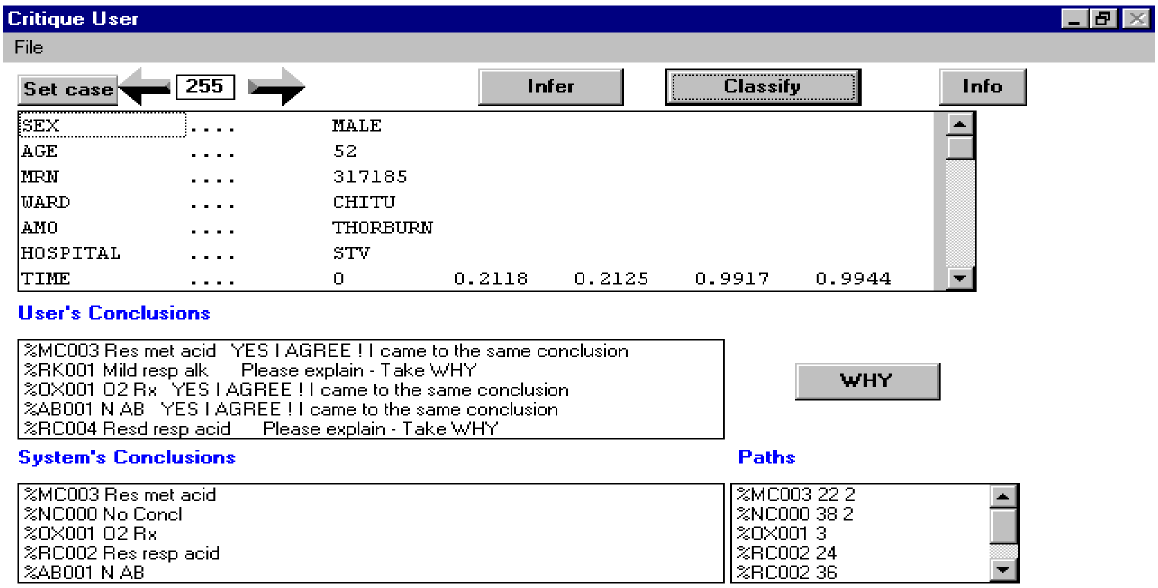


Figure 7.9: Critiquing Screen 1 in MCRDR/FCA.

Critiquing is offered at the conclusion and rule levels, in keeping with the KA approach offered by RDR. When a user decides to assign a new conclusion (the first step in developing a new rule), the user may select a conclusion on the RECLASSIFY screen and click the critique button to have the new conclusion evaluated against all other paths (rules) that gave the same conclusion. If those rules are inconsistent with the current case the user is notified of the anomaly. In Figure 7.10, using the data from the SISYPHUS III (Shadbolt 1996) experiments, I have attempted to add a rule that states that a particular rock, already classified as volcanic, should also be classified as plutonic. These conclusions are mutually exclusive, therefore it is inconsistent to assign both conclusions to the one case. In Figure 7.10, this inconsistency has been detected by the system. By showing the user the rules in conflict they can either change their conclusion, select an existing rule to add a refinement to, add a new rule higher in the tree or add a stopping rule to stop the conclusion being reported for this case (and any others with the same features).

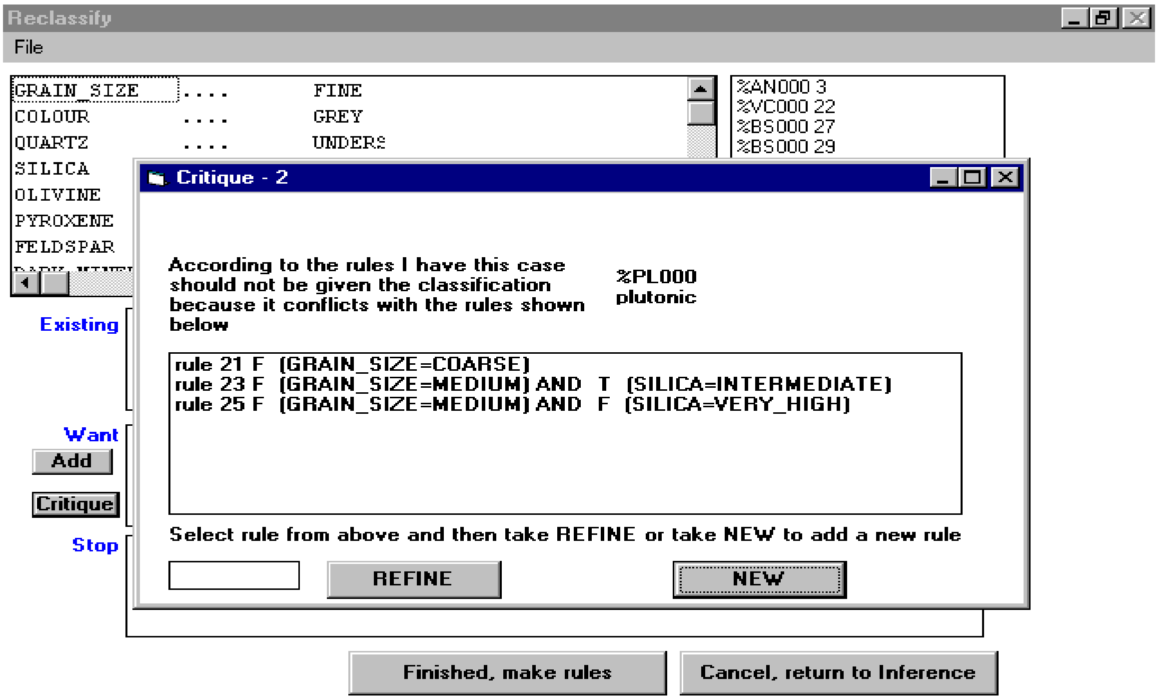


Figure 7.10: The critiquing screen 2 in MCRDR/FCA for Windows.
The user is shown any pathways that are inconsistent for the specified conclusion and current case.

Once a conclusion has been decided on, there is also assistance in forming the rule. When the user enters a new rule on the Make Screen they may click the EVALUATE RULE command button as shown in Figure 7.7 to take them to the Test Pathways screen shown in Figure 7.11. The proposed rule is placed in the Path 1 text box. The user is able to compare the proposed rule against existing rule pathways using the nearest neighbour algorithm described in 3.3.2.1 to provide a list of proximity scores with other rule pathways. Alternatively, they may chose to be shown where the new concept fits into the hierarchy of concepts derived using formal concept analysis. With this option the user is presented with a listbox of all the other pathways in the knowledge base that are matches, sub or superconcepts. If the new rule is identified with concepts that seem inappropriate this is a warning to the user that the knowledge in the new rule or an existing rule is incorrect. For this purpose only the intensional definition of the concepts derived using Wille's technique are used because as discussed in Section 3.3.1.3 the extensional definition is too restrictive when we are dealing with new concepts. It is feasible to compare two rules (objects) to see if they have similar properties but it is unlikely that the set of objects would be the same or similar since many objects will not yet have been defined.

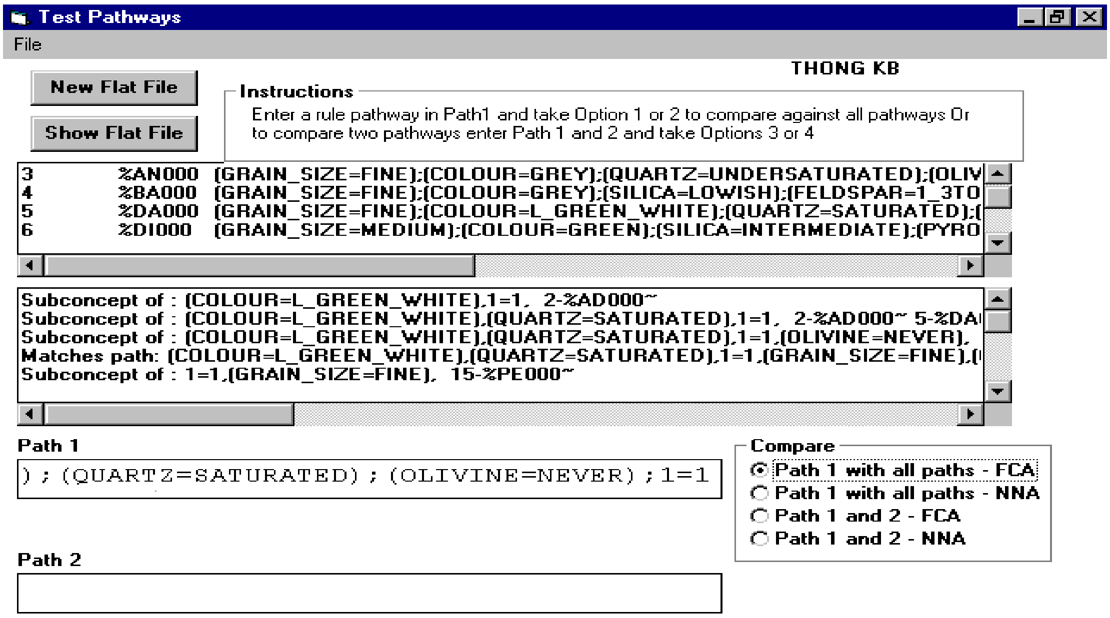


Figure 7.11: The Test Pathways Screen for comparing rule pathways in MCRDR/FCA.

If the user is satisfied with the new rule they return to the Make Screen in Figure 7.7 and take the INSTALL ALL button. Alternatively, the user may decide to modify the proposed rule or to modify a different existing rule that was identified as needing further refinement. In this way critiquing assists the user with KA and validation of the KBS.

The two main critiquing features in MCRDR/FCA are the conclusion critiquing screen (figure 7.10) and the test pathways screen (figure 7.11). Using the description of critiquing systems given earlier, let us see how MCRDR/FCA measures up.

- allows the user to enter the plan/recommendation. MCRDR/FCA allows the user to enter the conclusion (the recommendation) and see whether this conclusion is inconsistent for the current case based on other rules that reach the same conclusion. This allows the user to reconsider their recommendation. If they decide to keep that recommendation they can see what rule/s are in conflict and may need a refinement added to cater for the current case. Alternatively, the user may detect that the situation which the current case highlights has not already been identified and in this way missing knowledge may be identified and added.

The main limitation of this approach is that there may be other conclusions that are similar, but not identical, which would also reveal inconsistent or missing knowledge in the KBS. It is hoped that the next step where the proposed rule is critiqued will catch such situations. Another option is to allow the user to identify which conclusions are related, which could be stored in a table, and all related conclusions would be used in the conclusion critiquing screen. The use of conclusion families are a natural way of implementing such an approach.

- allows the user to test hypotheses and consider alternatives. The test pathways screen is a way of seeing what other rule pathways are similar to the proposed one. Each pathway represents an alternative. By considering the alternatives the user can propose new rules and in this way test out new hypotheses. Moving beyond the critiquing activity, the user can alter attribute values in the case to perform 'what-if' analysis or view parts of the knowledge base by selecting various foci of attention as alternative uses of the knowledge in which hypothesis testing and scenario analysis can be performed.
- the system critiques the user's plan or the system explains its own plan. The type of critiquing offered by MCRDR/FCA does not quite fall into either of these categories but offers aspects of both. The user's plan, in the form of a conclusion or rule conditions is used to determine which existing paths in the knowledge base may be of interest to the user. Figures 7.9 and 7.10 both provide feedback to the user on whether the conclusion proposed appears appropriate. The rule pathways provided in figure 7.9 and 7.10 form the explanation of why the system does not agree with the user's conclusion, thus the explanation is in terms of the system's plan, not the users.

  The main limitation of the type of critiquing offered by MCRDR/FCA concerns the nature of the explanations given. As described later in the explanation section, rule traces are often not terribly easily understood and a natural language interpretation would be more user friendly.
- system provides a second expert opinion. It is envisaged that critiquing in MCRDR/FCA would be used when the expert is unsure of the conclusion or rule conditions they are entering and wants to compare their own solution with those already found in the system. In this way the system becomes the second expert. If more than one expert is using and maintaining the system the ability to critique new rules will be particularly beneficial because the system will provide a repository of shared knowledge that each expert can consult. Without the ability to critique a

solution, inconsistencies are more likely to occur as it would be more difficult for one expert to determine what rules another expert had entered. The RE work described in the previous chapter could be employed to further strengthen this feature by annotating rules with the source of the rule and by using the FCA modelling tools to compare expert models.

- allows the novice to negotiate a solution. It was noted earlier that novices often prefer to be involved in the recommendation process. The test pathways screen was originally developed as a means of understanding the relationships between concepts (using the FCA option) and the closeness of concepts (using the NNA option). When the novice is given a recommendation they could use the rules that fired as the pathways which they want to compare to see alternative pathways that are similar. Based on the alternatives the novice can decide which recommendation they prefer.

  The novice is also able to use screen 7.9 as a way to test out their own solution. Critiquing would be particularly useful for teaching as the student could propose the solutions, without having been influenced by the system's conclusions, and then have the two compared. To improve support for the novice/student the full range of critiquing tools which allow a conclusion to be added or a rule to be proposed needs to be included in a combined critiquing-what-if activity that allows any changes made to be undone afterwards.

- includes deep and structured knowledge. The extra information critiquing offers adds value to the recommendation and adds a depth of knowledge not available in conventional MCRDR inferencing. The FCA option on the test pathways screen provides greater depth of knowledge by showing abstracted concepts not obvious directly from the rule pathways. The FCA option also provides the structure of the knowledge by showing the sub, super and matching concepts.

In conclusion, MCRDR/FCA allows the user to input their own recommendations and have the system either state that it agrees or show how the system would reason about the same case. The user is able to override the system's conclusion but is offered an explanation of the system's plan so that the user may contemplate whether they missed an important consideration. The MCRDR/FCA abstraction hierarchy provides some understanding of the relationships and structure of the knowledge while the nearest neighbour algorithm provides a measure of closeness between concepts. Without the addition of these two features to MCRDR it was not possible to offer critiquing beyond stating whether the system agreed or didn't agree. While critiquing is now satisfactorily supported further work on providing a more user-friendly presentation of the knowledge in the system should be provided to assist the user in evaluating the system's plan against its own.

7.3.2 Causal Modelling

In this section we look at the importance of causal modelling as an activity supported by KBS. Scott and Weckert (1995) consider causal explanation to be a means of offering justification by providing the link between the input from the user to the output of the system. This output may be the final conclusion - the what and how, or the reasoning behind the system asking a particular question - the why. White and Frederiksen (1989) found that causal models could be used as intelligent tutoring systems in the fields of science and engineering. This discovery was in response to evidence that many physics students could not solve qualitative problems and didn't really know how to apply the formulae they had been taught. Traditional approaches to problem solving had concentrated on quantitative methods where the variables in the problem needed to be matched to equations, similar to the use of heuristic pattern matching in ES. White and Frederikson (1989) hypothesized that if students were able to learn the domain in terms of qualitative causal models they would gain a more robust and thorough grasp of the underlying theory and its application. They saw the qualitative causal model as a better starting point for learning, which could later be improved by the introduction of more complex qualitative models and progress onto quantitative models. They argued that other research has shown that experts progress in their problem solving in a similar sequence. Their work is discussed further in Section 7.3.4.

A lot of interest has been generated in causal modelling in the Stanford "How Things Work (HTW): Knowledge-Based Modeling of Physical Devices" Project. This project aims to develop:

- *"representation techniques for encoding knowledge about engineered devices in a form that enables the knowledge to be used in multiple systems for multiple reasoning tasks;*
- *reasoning methods that enable the encoded knowledge to be effectively applied to the performance of the core engineering task of simulating and analyzing device behaviour" (Torsten_Heycke 1994).*

The first aim is clearly interested in the reuse of causal knowledge across tasks, domains and applications which is similar to the goal of this thesis except that this thesis is concerned with the reuse of more than causal knowledge alone. The project offers tools for simulating and analysing the behaviour of electromechanical devices to support design and redesign of such devices. The system being developed is known as the Device Modeling Environment (DME) which they refer to as a "designer's associate". As part of this project a common Compositional Modeling Language (CML) and the Causal Functional Representation Language (CFRL) (Vescovi et al 1993) have been developed. CFRL is used to define the intended device functionality. As was found in the HTW Project, causal modelling may require special knowledge representations and processing. Causal modelling may also require the knowledge to be presentated in a different way to conventional text explanations. Lehner and Kralj (1988) began their research with the theory that people evaluate the worth of a decision based on experiences they take with them. People expect to see what they are used to. This involves a complex model of causations. They performed an experiment to test cognitive consistency and modelling. They found that users see the interface as causal evidence of how an ES is internally working. They argued that the user interface gives cues to the user - such as covariance, continuity, temporal order and similarity. Following their line of reasoning, it is important to present knowledge to users in a way that fits their thinking about the subject. Explanation is typically thought of as words describing why a question has been asked or why a conclusion has been made. Users' expectations and backgrounds together with the user interface will affect the way they are able to use a system (Lehner and Kralj 1988). In many professions such as medicine and electrical engineering a diagram is a more natural way to consider causality. The explanations given by Lee and Compton (1995) are in such a form and are known as a CMOD model. Referring to a graph which showed how the final conclusion fitted in with the known findings Clancey comments: "for the purpose of teaching, this graph could perhaps be the best way to reify the process of diagnosis" (Clancey 1986, p.56).

Get a rule from rule base
Extract all items and conclusion from the rule
For each object, test causal links from the rules
If causal link is not found, ask expert to add
Causal effect (+/-signs) computed for each link
Draw a CMOD model

Figure 7.12 Developing a causal model from heuristic rules (Adapted from Lee & Compton 1995).

Lee and Compton (1995) have reused heuristic knowledge so that it provides causal explanations. They argue that causal models provide more useful explanations, particularly for learning, than those missing causal links but such explanations are difficult to construct. While it is easy for the user to assign a conclusion and enter rules that differentiate one case from another it is exceedingly more difficult for users to explain the underlying theories required to enter causal models. The work done by Lee and Compton takes the heuristic model and uses an algorithm, summarised in figure 7.12, which seeks to make explicit the causal links implicit in the heuristic rules. It has been found that it is not possible to determine all links and some collaboration with the expert to determine causation has been used. The model is also useful for validation as it is checked for consistency each time a heuristic rule is added. If links are missing or wrong the model is modified automatically or with user assistance.

Webster (1995) also looked at the relationship between models and heuristics and whether the causal links that Lee (1996) was deriving, often using expert opinion, could be derived by some form of machine learning. He investigated several ML techniques including AQ, C4.5, FOIL and MIS. The propositional learners, AQ and C4.5 were found to be unsuitable because they are aimed at finding minimal solutions to solve a classification problem. A rule such as IF a, b and c THEN d will provide a heuristic but does not give us any real idea about causality. The Inductive Logic Programming (ILP) techniques such as FOIL(Quinlan 1990) or MIS (Shapiro 1981) were also found to be inappropriate because they required positive and negative examples and background knowledge. The method that Webster used to derive causal models from heuristic rules hinged upon using the conclusion as the cause and condition attributes as the effect. The direction and sign were determined for each rule. Attributes that had a normal value or were in a state of equilibrium were ignored. This simplistic approach worked were there was only one pathway from cause to effect. Where there was more than one pathway the ambiguity needed to be clarified to determine the actual causes. This was done using the concepts of transitive and euclidean links. The assumption was that if a transitive or euclidean link exists then that is evidence of ambiguity. Where ambiguity was detected the expert was asked about the links and was able to forbid or confirm a link. This approach was tested with rules from the Garvan KBS and another KBS developed for the health fitness domain. It was found that just querying the transitive links worked reasonably well. This system was greatly limited by its formatting requirements so that it could be input into prolog. Webster’s work has concluded that machine learning alone is not sufficient and that a collaborative approach using ML and expert input is required. This is in keeping with the findings by (Shiraz and Sammut 1998, Webb and Wells 1995) where an expert interacts with a machine learning system in the KA process.

Mahadidia (1994) has developed an ILP technique that he calls 'unfolding learning' to derive causal models from rules not in Horn Clause form such as data in A-V pairs. He discusses a system called JUSTIN which derives causal explanations in neuroendocrinology. JUSTINand the solutions proposed above by Lee and Compton (1995) and Webster(1995) all rely on expert involvement to resolve conflicts in the causal models they derive. Unfolding learning has been developed to overcome this limitation and to deal with non-Horn clause logic. B $\wedge$ Bar Effects are used to generate negative clauses (U_{ni}), B $\wedge$ Causes are used to generate positive unit clauses (l_i), where B is the background knowledge of an existing causal qualitative model in the form of a horn-clause logic program. Each negative clause is combined with one or more positive clauses to form an hypothesis. Different suitability criteria, such as adding the minimal number of clauses, can be applied to restrict the number of hypotheses. In this way a too specific theory is generalised in keeping with Shapiro's MIS framework (Shapiro 1981).

Causal modelling has not been investigated in great depth in this thesis. This is partly because the PhD work of Lee (1996) has addressed this knowledge usage and partly because this activity is not applicable to all domains. However, the line diagrams produced using FCA are similar to the connected graphs to which Clancey (1986) refers. As is shown in the dialogue presented with the line diagram in Figure 8.21 for the Blood Gases Domain (where the domain can be described as causal) it is possible to infer that certain conditions are positively or negatively causally related to one another. For example at one node we may have a particular condition occurs when BLOOD-PC02 is HIGH and BLOOD_BIC is LOW. At another node we may have BLOOD_PC02 is LOW and BLOOD_BIC is HIGH which results in the opposite conclusion. It would appear that the two attributes are inversely related and that when one is elevated the other goes down. Line diagrams and typical causal diagrams can both represent the ideas of sources, sinks and flows with directional lines but the line diagram does not include the "+" and "-" annotations which indicate the nature of the relationship. However, since each attribute is actually an A-V pair indicating the value of the attribute it can be determined if the flow is increasing or decreasing. To make the research in this thesis more complete it would have been interesting to implement Lee's algorithm and to look at possible modifications to the FCA line diagram to support causal relationships such as specifying the nature of a link. However, this is not a small task. As Buchanan and Shortliffe (1984) point out, providing an explanation of causal knowledge requires more than stating the rule, which can suffice for other types of knowledge. Thus, on the one hand it can be argued that causal modelling may be performed to some extent in MCRDR/FCA using the line diagrams. On the other hand, there has not been sufficient investigation of this activity and its inclusion into MCRDR/FCA to claim that causal modelling is adequately supported and achieving this is left to further research.

7.3.3 Explanation[2]

Unlike causal modelling which is not applicable to all domains, explanation is an activity that must be supported in all KBS. ES need to offer explanations because of imprecise domains and the use of heuristics (Lee and Compton 1995). Verification is not enough. ES need to justify and be accountable (Swartout and Moore 1993). Through explanation the user can see if the recommendation is valid. Explanation is a driving force in second generation systems. First generation ES tended to paraphrase rules. This

---

[2] Much of sections 7.3.3 and 7.3.4 are included in a paper submitted to PKAW'98 titled "A New Look at Explanation for Knowledge-Based Systems".

meant some of the important parts of the knowledge were left out and users found the rule trace format unnatural. There was generally no distinction between knowledge roles (Clancey 1984 was an exception). Some rules may have been used to handle the user interface or control inferencing but they were still used in the trace given to the user. The problem of unnatural dialogues required better technologies for explanation generation. Second generation ES explanation focuses on context, goals and actions, methods and justification (Swartout and Moore 1993).

There has been much criticism of the usefulness of the rule-trace style explanation. Chandrasekaran, Tanner and Josephson (1988) expound the need for control strategies and deep models in support of explanation. Expert systems aim to handle knowledge rather than simply data. They argue that most KR are at too a low level of abstraction to provide adequate explanation. Swartout and Moore (1993) agree that the problem of lost knowledge can be addressed by the knowledge representation.

Chandrasekaran, Tanner and Josephson (1988) describe 3 categories that cover what they consider to be the best ideas that have emerged on explanation in first generation ES. The first approach was embodied in the MYCIN ES where the user could ask WHY and receive a trace of rules. The second approach was developed through NEOMYCIN (Clancey 1986) which is able to give strategic explanations because the knowledge has been abstracted into different levels. These first two approaches do not provide justification. The third approach is found in Xplain (Swartout and Moore 1993) where the system details are input to an automatic programmer which generates the explanation. Through the use of the automatic programmer the relevant and related knowledge is used to build the rule base. A trace is kept which stores the initial knowledge which may be later retrieved for explanation purposes. Chandrasekaran, Tanner and Josephson (1988) define 3 possible types of explanation. Firstly, the system can explain how the raw data matches the goals. Secondly, justification may be given through the description of the KB contents itself. The third type is the control strategy explanation. They see the issue of problem solving types as necessary for explanation and suggest the use of generic tasks (GTs) to provide explanation types one and three. They summarise problems into four generic tasks: classification, state abstraction, knowledge directed information passing or design by plan selection and refinement. They claim that generic tasks act as higher level building blocks. Chandrasekaran, Tanner and Johnson state that there is much support in the explanation literature for providing an explanation and justification of the reasoning processes and PSM.

This claim has interesting implications for the current work in MCRDR where the representation and inferencing strategy that is used for classification has been used for a configuration task with the key difference being that the latter requires more knowledge to be supplied and additional inference cycles. Since the systems reasoning processes are the same for classification and configuration an explanation of this process does not appear to be terribly important. This finding is potentially good news for explanation research because it allows the effort to focus on explanation of domain knowledge. An argument explored further in Section 7.3.4.1 is whether it is necessary to understand the system's reasoning since we often don't understand or use each other's reasoning and since humans tend to offer justifications directed to the audience rather than explanations.

Swartout and Moore (1993) discuss five aspects of a good explanation:

1. Fidelity - how accurate is the representation ? Fidelity is aided by a simple inference engine.
2. Understandability - is the content and context understandable? The components to be considered are terminology, user sensitivity, abstraction into different levels, summarisation (the amount of detail offered in an explanation), the possible system perspectives, linguistic competence and feedback.
3. Sufficiency - is there enough knowledge to provide explanation in different contexts? This will require knowledge to be stored that allows explanation about the system's behaviour, justification, preferences, domain explanations and terminology definitions.
4. Low construction overhead - how time consuming and difficult is it to build explanations ?
5. Efficiency - what will the system's response time be like, will it take a long time to generate an explanation ?

These five aspects of a good explanation will be used as the basis of an evaluation of the RDR approach to explanation and is covered in 7.3.4.1.

Similar to the first generation ES approaches to explanation described by Chandrasekaran, Tanner and Johnson (1988), Swartout and Moore (1993) describe the four main architectures found in second generation ES explanation.

1. The approach offered by Clancey in NEOMYCIN used metarules for problem solving strategy and demonstrated how control knowledge could be abstracted.
2. Chandrasekaran offers another perspective through the use of GTs. The approach is founded on the premise that problem solving methods have similarities that can be grouped and abstracted. Each GT has its own language and its own explanation routines. Explanation nodes may be suspended, rejected or accepted. Backtracking is possible by returning to suspended nodes. The grouping of the knowledge is

beneficial and the use of a generic method improves coherence. The drawback of the approach is that GTs are highly tailored to a particular task which means they have high construction overheads and pose a fidelity risk because if the problem solving method is changed the explanation method must also be modified.

3. Swartout calls his technique Explainable Expert Systems (EES). The first step is to develop an abstract high-level specification to pass to the automatic programmer. The Design History keeps the decision steps and is used for explanation. The Interpreter executes the ES code. EES is able to supply the reasoning behind actions. The automatic programmer means links are automatically derived, unlike most approaches. In an EES KB three types of knowledge are held: terminological (using a KL-ONE type structure), domain model and problem solving knowledge. Reformulations are used to adjust goals if no plan is found. There are three types of reformulations: covering, redescription and individualistic. The argument behind EES is that in order to provide adequate justifications some of the knowledge available at acquisition time needs to be programmed into the system. Swartout claims greater depth of knowledge (because the “connective tissue” is kept), improved fidelity, sufficiency and efficiency (executable code is produced which doesn't have to be generated while running). The main drawback of EES is the increased overhead in building the inputs to the system.
4. The fourth method used is reconstructive explanation. This technique uses one KB for reasoning and another for explanation. The philosophy behind this is that to have understandable explanations you need to use the users terms. In the research by Wick and Thompson (1992) a trace is kept by the problem solving KB which is passed through a filter to the explanation KB. Such a system offers high understandability but also high overheads and poses a fidelity risk.

Feedback is a key role of explanation. If feedback is provided in the explanation strategy the problem can be negotiated between the system and the user. Using feedback it is possible to start with imprecise domain and user models because the user can follow the reasoning and ask further questions. It may also be useful to keep track of goals and what was understood by the user. Several communication strategies may be necessary in case one way is not understandable. This also necessitates recovery strategies.

Cawsey (1993) describes two main approaches to explanations. The first tailors the text to the user using a model of the user's knowledge and gives explanations as deemed necessary. The second approach lets the user ask follow-up questions, this feature is often in the form of asking ”why” when prompted for some input or “how” when a conclusion is given. A good model of the system and the user is needed for the first approach and is much more difficult and costly to implement. The locus of control is different with both approaches. The system takes responsibility for providing explanations in the first method. The downside of letting the user control when and what explanation is given is that the user may not know when and what to ask. This may be particularly a problem where the purpose of the explanation is teaching and the user does not know what to ask next. The current HCI view is to always give the user control (Kaplan and Rock 1995). This study has adopted this view and even though the user may not always know what they need to learn about or look at next they are probably in no worse position to make this judgement than the programmer who can not possibly predict the situation and the needs of each user.

Explanation is much more complex than choosing between the two approaches described above. Explanation can vary in the level of detail, technical terms and expressions, the way things are explained or presented and what material is included. Some have sought to simplify the possible combinations by creating one level of explanation for novices and another for experts. However, this is not as simple as it sounds. Cawsey (1993) has found that "there is no simple relation between... [the] level of user expertise and level of detail presented" (Cawsey 1993, p4). Cawsey has developed a system called EDGE which uses content planning rules to determine what to add into the explanation. The domain knowledge is kept separate from the user model. A noteworthy point made by Cawsey is that "tailoring based on a bad user model is probably worse than no attempts at user tailoring at all" (Cawsey 1993, p.7). She sees the ability to update the user model easily and naturally as part of the explanatory dialogue as essential. The system must be able to ask the right questions at the right time so that it can correctly assess the appropriate explanation level.

Swartout and Moore (1993) comment that the development of user models for customised explanations is very time consuming. They suggest that the use of stereotypical models may be a better approach. Using stereotypes the level of detail is customised to each stereotype. The levels of experience mentioned above are common stereotypes used. Rules have different levels associated with them and are fired if the user belongs to that level. Such a system is easy to implement but the explanation will be unsatisfactory if the user does not fit the stereotype.

Another form of customisation is tailoring to the users goals. A good explanation should show how the user's goals were used and point out alternatives. Such a system has been implemented by Cohen et al (1989).

Scott and Weckert (1995) point out that the knowledge needed for decision making may not be the most appropriate method for explanation. They consider two main issues: what type of knowledge is needed for explanation and how should it be represented.

Using a subdomain of viticulture, the diagnosis of phosphorus in grapevines, they have explored the combined use of schema, rules and models to provide explanations that are generated for each consultation. The system, Vitiex, includes knowledge of the domain, knowledge of explanation and knowledge of the user. As described under causal modelling, Scott and Weckert provide causal explanations.

7.3.3.1 A Summary of the Explanation Activity

Three ideas were repeated by the explanation researchers above, better explanation requires *more*, *deeper* and *abstracted* knowledge. These ideas were identified in Chapter One as important to the reuse of knowledge in general and have been discussed in Chapter Three in relation to RDR. The development of conceptual hierarchies based on RDR rules using FCA is the approach taken in this thesis to provide more knowledge which is structured into various levels of abstraction that is not so obvious from browsing the rules in their primitive form.

Despite its limitations, the primitive form of explanation using rule traces seen in first generation ES was seen as a major benefit of ES over other technologies. In RDR KBS rule traces also offer a history of changes and cases provide contextual explanations. The need for explanation comes back to the need for user control. The ability to explain the reasoning process allows experts to exercise their judgement whether to accept a conclusion or override it. Just as experts tend to justify their decisions based on the situation rather than provide a detailed explanation of the decision processes involved, an expert system must be able to point out why it has asked a question or recommended a course of action based on the context.

From the discussion above of explanation it can be seen that there are numerous approaches to offering explanation. To properly address explanation for KBS, whether using RDR or another knowledge representation, would require this study to focus completely on explanation. This thesis is interested in reusing knowledge for multiple activities with explanation being only one of those usages, albeit a significant one. Thus the approach to explanation offered is a general package of ways that the knowledge in an RDR KBS can be understood. As much of the toolkit is seen as beneficial for teaching we first consider the activity of tutoring and then the RDR approach to explanation and tutoring is described.

7.3.4 Tutoring

Kaplan and Rock (1995) claim that Intelligent Tutoring Systems (ITS) are the lost cousin of AI. Typically in an ITS there is a model of the student and pedagogue which allow the system to behave differently depending on the situation. This ability to adapt adds to the amount of knowledge to be kept and the complexity of how it is utilised. System adaptation must also be achieved while bearing in mind that user control is essential in ITS (Kaplan and Rock 1995).

Buchanan and Shortliffe (1984) describe ITS research as the attempt to answer:

- the nature of expertise (what do we want to teach ?)
- modelling (how to determine what the student knows ?)
- tutoring (how to improve the student's performance ?)

They clarify that teaching is different to explaining. In teaching, the student needs to solve problems independently. Therefore mnemonics are good in justifying rules because they assist memory. The ability to explain a conclusion is even more essential in a tutoring system because it is assumed that the user is a novice and will be unable to fill in the gaps that may be possible for an experienced user. Systems that only offer How or Why (in the form of a rule trace) are inadequate. Buchanan and Shortliffe (1984, p.540) mention four different types of justification which include: identification, causal, world fact and domain fact rules. Identification concerns determining the class to which an object belongs based on its properties. The nature of the knowledge in a causal rule will affect the type of explanation it can offer. For example, causal rules may provide one or more of the following: a causal link which is known to exist but is not understood (an empirical association); the direction of the causal link is known but the process is not well understood (complication); or a causal link whose process is well understood and can be modelled (mechanism). World fact rules include commonsense knowledge about the world. Domain fact rules concern domain specific knowledge and provide links between various hypotheses. The explanation given in MYCIN depended on the kind of justification upon which it was based. The need to determine the type of justification to be given is consistent with the observations of Jansen and Compton (1990) that the context affects the justification which is offered as an explanation.

While not the same, there is still a strong correspondence between the needs of explanation and tutoring systems. Both need structural knowledge which acts as the glue that holds it all together. GUIDON, which was built by Clancey as part of the MYCIN family, is perhaps the best known tutoring system. In GUIDON (Clancey 1984) the knowledge needed for teaching was kept separate from the domain knowledge. Knowledge was distinguished by its roles. These roles include:

- The heuristic rule, which links the data and the diagnosis or therapy,
- Structure, how rules are indexed and controlled,
- Strategy, the when, why and how of rule application
- Support or justification for the rule and includes world knowledge/commonsense.

GUIDON has a modelling component representing the user. It tries to work out such things as how much the student knows and if they picked an answer for the right reason. Confidence factors are used to determine the system's confidence that the student really understands the domain. Various types of knowledge are kept in the system, including knowledge of dialogue patterns for explanations and discussion, domain knowledge and knowledge of the communication. The communication model has an overlay student model, case lesson plan and focus record.

Clancey (1984) explains that in tutoring there is not necessarily one correct path or what appears to be alternatively arbitrary choices (according to the rule base) are not necessarily arbitrary in real life. The order of rule conditions may affect the way the system behaves. It is therefore necessary to provide look ahead and planning to guide the user but also to provide flexibility.

White and Frederiksen (1989) have found causal models to be useful as intelligent tutoring systems in the fields of science and engineering. They use a progression of models based on qualitative causal models. Each model builds on the previous one, starting with the simplest and moving to the most complex. The models are introduced in the form of problems to be solved which are used to build models that can be used for causal explanation and to show problem-solving strategies. Rather than trying to manage the complexity of all possible combinations of how the student could use the system, the causal model derived by the system is compared to that derived by the student. In this manner the student can pin-point where their model is deficient. The system generates problems and explanations that help the student discover the differences between the two models and how to amend their own.

A major benefit that White and Frederiksen (1989) claim in using causal modelling for tutoring is that it provides the freedom and flexibility necessary in ITS without the complexity that many other approaches need. The user can solve problems, watch demonstrations and ask for explanations progressing from novice to expert. The strategies offered include: exploration and discovery learning, learning via induction and feedback, learning from examples and explanations or a combination of mixed strategies. A map is kept for each student so that they can see the progress they have made and the direction in which they are heading. The freedom of this ITS supports multiple learning styles and encourages initiative and motivation which they will need to eventually qualify as experts. Their application was evaluated with seven students.

The socratic method of teaching is where a student is lead through various questions to formulate general principles from individual cases. It is a means of proposing and testing hypotheses. Domains that support causal modelling are best at offering such a teaching strategy (Wenger 1987).

Wenger (1987) sees “communicability” as a key differentiating feature of ITS. There is a spectrum ranging from *black box* to *glass box*. Wenger provides the following history of ITS in which the similarity of ITS to causal modelling is evident. One of the early systems was SCHOLAR which used semantic nets to support an *overlay model*. The distance between nodes in the semantic net was an indicator of relevance. Numerical tags were attached to nodes to guide the selection of topics. The use of the semantic net and numerical tags is similar to the use of a symbolic hierarchy and weights in the CBR approach used by Kriegsman (1993). Wenger argues that the semantic net approach could not represent procedural knowledge adequately and as found by Brachman and Levesque “inheritance of properties in conceptual hierarchies is a complex problem in general” (Wenger 1987, p.33).

The use of scripts followed the use of semantic nets. Scripts use stereotypical sequences of events and are better at representing processes. The scripts are organised in a hierarchy allowing the user to progress from general understanding of the concepts through to more specific details. Although seen as an improvement, the scripts in WHY were insufficient to explain the mechanisms or correct student misconceptions (Wenger 1987, p. 42). Scripts provided a view that was still too local. The need for multiple viewpoints was seen and developed into the search for multiple mental models. The componential view of mental models is similar to Lee’s (1996) use of compartmental models in her work on causal modelling. Wenger (1987, p. 53) found that causal models need to distinguish between time bound processes and functional relationships. There appears to be a move in ITS towards simulation models such as those supported by Augumented Transition Networks (ATN). The HTW work (Vescovi et al 1994) described under causal modelling includes the use of simulation models.

The following discussion reviews how RDR, including previous implementations and the MCRDR/FCA tool developed in this study, support the activities of explanation and tutoring. First we will revisit how previous RDR implementations support explanation and then look at how the addition of FCA and other work reported in this thesis has added to the explanatory power of RDR. There is no intention to build user models or customised explanations as this would conflict with the goal of allowing the user to choose what activity they wish to perform and in what way.

7.3.4.1 The RDR approach to Explanation and Tutoring

RDR offers a paradigm shift from existing explanation research. In section 7.3.3 there was major criticism of the first generation ES style of explanation that simply involved rule traces. One of the problems with conventional rule traces was that they did not give a global, or deeper, view of the knowledge.

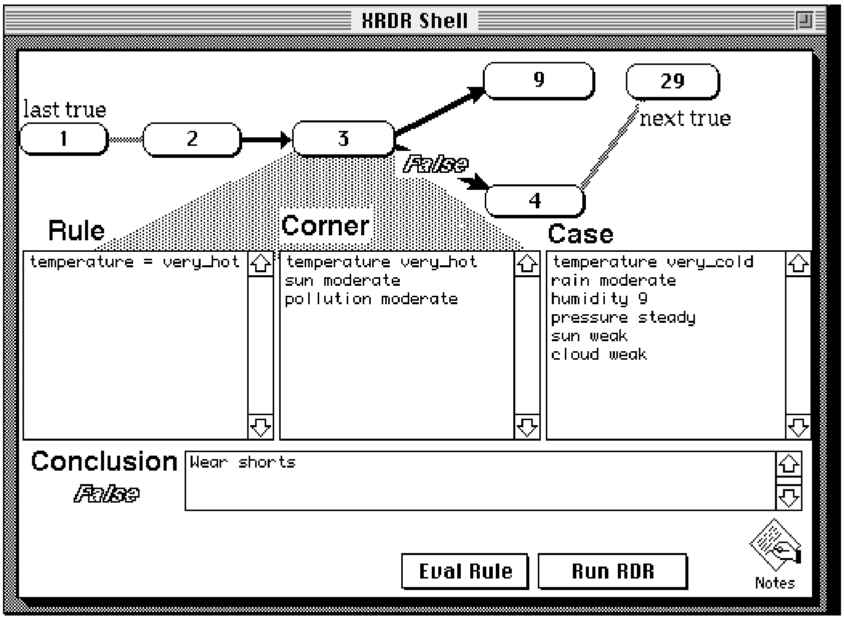


Figure 7.13: A rule trace in single classification RDR

Explanations that rely on the use of sub-goaling to reach conclusions often do not provide explanations that match the way in which the expert and/or user reach the same conclusion (Clancey 1984). Clancey found that for explanation or learning purposes ES "need to articulate how rules fit together, how they are constructed" (Clancey 1984, p.59). Similarly, Swartout (1993) argues that in most systems the *deeper knowledge* needed for explanation is *compiled out* and lost at development time. This loss of information is not so severe in an RDR KBS since a history of the evolution of the rule base is stored in the tree structure and associated cornerstone cases. It has been claimed that this history can help to reduce ES brittleness (Kang 1995). Clancey (1984) makes the point that students should not just be able to confirm a diagnosis but should be able to learn under what circumstances that conclusion should be considered and what other possible diagnoses explain the data. This is supported through the RDR exception structure by offering not only an explanation of why a particular conclusion was given but also why other conclusions had not been given and what values could be altered to reach another conclusion. This can be seen in the rule browser in figure 7.13 which is part of the XRDR system on the Macintosh. Alternatives are harder to determine in conventional ES due to complex interaction of rules and the numerous possible pathways to arrive at a conclusion. The rule conditions identify the salient features in a case that prompted the recommendation. The rule conditions on the true and false branches in the rule trace provide a deeper view of the knowledge because they provide the history and context in which a rule exists. The cases associated with each rule support another contextual dimension of explanation.

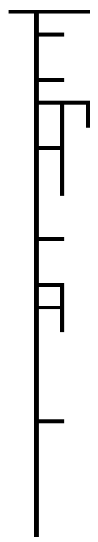


Figure 7.14: Typical RDR Tree

In figure 7.14 we can see the structure of a typical single classification RDR KBS. The false branch is shown moving down and the true branch moves to the right. We can see that most rules are attached to the false branch. Each time we run a case a new path is traversed and the trace is kept. Figure 7.1 showed the rule trace generated when case 53 is evaluated. For the purposes of explanation, it seems pointless to keep the false branches before the first branching from the main stem occurs, since there may be many different combinations of attribute-value (A-V) pairs. It is debatable whether the first true rule is significant and representative of some more general features that distinguish this case from others and that subsequent rules are refinements of this knowledge. The important information that a rule trace provides is not found by consideration of true or false branches only but is provided in the differences between the cases along the way (Compton, personal communication). In keeping with this, it did seem useful to provide some of the false branch information as part of the explanation given to the user but only after branching from the main trunk. To test this idea a modification was made to the Test Screen as shown in Figure 7.15. The false rules prior to the first true branch have been stripped away. The conclusion and the conditions of *all* rules after the first true rule were displayed to the user showing the true conditions in one box and the false conditions in another. The conditions of true and false rules were shown in separate boxes for clarity. Previously, only the conclusion from the final true rule and the conditions of true rules were given to the user. It can be seen in Figure 7:15 that a key reason why conclusion 36 (the conclusion for rule 134 which was the last true rule) has been given is because the sex of the patient is unknown. This feature in the case has resulted in rule 19 (with the condition SEX=FE) failing. We can see that if the sex of the patient was female conclusion 61 would have been recommended. We could test out this by changing the value in the case and seeing if any other exceptions would override conclusion 61. According to Clancey's criteria for ITS previously mentioned, the expanded explanation provided the means to determine the final diagnosis as well as which other diagnoses should be considered and what alternative conclusions are likely. It was also possible to alter the value of attributes in the case and see how the path and conclusions changed. It is possible to use this trace to answer all of the questions a user may want to ask as outlined in Section 1, except for critiquing (question 8).

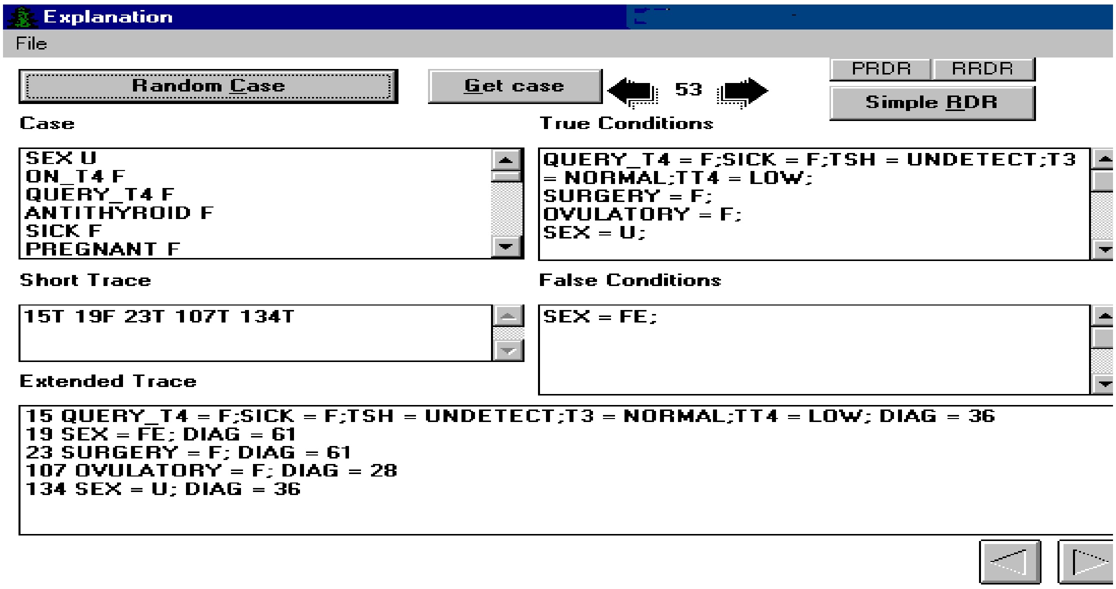


Figure 7.15: An extended explanation screen.
This screen shows the conditions on the true and false branches after the first branching from the main stem. False rules prior to branching are discarded from the trace.

The impetus for providing better explanation stems from the attitude that if the user understands why the ES came to a certain conclusion or asked a particular question then the user is empowered to use the knowledge in a better way. An alternative and possibly contentious suggestion (Edwards 1996) is that explanation is not really so important in deciding whether to accept advice. Rather, the lack of trust stems from a lack of system credibility. If statistics were kept regarding the number cases seen and correctly classified, it would be possible to determine the accuracy of the system and let the user evaluate the system's credentials before deciding to accept the current conclusion. As mentioned in Section 3.1.3, Edwards (1996) has implemented such a strategy for RDR KBS which collects various statistics and determines when a system is acting prudently.

The focus of much explanation research is on providing a detailed line of reasoning to the user. It is argued here that a different approach to explanation is worth considering since we do not really understand what expertise is and how an expert arrives at a conclusion. It may be that if we provide the user with enough information on the circumstances under which the rule was meant to apply and the relationships of that rule to other rules, which is offered to a large extent by the RDR structure, the user can develop a line of reasoning that works for them.

This is the approach adopted in this thesis. The RDR toolkit which supports the user in performing explanation and tutoring includes: the rule trace which can be visually represented as a tree (Figure 7.14), rule and tree browsers (Figure 7.13), traces and extended traces (Figures 7.1 and 7.15), the test pathways screen for comparing two rules or one rule against all other rules (Figure 7.11), the knowledge structured into an abstraction hierarchy as shown in the concept matrix (Figure 5.3) or lattice (Figure 5.6), the ability to select numerous views of the KBS on which to focus (Figure 5.11) and the ability to compare the models in different KBS (Figure 6.9). The user is free to use and reuse knowledge in any of these ways according to their current needs and preferences.

Although ITS has additional requirements, much of the above discussion on explanation applies to ITS. The use of RDR for tutoring has not been explored in great depth. From the ITS literature, to adequately support ITS it would be necessary to incorporate teaching knowledge together with a model of the student, to guide explanation. The closest usage of knowledge in an RDR KBS for teaching is the work done by Lee and Compton (1995) on causal modeling. The approach taken in this thesis to ITS is the same as the approach to explanation and the same toolkit is offered for learning purposes. Taking a situated view of knowledge and the variability in student's needs, the approach in this thesis it to let the user decide what tool is most appropriate to their current educational requirements. To test this idea some evaluation has been performed using the concept lattice as a teaching tool. The method and results are described in Section 8.2.

Explanation in RDR can also be evaluated using Swartout and Moore's (1993) criteria for a good explanation. Let us consider each of the aspects they mention in relation to RDR explanation.

1. **Fidelity** asks how accurate is the representation ?
   The explanations offered in MCRDR/FCA cover a range of representations based directly on the rules and generated automatically by the system. There is no discrepancy between what is shown in the explanation and what is held in the KBS which is more likely to occur when a separate system is used for explanation or when an automatic programmer is used that selectively chooses what information should go into the performance system and what should go into the explanation system. The use of GTs with specialised explanations are also prone to errors in fidelity because as pointed out before if the problem solving method is changed the explanation system must be changed accordingly.

   Swartout and Moore also commented that fidelity is aided by a simple inference engine. The inference engine in RDR is simple. To obtain a conclusion using single classification RDR the system simply tests the case against the rules and follows the true or false branch depending on the result. The last true rule on the pathway provides the conclusion. Inferencing in MCRDR is also simple and requires the children of all true nodes to be evaluated. The conclusion for rules at the end of pathways are the conclusions given for the case.

2. **Understandability** asks whether the content and context are understandable? Cases provide the context in RDR and provide an easily understood approach to context. Testing the comprehensibility of the content of the RDR explanations is not so easy. An evaluation of the line diagram format is given in Section 8.2.1 where 12 computer engineering students and academics took part in an evaluation of the line diagrams compared to other representations of the same data. In that survey it was found that 10 out of the 12 participants were able to learn to read a line diagram within a few minutes. However, trying to use the line diagrams to learn from was not found to be such an easy task. Within the half hour experiment some people were able to draw some conclusions regarding knowledge in the domain but others found the representation difficult to follow. For more details refer to that chapter.

   The components offered by Swartout and Moore can be used to focus on more specific aspects of understandability. They are:

   terminology: an RDR system is developed by the expert and the terminology used for rule conditions is based on the cases and the terminology used for conclusions is determined by the expert. Therefore the terminology should be acceptable to the expert user.

   user sensitivity - MCRDR/FCA does not use any type of user modelling and thus is not sensitive to individual users. However, it is envisaged that any application system based on MCRDR/FCA would be customised to the user group so that the range of available *activity-uses* would support the range of different users within that group.

   abstraction into different levels - the FCA line diagrams uncover and order the concepts in the MCRDR rules into an abstraction hierarchy.

   summarisation (the amount of detail offered in an explanation) - Rule traces allow the user to focus on the part of the KBS of interest to them. The line diagrams may be generated from a number of different views that provide a summary of just that aspect of the KBS and the amount of labelling in a line diagram may be adjusted to show a combination of attributes, rule conclusions, concept names or full labelling which includes attributes and objects that can be reached by ascending or descending pathways, respectively.

   the possible system perspectives - as noted in the previous point, and described in Section 5.2.1, there are a number of views of the knowledge from which the user can select to derive a line diagram.

   linguistic competence - as noted above the terminology used is essentially what is appropriate for the domain and what is used by the expert. However this terminology may not always be the easiest to follow particularly for a new user. A major problem with explanations based on rules is that when an automatic explanation is generated they are limited to the terminology used in the rules and transposition into natural language often does not result in sentences which, from a human perspective, appear all that natural.

Feedback - was discussed in Section 3.3.1.4 and is seen as an integral part of any system that relies on the user for system development and maintenance. Feedback in MCRDR/FCA is immediate in that changes made to the value of an attribute in a case or the addition of a new rule is immediately reflected in the KBS and in any subsequent line diagrams using that knowledge.

3. **Sufficiency** asks whether there is enough knowledge to provide explanations in different contexts. Sufficient knowledge to support an adequate explanation may not be present in an MCRDR KBS. Lee and Compton (1995) found that for causal explanations more knowledge needed to be acquired from the expert where the link was not provided in the heuristic rules. Clancey (1984) found that rules which gave guidance on the teaching strategy were necessary in GUIDON and others have found that the problem solving method should be part of the explanation. All rules in an RDR KBS are operational domain knowledge and do not include rules with different roles. This study questions whether PSM knowledge is really necessary. Since we don’t really know how the expert solves the problem, the usefulness of explaining how the computer solves the problem is debatable.

   Swartout and Moore (1993) describe the types of knowledge that may be needed to support sufficient explanation and include knowledge about the system's behaviour, justification, preferences, domain explanations and terminology definitions. The explanations provided by MCRDR/FCA are domain explanations with the associated cornerstone cases providing the justifications. Since there are no control rules or user models, system or user preferences are not applicable. The system’s behaviour is standard for all MCRDR systems and is not related to the domain or the task. The lack of terminological definitions, such as comprehensive descriptions of conclusions and the attributes and values found in rule conditions, is an area that could be improved. Any deployed system should offer assistance in understanding both the terminology related to the MCRDR interface and the domain in such forms as help screens and user manuals.

4. **Low construction overhead** - how time consuming and difficult is it to build the explanations ? The explanations provided by MCRDR/FCA have very low construction overheads as they are all automatically generated from the rules.

5. **Efficiency -** what will the system response time be like, will it take a long time to generate an explanation ? Although the construction overhead is low the efficiency of building those explanations, in particular line diagram explanations, is computationally expensive as the calculation of concepts is NP-Hard and can become a memory and response time problem as the KBS size grows. This is a common problem in AI and is related to the search strategy used by FCA in deriving concepts. The solution to the problem is to restrict what is included in a formal context as described in Section 5.2.1.

A summary of how MCRDR/FCA measures up to Swartout and Moore's criteria for good explanations is given in Table 7.1. The four main explanation architectures they described are included for comparison. A question mark indicates that insufficient information on the architecture did not allow an evaluation to be made. The ratings used are poor, good or excellent and are guesstimates based on discussions in the literature.

Approach / Criteria	Neomycin	Generic Tasks	EES	Reconstructive	MCRDR/FCA
Fidelity	Good	Poor	Good	Poor	Excellent
Understandability	Good	Excellent	Excellent	Excellent	Good
Sufficiency	Good	Good	Good	Good	Good
Low Const. O/H	Good	Poor	Poor	Poor	Excellent
Efficiency	Good	?	Good	?	Good

Table 7.1: A Comparison Table of how MCRDR/FCA compares to other well-known KBS explanation tools.

It can be seen from Table 7.1 that MCRDR/FCA compares favourably with other explanation approaches and is particularly strong when fidelity and construction overheads are considered. The approach to explanation offered by MCRDR/FCA is novel when compared to these other approaches. The other approaches place most of the responsibility for explanation on the system with the user playing a less active role in deciding the type of explanation they should be given. In contrast, MCRDR/FCA offers a toolkit that the user employs to provide the type of explanation they are seeking.

7.3.5 What-if analysis[3]

The ability to test out different scenarios is an important human activity as evidenced by the popularity of spreadsheet software which contributed to the widespread acceptance

[3] Much of this section has been taken from Richards, D.C. (1994) "The Adaptation of Traditional Expert Systems to Provide 'What-if' Analysis in the Decision Support System Style" Master's Thesis, Charles Sturt University, Riverina.

of the personal computer and Decision Support Systems (DSS) in the 1980s. Due to the high end-user support for such systems, this section considers not only ‘what-if’ analysis but also the various features of a typical DSS that may be desirable to include in a user-centred system.

Philippakis (1988, p.366) sees choice and exploration as the two main purposes of ‘what-if’ analysis. Philippakis also explains that traditional ‘what-if’ functions found in many Decision Support System (DSS) are unsatisfactory due to the lack of optimal decision processes for large volumes of data and 'avoidance of disconfirming evidence'. In ‘what-if’ analysis subsequent steps depend on the outcome of earlier steps (Leigh and Doherty 1986), not a feature easily incorporated in traditional ES.

Schultheis and Sumner (1992) offer the following description of ‘what-if’ analysis in the context of DSS:

- User has the ability to enter a number of different values and test the result of each situation (p.138, p.360);
- Important decision tool, that assists but doesn’t make the actual decision (p. 360);
- Allows the development of models (p.139);
- Project management software allows questions to be asked regarding changes in project event sequences and resource allocation, providing optimisation of resources (p. 144);
- ‘What-if’ modelling capability is found in some financial analysis software (p. 377);
- Built in financial, statistical and mathematical functions are often found in software providing 'what-if' processing (p.377);
- Usually associated with numerical or financial data;
- Some programs let the user specify a goal. The system then works backwards, providing the values that satisfy the goal (p. 377). Also known as goal-seeking (Turban 1993).

Flaaten et al (1989) include the following additional features. DSS which offer ‘what-if’ analysis often have graphic capabilities, usually support strategic and tactical decisions and are mostly PC-based systems. The data used in DSS are generally copies taken from the transaction processing system for individual use. The latter leads to the problems of keeping data current and avoiding data redundancy. Flaaten et al (1989, p. 132) point out that a major concern with DSS is that the conclusions they give may not be current. Many of the users of DSS are not aware of the proper use of statistical formulas, mathematical functions and even graphs.

The presentation and appearance of the output of DSS is of great importance to users, generally management. Even this can be misused to make bad or wrong data look presentable and be accepted as correct. Flaaten et al (p. 266) claim that the processing involved in the analysis of this data can take hours, a point disputed by Leigh and Doherty (1986) and Hopple (1988).

Turban (1993, p.65) refers to 'what-if' analysis as a type of sensitivity analysis and concisely describes it as a means of testing what will happen to outputs if there is a change in inputs. Turban discusses goal programming in the 'what-if' context. Goal programming handles optimisation problems, allowing the user to allocate resources where there are multiple and conflicting goals. 'What-if' is facilitated in three ways: by letting the user alter the priority of goals, modify the goal target values or step back through the consultation changing values.

Turban (1993, pp.851-872) discusses some of the techniques used in the financial analysis software, Interactive Financial Planning System (IFPS). Features of interest to this study are the ability to temporarily change data, reset data back to its original status (base case) and the use of command files to store different scenarios. Turban & Watkins (1985, p.150) explain that DSS may answer 'what-if' but in an ES the user can also ask 'why'.

While not specific to 'what-if' analysis the following list provided by Klein and Methlie (1990, p.150) is useful in describing the desirable features of a DSS style system.

1. End-user usage and interactivity;
2. End-user definition;
3. Easy access to pertinent information;
4. High interaction between users, the system and the learning situation;
5. Capacity to adapt to fast evolution of user needs;
6. Portability and peripheral support;
7. Reliability;
8. Performance.

Bennett (1983, p.20) provides a different, but equally useful list of DSS features, summarised below:

1. Multiple processes and decision types exist.
2. Inclusion of tools to help conceptualise problems, support Intelligence, Design and Choice activities and memory aids.
3. Customised decision making and user control is provided.

In a discussion on modelling, Young (1989, p.51) stresses the importance of user control. The user needs to be able to initiate the process, vary inputs, modify the model and select outputs and formats for display. A non-procedural language should be used so that the developer can control the model in a natural way without concern for syntax (p.52). He defines three types of variables to be considered: outcome variables, controllable impact variables and uncontrollable impact variables (p.84). He finds two classifications of models useful: the treatment of time and the treatment of uncertainty (p.81).

The features of 'what-if' analysis offered by the above authors becomes the criteria against which the altered system can be evaluated. These points are outlined in Table 7.2 below.

1. Users can test different scenarios through the manipulation of variable values
2. Assists decision making, particularly tactical and strategic decisions
3. Allows models to be set up and tested
4. Projections and forecasting ability is offered
5. Allow optimisation to be performed
6. Built in functions are usually available to assist model building
7. Ease-of-use and presentation are important, often includes graphics
8. Usually PC-based systems
9. In some packages users can specify a goal and be given the values that satisfy that goal
10. Usually associated with numerical data
11. Data is often derived from external sources, eg a transaction processing system
12. Problems of data repetition and lack of expertise in using the functions provided
13. The end-user usually defines, creates and uses the system
14. Need to be flexible and adaptable to the user's needs.

Table 7.2: Features of a DSS providing 'what-if' processing.

Much of the work by Richards and McDonald (1995) on adding decision support system (DSS) features to an existing ES was based on traditional ES that are not easily modified and offer the user very limited control. RDR ES are primarily built and controlled by the user and offer many of the features of 'what-if' and DSS software.

7.3.5.1 The RDR approach to What-if Analysis

The ability to let the user test out different scenarios by changing attribute values is a feature of existing RDR implementations. In current versions of XRDR and MCRDR for Windows it is possible to add or delete attributes or alter one or more values in a case and then look at the result. With the addition of the extended rule trace, mentioned above under explanation, it is possible to determine how the change in one or more attribute values affected the conclusion given. The ability to enter a conclusion and determine what value/s would satisfy that conclusion, which is a version of backward chaining known as goal programming, has not been explored. As with the study by Richards and McDonald (1995) it was found that the problem was one of combinatorial explosion on the number of possibilities that a user could enter or alternatively leave for the system to determine.

In many consultation-style systems that ask the user for input it is of course possible for the user to alter the values they provide to the system and then to see the effect of the change on the recommendation. However, a key aspect of 'what-if' analysis is the ability to choose whether or not the values should be updated and the ability to restore values to their original value. In many systems some of the values are received from external sources, such as was the case with AusVit, and it is not desirable to overwrite these historical values. RDR does allow manipulation of all attribute values in cases without affecting the actual case.

To conclude, the features of a 'what-if' system as outlined in Figure 7.2 are used as evaluation criteria for the MCRDR/FCA system. Some of the comments relate specifically to the MCRDR/FCA system and others apply generally to RDR systems.

1. In MCRDR/FCA users are able to change the value in a case to test different scenarios. An individual case represents a scenario that can be replayed and a set of cases is similar to the use of command files in IFPS which are used to store different scenarios. This tool has great value for 'what-if' activities as it is possible to see the impact of changes and to determine what changes are worth investigating.
2. The purpose of RDR systems, as with most KBS, is to assist decision making. However, DSS tend to be particularly employed to assist in tactical and strategic decisions. To date RDR systems have been more concerned with operational decisions perhaps with the exception of using RDR for acquiring search knowledge (Beydoun and Hoffman 1997) or flight control knowledge (Shiraz and Sammut 1998).
3. RDR systems allow performance knowledge to be captured into a simple model but it is difficult to determine what higher level models may exist. MCRDR/FCA overcomes this limitation. By varying the selection criteria, higher level models can be set up and explored.
4. The ability to use the knowledge in an MCRDR/FCA system for projections and forecasting has not been explored. This would require at least some model of time to be built into the system that would alter the behaviour of the system over time. However, the ability of many RDR implementations to handle time course data could be exploited together with the work on causal modelling to provide forecasts.
5. No particular optimisation strategies are offered with MCRDR/FCA. At this point the user would be responsible for trying out a number of variable values to determine which values produce the optimal result.
6. RDR systems are designed to be easy-to-use so that they may be managed by the end-user. Presentation of the knowledge is seen to be important and the ability to offer a number of representations of the same data is seen as beneficial to the user.

However, the focus of research into RDR has not been on the presentation of the user interface and there is no claim that the screens in this thesis are well-designed.

7. The RDR systems have been PC-based, this includes Macintosh microcomputers.
8. The ability to enter a goal and see the values that satisfy that goal has not been particularly explored in MCRDR/FCA. However, there are two *activity-uses* that could be used for this purpose. The first one is critiquing which does allow the user to input their own conclusions. When the user clicks the critique button they are shown all other pathways that lead to that conclusion and the user could derive the values from inspection of the rules. Alternatively, the user can select a particular conclusion from which to generate a formal context and then be shown all rules that have that conclusion in the form of a line diagram.
9. The data used by RDR KBS is typically qualitative although the data in the cases may be numerical which have been preprocessed to fit into a qualitative category. So while qualitative data are more common, numerical data may also be used.
10. The data in initial RDR systems, such as PEIRS, came from external sources.
11. RDR does suffer somewhat from repetition of the knowledge but this is due to the exception structure and not due to the prolific development of PC-based systems using copies of the mainframe data that is often associated with DSS.
12. The lack of expertise found in the users of many DSS is due to lack of understanding of the appropriate use of various statistical functions and graphs. The functions offered in RDR systems are typically user-defined so that their appropriate use should be understood. When it comes to development of the knowledge base it is assumed that the developer is an expert in that domain so there should not be a lack of expertise regarding domain knowledge either.
13. RDR systems are designed to be defined, created and used by the expert end-user.
14. The simple knowledge structure and inferencing method used by RDR has resulted in systems that appear to be more flexible that many other KBS approaches that are highly tailored to a particular task or domain. The MCRDR/FCA system has been particularly built based on the belief that systems need to be flexible and adaptable to the user's needs.

These observations are summarised in Figure 7.3. Points 4, 5, 6 and 12 are not included in Table 7.3 as features 4, 5 and 6 have not been considered necessary for the applications currently considered by MCRDR/FCA. Point 12 relates to problems associated with DSS and are therefore not desirable features to be included in an evaluation.

Feature	Supports
1. Manipulate variable values	YES
2. Assists decision making	YES
3. Build and test models	YES
7. Ease-of-use and presentation	YES but can be improved
8. Usually PC-based systems	YES
9. Users can specify the goal	YES to some extent
10. Uses numeric data	YES but tends to use symbolic
11. Data derived from external sources	YES for initial systems
13. The end-user defines, creates and uses the system	YES
14. Flexible and user-adaptable	YES

Table 7.3: An Evaluation of MCRDR/FCA against the DSS features in Table 7.2

The addition of FCA to MCRDR supports 'what-if' analysis not just for inferencing but also for KA and maintenance. By using the Test Pathways screen the user is able to consider how the addition of a new concept will impact on the existing KBS. While not currently implemented it would be a small enhancement to temporarily save the proposed pathway and to draw line diagrams that included the new concept/s and which were based on a number of different views according to the selection criteria used in generating a formal context.

It is also envisaged that improvements to the line diagram interface which allowed the user to drop or expand selected nodes would also facilitate 'what-if' reasoning about the knowledge. The ability to perform 'what-if' analysis at both a conceptual and behavioural level using MCRDR/FCA is a significant extension to what is typically supported by 'what-if' systems. We now go on to look at a refinement of the 'what-if' analysis activity which is referred to as hypothesis testing.

7.3.6 Hypothesis Testing

Another form of 'what-if' analysis is hypothesis testing. Hypothesis testing should be an activity supported by KBS as Clancey (1984) argues that people use hypotheses to reason and learn. Hypothesis testing is a more specialised application of 'what-if' analysis because the user starts with an expected outcome and explores various scenarios in pursuit of confirming or disconfirming their hypothesis. As noted in the previous section, a major problem with 'what-if' systems is that people tend to avoid "disconfirming evidence" (Philippakis, 1988, p.366). The use of ES for hypothesis testing is seen as superior because there is expert knowledge which can be used to show inconsistencies in the reasoning process. For example, the work by Feldman, Compton and Smythe. (1989) found previous assumptions (hypotheses) could not account for all the results they had and prompted further exploration and refinement of the hypotheses.

Similar work on a smaller scale but using the same data was performed by Menzies and Compton (1997). This work did not involve RDR. No work in this thesis has been conducted specifically for the purpose of hypothesis testing but the activities already covered in the critiquing, causal modelling, explanation, tutoring and 'what-if' activities all support hypothesis testing when the user starts with an expected outcome in mind and changes values in cases or looks at different views of the knowledge and refines the hypothesis based on the models presented by the system. In the evaluations described in Chapter Eight, it is shown that the FCA line diagrams can be used to propose a hypothesis which can be tested and refined by using some aspect of the original line diagram to generate a new line diagram.

7.3.6 Student Modelling

Buchanan and Shortliffe (1984) classify student modelling as a different usage of knowledge which involves just experimenting with knowledge structures and control strategies. They see that student modelling can assist cognitive research.

Clancey makes the point that representations for generation (for example automatic programming) and recognition are different and will affect different applications of the knowledge:

> *"This distinction between representations required for generation and recognition is crucial for any knowledge acquisition or student modelling system that does not receive a well-formed specification, but must abstract it from a sequence of observed behaviours" (Clancey 1992, p.71).*

The ramification of this is that a person's frame of reference will determine how a strategic theory is perceived. When strategy is part of what we are trying to teach then we are interested not only in giving the correct conclusion but showing the appropriate means of reaching that conclusion. This is a very difficult task as people will use different strategies from one another and particularly from a machine in inferring a solution.

A nonmonotonic inductive student modeling system known as THEMIS has been developed by Kono, Ikeda and Mizoguchi (1994). THEMIS uses DeKleer's ATMS to develop single world and multiple world contradictions from the observed knowledge of the student. These contradictions are seen as key parts of the student's learning process and identification of them is valuable in determining where a students knowledge is inconsistent and in tracking changes in the student's understanding.

As with hypothesis testing, the student modelling *activity-use* has not been explored in any detail in this thesis. However, given the difficulty of trying to build systems that cater for a wide range of learning strategies a group of students may use, the solution offered in this thesis is to provide sufficient information in various forms so that the student may interact with the system in a way that fits their learning style. The ability to critique and test out hypotheses using MCRDR/FCA does provide feedback to the student that their thinking may be incorrect and provides an explanation of the expert's knowledge contained in the KBS. However, since no model of the student is kept it is not possible to track changes in the students thinking or to query the system as to what areas the student's knowledge is most deficient. Perhaps statistics regarding use of the modelling tool could be kept for each user in a manner similar to the storage of statistics on the behaviour of the system as is done in Edwards (1996) *credentials.* There is, however, value in having a student develop an RDR KBS as a means of testing out their knowledge about a domain. In a study performed by the Health Insurance Commission (HIC) of NSW (Wang et al 1996) RDR was used to develop a KBS regarding detection of fraudulent behaviour. As part of the process the HIC had an expert classify the practice profiles of 1,500 medical practitioners. A member of the computer department then used these cases to develop a KBS. Through the use of RDR it was soon realised that there were many inconsistencies in the classifications assigned by the so-called expert. If the RDR KA technique had been followed the expert would have been presented with the cases and been forced to provide consistent conclusions as each rule was being built. RDR would have forced the *expert* to review his knowledge of the domain particularly where he held conflicting opinions. This would have also saved a duplication of effort and would have resulted in a set of well-classified cases and a validated knowledge base.

7.4 Chapter Summary

This chapter considers whether and how the original goal of this work has been achieved. That is, is it possible to reuse the same knowledge for a wide range of activities. The activities described in this chapter include inferencing, KA, maintenance, validation, critiquing, causal modelling, explanation, tutoring, 'what-if' analysis, hypothesis testing and student modelling. The first four activities were already well supported in RDR systems. The approach to KA/maintenance and validation have been strengthened through the inclusion of FCA into MCRDR which allows these activities to be performed reflectively.

The reflective activities such critiquing, explanation, tutoring and 'what-if' analysis had different requirements to the reflexive activities that MCRDR was best at supporting. The addition of FCA modelling tools has provided the links between concepts and the uncovering of higher level abstractions that were previously not obvious in the RDR assertions. The activities of causal modelling, hypothesis testing and student modelling have not been considered and treated in any great depth in this thesis but it has been argued that MCRDR/FCA still offers some support of these activities which could be extended further, if required.

The main benefits that the MCRDR/FCA system offers over the original MCRDR system are enhanced validation techniques, the ability to critique proposed conclusions and rules, the ability to perform ‘what-if’ reasoning at higher levels and a wider range of explanation and tutoring tools that allow the user to view and explore the knowledge in more ways by making explicit the conceptual relationships, abstractions and their structure that were implicitly contained in the RDR rules. Many of the claimed benefits of MCRDR/FCA hinge on the usefulness of the line diagram. It is therefore important to evaluate such things as how easily the line diagram can be learnt and comprehended. Such an evaluation is the concern of the next chapter.

Chapter 8

8 Evaluating the System

The goal of this thesis is the development of a KBS approach that supports the reuse of knowledge for a wide range of activities. RDR was chosen as the starting point of the investigation due to its various strengths including its demonstrated ability to support the expert in KA, maintenance and inferencing. While a discussion of these activities was offered in Chapter Seven it is not necessary to offer an evaluation in this thesis as this has been provided by other researchers. For a current evaluation of the MCRDR technique for these type of activities the reader is referred to Kang (1995) and Kang et al (1998).

The aim of this chapter is to provide some evaluation of the reflective activities. Chapter Seven included discussion of each activity and how MCRDR/FCA measures up to the criteria identified from the literature. This chapter is concerned with a different type of evaluation. Many of the reflective activities rely on the line diagram and this chapter concentrates on an evaluation of the FCA line diagram from a number of perspectives. The first one is generality. In Chapter Six it was claimed that a T-Box can be generated from any A-Box that can be converted into a decision table. As examples of this, in Section 8.1.1 two CLIPS rule-bases are converted into decision tables (cross tables) and line diagrams produced to describe these domains. Section 8.1.2 considers a different aspect of generality by evaluating whether the line diagrams drawn with FCA can be seen as a generally acceptable knowledge representation method. To achieve this the FCA diagrams are compared to a number of other graphic rule representations. The reader will of course decide which representations they prefer, and why, but some statistics from a survey, which is described in section 8.2.1.4 are given on how well each of these representations scored using a set of four criteria: pleasing appearance, comprehensibility, good structure and ability to determine implications.

In Section 8.2 a second set of tests has been performed to evaluate how well the diagrams can be used for instruction or tutoring. The first evaluation in Section 8.2.1 uses a survey to determine how quickly an individual can learn to read a line diagram, how the line diagram representation compares to using a rule trace of the same information and whether the line diagram can be used to learn something about the domain and/or knowledge base. The survey is reinforced in Section 8.2.2 by using domain experts to comment on what I, as a novice, have been able to teach myself about each domain from the FCA models based on the MCRDR rules. First we look at the issue of generality.

8. 1 The Generality of the MCRDR/FCA Modelling Tool

Two claims are evaluated in this section. The first is that Formal Concept Analysis can be used on any knowledge representation that can be mapped into a decision table and is considered in section 8.1.1. The second claim is that the modelling representation itself is comparable to other visual representations of the same knowledge and is discussed further is section 8.1.2.

8.1.1 The Generality of Using FCA to Model different Knowledge Representations.

This thesis has claimed that FCA can be used to display concepts in rule-based systems or any representation that can be converted into a decision table. Colomb (1993) has proven formally that any propositional KBS or decision tree can be mapped into a decision table. Once we have the knowledge as a binary decision table or crosstable, the mapping from a decision table to a formal context is a direct translation. In this section the viability of applying FCA to KBS representations other than RDR KBS is shown by using two KBS built using a commonly used and available knowledge representation, known as CLIPS. Two of the sample KBS that are supplied with CLIPS version 5.1 have been selected. The first KBS is an 84-rule KBS that classifies animals and the other is a 61-rule KBS that assists a user in selecting an appropriate wine.

8.1.1.1 The CLIPS Animal KBS.

The CLIPS Animal KBS is easily converted into a decision table. Such a conversion requires the expansion or *inflation* (Colomb 1993) of each rule into its primitive conditions. Conversion of the CLIPS KBS into decision table format requires all abstractions to be removed from the original KBS. Figure 8.1 shows each rule as a set of primitive concepts which is used as input into MCRDR/FCA.

Many interesting views of the animal classification system were found using this data. In Figure 8.2 the condition HOOVES=YES was selected. We can see that any animals that have hooves are also vertebrates (BACKBONE=YES), mammals (WARM.BLOODED =YES) & (HAS.BREASTS=YES) and herbivores (CAN.EAT.MEAT=NO). We can also see that the number of toes is the next consideration, which is then broken up into various categories such as if they have horns and how many, if they have plating, if they have fleece, if they live in the desert and whether they are domesticated. It is possible to use the diagrams to see what are the relevant features in classifying animals within an area of interest.

```
1 %BIRDP(backbone=yes);(warm.blooded=yes);(has.breasts=no);
2 %FLATW(backbone=no);(live.prime.in.soil=yes);(flat.bodied=yes);
3 %WORML(backbone=no);(live.prime.in.soil=yes);(flat.bodied=no);
4 %FISH0(backbone=yes);(warm.blooded=no);(always.in.water=yes);(boney=yes);
5 %SHARK(backbone=yes);(warm.blooded=no);(always.in.water=yes);(boney=no);
6 %CMINS(backbone=no);(live.prime.in.soil=no);(body.in.segments=yes);
   (shell=no);
7 %BAT00(backbone=yes);(warm.blooded=yes);(has.breasts=yes);(can.eat.meat=
   yes);(fly=yes);
8 %TURTL(backbone=yes);(warm.blooded=no);(always.in.water=no);(scally=yes);
   (rounded.shell=yes);
9 %FROG0(backbone=yes);(warm.blooded=no);(always.in.water=no);(scally=no);
   (jump=yes);
10 %SALAM(backbone=yes);(warm.blooded=no);(always.in.water=no);(scally=no);
   (jump=no);
11 %LOBST(backbone=no);(live.prime.in.soil=no);(body.in.segments=yes);
   (shell=yes);(tail=yes);
12 %CRAB0(backbone=no);(live.prime.in.soil=no);(body.in.segments=yes);
   (shell=yes);(tail=no);
13 %JELLY(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=yes);(stationary=no);
14 %PROTO(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=no);(multicelled=no);
15 %CRCAL(backbone=yes);(warm.blooded=no);(always.in.water=no);
   (scally=yes);(rounded,shell=no);(limbs=yes);
16 %SNAKE(backbone=yes);(warm.blooded=no);(always.in.water=no);(scally=yes);
   (rounded,shell=no);(limbs=no);
17 %SANEM(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=yes);(stationary=yes);(spikes=yes);
18 %CORAL(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=yes);(stationary=yes);(spikes=no);
19 %SNAIL(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=no);(multicelled=yes);(spiral.shell=yes);
20 %MONKY(backbone=yes);(warm.blooded=yes);(has.breasts=yes);
   (can.eat.meat=yes);(fly=no);(opposing.thumb=yes);(prehensile.tail=yes);
21 %RHINO(backbone=yes);(warm.blooded=yes);(has.breasts=yes);
   (can.eat.meat=no);(hooves=yes);(two.toes=no);(plating=yes);
22
          %HORZE(backbone=yes);(warm.blooded=yes);(has.breasts=yes);(can.eat.
   meat=no); (hooves=yes);(two.toes=no);(plating=no);
23 %WHALE(backbone=yes);(warm.blooded=yes);(has.breasts=yes);
   (can.eat.meat=no);(hooves=no);(lives.in.water=yes);(hunted=yes);
24 %DOLPH(backbone=yes);(warm.blooded=yes);(has.breasts=yes);
   (can.eat.meat=no);(hooves=no);(lives.in.water=yes);(hunted=no);
25 %CLAMO(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=no);(multicelled=yes);(spiral.shell=no);(bivalve=yes);
26 %SQOCT(backbone=no);(live.prime.in.soil=no);(body.in.segments=no);
   (digest.cells=no);(multicelled=yes);(spiral.shell=no);(bivalve=no);
27 %MAN00(backbone=yes);(warm.blooded=yes);(has.breasts=yes);
   (can.eat.meat=yes);(fly=no);(opposing.thumb=yes);(prehensile.tail=no);
   (nearly.hairless=yes);
```

Figure 8.1 Part of the CLIPS Animal KBS expanded into its primitive form

One more example is a search for animals closest to man. There were two ways that this could be performed. One was to select the Use Related Conclusion option on the Selection screen, see Section 5.2.1.4, for the MAN conclusion code %MAN00. This, however, resulted in the inclusion of any rules which had any of the same rule conditions as the man rule. This meant that all vertebrates were chosen and resulted in too many rules being selected. To overcome this problem the selection was restricted to the aspects of man that were of primary interest. This meant cutting and pasting the parts of the rule that were seen as most relevant to man. The condition: "(backbone=yes) ;(warm.blooded=yes); (has.breasts=yes); (can.eat.meat=yes); (fly=no); (opposing.thumb=yes); (prehensile.tail" were selected and the Generate from Clause option on selection screen 2, see Section 5.2.1.6, was used. A nice extension to the MCRDR/FCA tool for this purpose would be to allow the user to click on a concept and have the system display those concepts that were the closest.

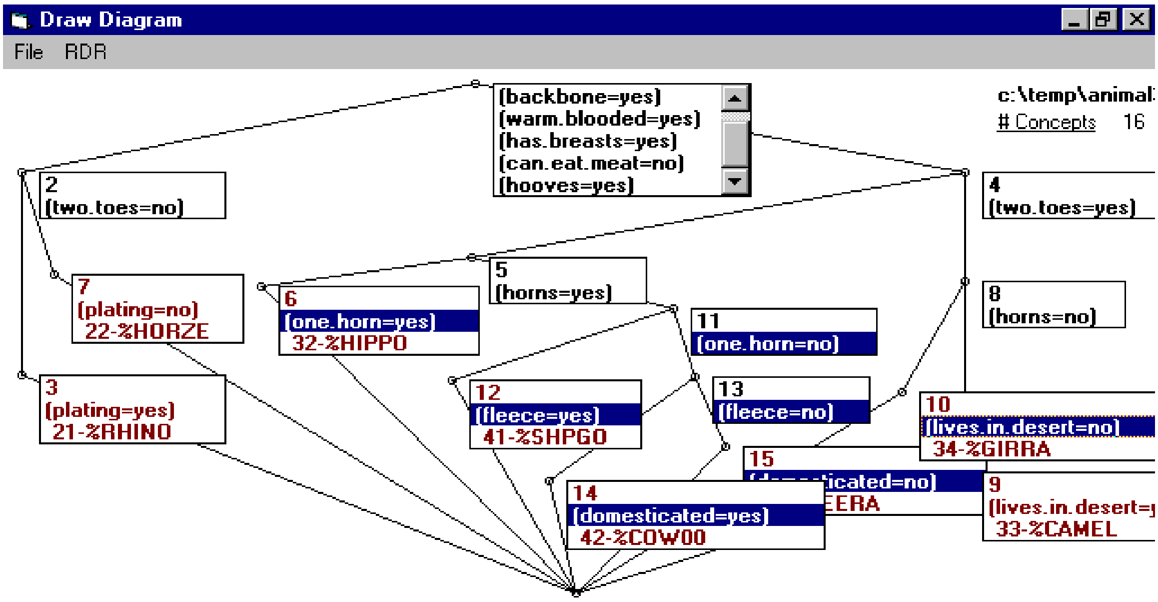


Figure 8.2The MCRDR/FCA diagram for the selection HOOVES=YES in the CLIPS Animal KBS.

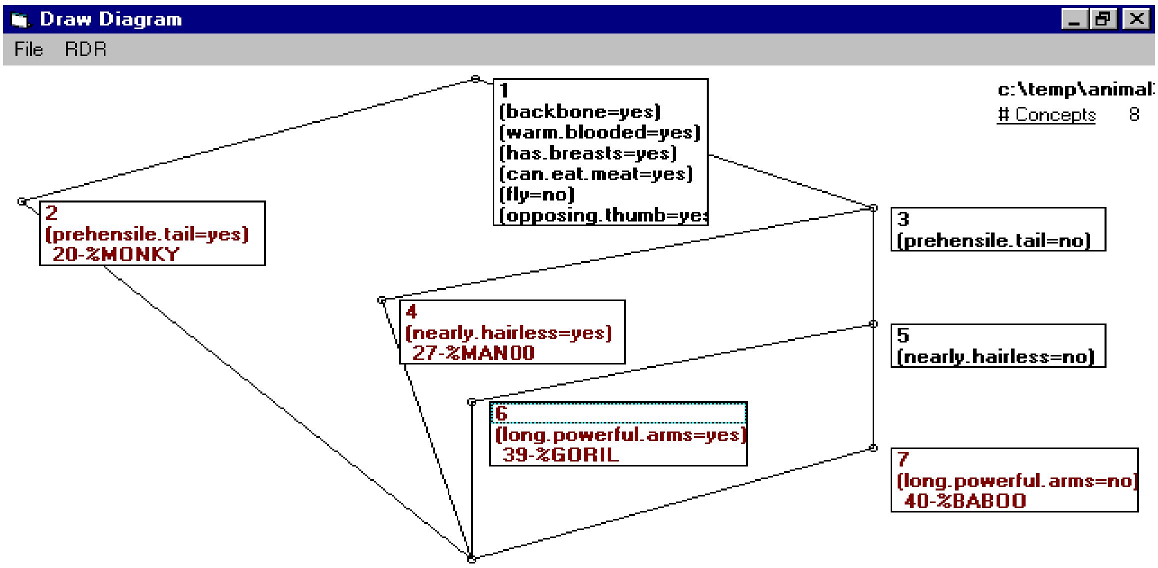


Figure 8.3 Finding the animals closest to man using MCRDR/FCA on the CLIPS Animal KBS.

As would be expected, in Figure 8.3 we can see in the top concept that all animals included have the selection criteria. The monkey is separated the most from man, then the gorilla and baboon are most similar with the difference being that the gorilla and baboon have body hair and man is nearly hairless. The gorilla and baboon are further differentiated by having long or not long, respectively, arms. If man was classified using the long.powerful.arms attribute it would appear closer to either the gorilla or baboon depending on the attribute value.

```
1      %GAMAY (recommended-color=red);(recommended-body=medium);(recommended-
sweetness=medium);
2      %GAMAY (recommended-color=red);(recommended-body=medium);(recommended-
sweetness=sweet);
3      %CHABL (recommended-color=white);(recommended-body=light);(recommended-
sweetness=dry);
4      %SBLAN (recommended-color=white);(recommended-body=medium);(recommended-
sweetness=dry);
5      %CHARD (recommended-color=white);(recommended-body=medium);(recommended-
sweetness=medium);
6      %CHARD (recommended-color=white);(recommended-body=medium);(recommended-
sweetness=dry);
7      %CHARD (recommended-color=white);(recommended-body=full);(recommended-
sweetness=medium);
8      %CHARD (recommended-color=white);(recommended-body=full);(recommended-
sweetness=dry);
9      %SOAVE (recommended-color=white);(recommended-body=light);(recommended-
sweetness=medium);
10     %SOAVE (recommended-color=white);(recommended-body=light);(recommended-
sweetness=dry);
11     %RIESL (recommended-color=white);(recommended-body=light);(recommended-
sweetness=medium);
12     %RIESL (recommended-color=white);(recommended-body=light);(recommended-
sweetness=sweet);
13     %RIESL (recommended-color=white);(recommended-body=medium);(recommended-
sweetness=medium);
14     %RIESL (recommended-color=white);(recommended-body=medium);(recommended-
sweetness=sweet);
15     %GTRAM (recommended-color=white);(recommended-body=full);(sauce=spicy);
16     %CBLAN (recommended-color=white);(recommended-body=light);(recommended-
sweetness=medium);
17     %CBLAN (recommended-color=white);(recommended-body=light);(recommended-
sweetness=sweet);
18     %VALPO (recommended-color=red);(recommended-body=light);
19     %ZINFA (recommended-color=red);(recommended-sweetness=medium);
20     %ZINFA (recommended-color=red);(recommended-sweetness=dry);
21     %CBSAU (recommended-color=red);(recommended-sweetness=medium);
22     %CBSAU (recommended-color=red);(recommended-sweetness=dry);
23     %PINOT (recommended-color=red);(recommended-body=medium);(rec-
sweetness=medium);
24     %BURGY (recommended-color=red);(recommended-body=full);
25     %BFULL (sauce=spicy);(recommended-body=full);
26     %BLIGH (tastiness=delicate);(recommended-body=light);
27     %BLI30 (tastiness=average);(recommended-body=light);
28     %BME60 (tastiness=average);(recommended-body=medium);
29     %BFU30 (tastiness=average);(recommended-body=full);
30     %BME40 (tastiness=strong);(recommended-body=medium);
31     %BFU80 (tastiness=strong);(recommended-body=full);
32     %BME40 (sauce=cream);(recommended-body=medium);
```

Figure 8.4 The CLIPS Wine KBS in MCRDR format

Some of the features used to classify animals in this KBS appear a little unusual and have occurred due to the purpose of this KBS which is to correctly classify an animal not to show the evolution of man or the food chain. For example, a bird was not defined in terms of having feathers or being able to fly but is defined in Rule 1 as : %BIRDP (backbone=yes); (warm.blooded=yes); (has.breasts=no). The reason this KBS could give the correct classification is that animals which become incorrectly classified as birds were further distinguished on the basis of not being able to fly (as shown in the man example in Figure 8.3) or the bat was distinguished as being able to eat meat and fly (See rule 7 in Figure 8.1). Of course many birds eat meat and most fly. This raises the question of whether the knowledge in this KBS could be reused for other purposes such as teaching a student the structure of the animal kingdom. In this case it appears that this KBS would require major modifications.

8.1.1.2 The CLIPS Wine KBS.

The CLIPS Wine KBS was structured differently to the Animal KBS. There was domain knowledge which included how to decide which wine to select and there was control knowledge which could be further divided into rules that controlled the user interface and other rules that controlled how and when user preferences should influence the general domain rules. The expert domain knowledge was easily converted to decision table format. The user interface rules were ignored. The user preference rules used many variables. Conversion to a decision table would result in a lot of similar rules with only one value different to cover all possible values of the variable. For the purposes of this evaluation only the domain rules have been converted into a decision table and used as input to the MCRDR/FCA modelling tool. A table with 43 rows was developed. The first 32 rules are shown in Figure 8.4. The wine KBS also differs from the animal KBS because the rule conclusions use confidence factors. To handle confidence factors a conclusion such as %BLI30 indicates that the recommendation is body=light with a confidence of 30% or %CWH90 is a recommendation of colour=white with a confidence of 90%. The confidence factors are treated as additional information associated with a conclusion and in the line diagram does not involve any sort of propagation of weights through the KBS.

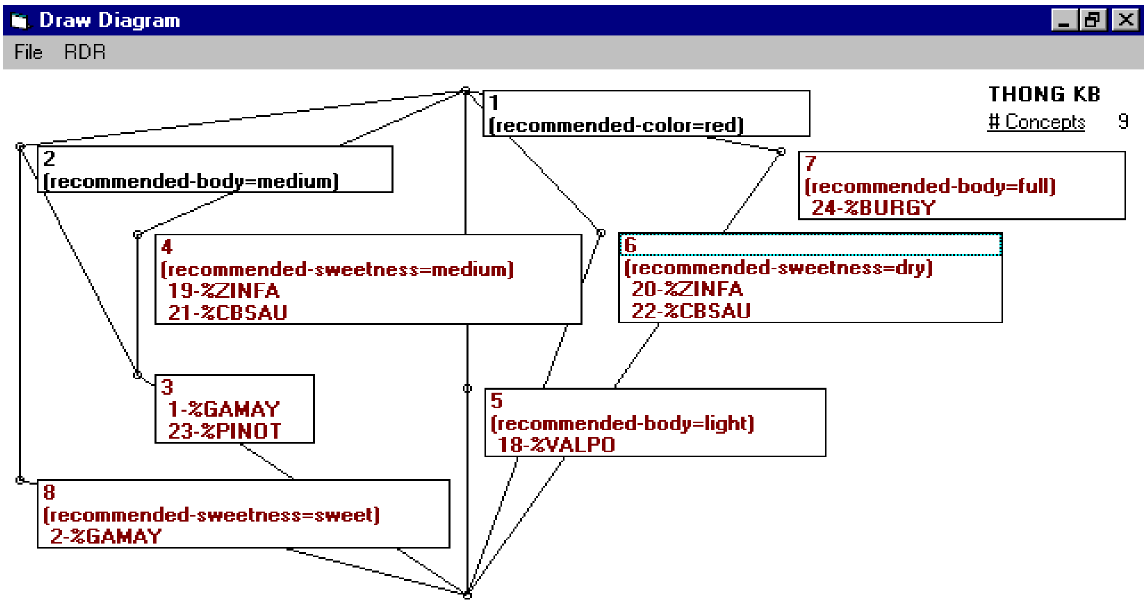


Figure 8.5 The MCRDR/FCA diagram for the selection COLOUR=RED in the CLIPS Wine KBS.

In Figure 8.5 the clause COLOR=RED has been selected. This diagram shows us not only which wines are red but also the BODY and SWEETNESS where it is part of the decision. For example, we can say that if the RECOMMEND-BODY=MEDIUM and the RECOMMENDED-SWEETNESS=MEDIUM the red wine GAMAY or PINOT NOIR would be suitable selections. However, if the RECOMMEND-BODY=MEDIUM and RECOMMENDED-SWEETNESS =SWEET the wine GAMAY would be a better choice.

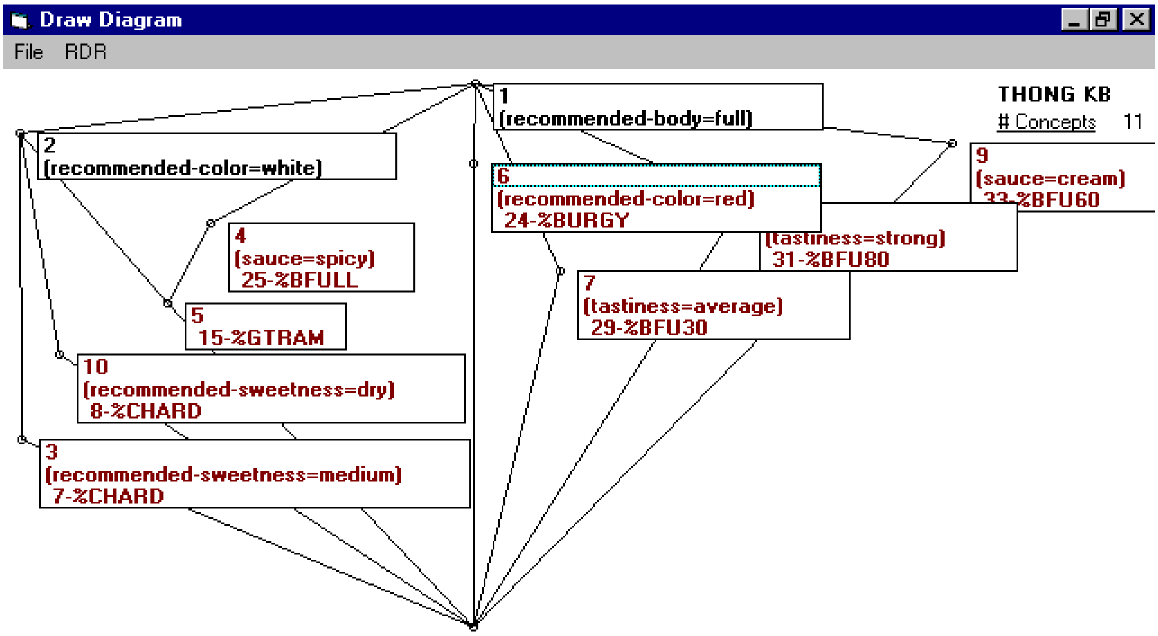


Figure 8.6 The MCRDR/FCA diagram for the selection RECOMMENDED_BODY=FULL.

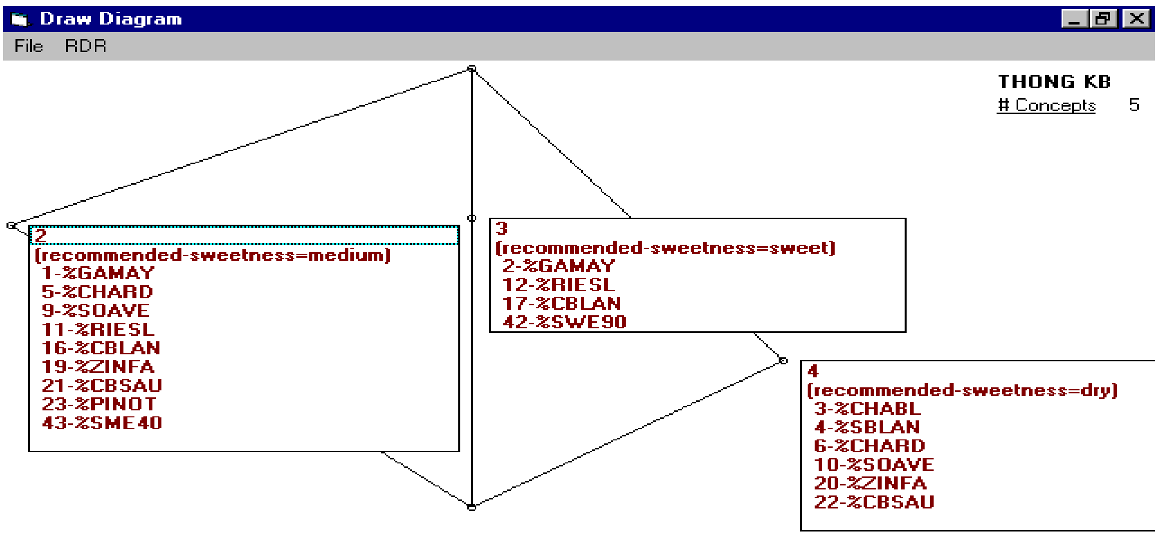


Figure 8.7 The MCRDR/FCA diagram for the selection RECOMMENDED_SWEETNESS using the option Clause with no related attribute in the CLIPS Wine KBS.

In Figure 8.6 the clause “RECOMMDED_BODY=FULL” has been selected. The diagram shows which aspects of a dish are important in this decision. As mentioned, the Wine KBS uses confidence factors (CF). BFU30 means BODY=FULL with a CF of 30%. Therefore, if tastiness is strong then a full body is more highly recommended than when the tastiness is average. If there is a spicy sauce a full-bodied wine should definitely be selected but if the sauce is creamy there is only a 60% confidence that a wine with a full body should be selected.

In Figures 8.5 and 8.6 all attributes used in rules with the specified attribute were included. If the extra information is not of interest the screen will be unnecessarily cluttered. In Figure 8.7 the attribute “RECOMMENDED-SWEETNESS” was selected with the option to not include related attributes. Using this selection we can focus on the sweetness of a wine to see which wines are sweet, medium or dry.

8.1.1.3 Summary

The animal and wine CLIPS KBS have shown that knowledge in rules-bases not structured as MCRDR KBS can be utilised to develop line diagrams from which a better understanding of the knowledge, the relationships between concepts, the higher level concepts and the structure of the KBS can be gained. These are only examples but there is no reason why it would not apply to all propositional systems. We now consider generality from the aspect of visual representation.

8.1.2 The Generality of the Visual Representation

In this section a comparison is made between various representations of the same knowledge to show the compatibility of the line diagram with other possible visual representations.

8.1.2.1 Comparing against Other Taxonomic Representations

In this section we explore the suitability of the FCA line diagram as an ontological and particularly a taxonomic representation. Ontologies are often equated with taxonomies that can be used to classify categories or concept types in a knowledge base (Sowa 1991). A taxonomy contains a stronger concept of structure and organisation than the word ontology as the word is derived from the Greek *taxis* and *nomos*, which respectively mean arrangement and law. Woods (1991) notes that: conceptual taxonomies which allow probabilistic and default rules and abstract and partial definitions maybe “useful for indexing and organising information and for managing the resolution of conflicts” (Woods 1991, p.45). The line diagram was used in Chapter Six for conflict resolution. Taxonomies include a classification system which provides structure that allows new concepts to be positioned. In the FCA framework, any new concept may be located based on the description of its *intents* and *extents*, the attributes and objects that describe the concept. In the MCRDR/FCA framework the taxonomy is recreated each time from the specified formal context and each individual concept is ordered in relation to each other concept. Limited default reasoning is supported in the line diagram through the inheritance of properties (attributes) but the use of roles for default reasoning is not well supported as there are not different links or arcs for different types of relationships between concepts. The line diagram can be described as a concept map with typed nodes but untyped arcs that is a hybrid system which includes a spatial layout and location system, node based system and a link-based system (Kremer 1998, Lambiotte et al 1984).

The line diagram is a complete lattice. Woods remarks that lattices offer “expressive power and the detection and resolution of conflicts in inherited information” (Woods 1991, p.80) that otherwise might not have been discovered until runtime. The concept lattice also satisfies the informal definition of a semantic net given by Shappiro (1991, p.137) as a “labelled directed acyclic graph (DAG) in which nodes represent entities and labelled arcs represent binary relations between entities”. The line diagram in FCA is a labelled DAG, although the arcs are not labelled but can be read as a binary relation which means “isa” when traversing the graph from bottom to top for extensions. Nets also provide “clustering of properties around concepts and the incorporation of inheritance hierarchies” (Schubert 1991, p.102) and inference propogation. Each of these features are supported by the FCA line diagram. Woods (1991) sees semantic networks as different to other KR because of the use of links to record facts and store associations which can be used for reasoning. Schubert, however, points out that many, if not most, KR are able to provide many of these features and tries to distinguish a semantic net from other representations. He draws the conclusion that semantic nets should not only be graphic taxonomic reasoners but that they should use graph theoretic notions. This point is made by Shaw and Gaines 1991b in this section’s closing quote.

The use of extensional definitions in FCA can be seen as problematic when we come to describe situations for which there are no real examples. However, say that we wanted to compare a horse, zebra and unicorn. There is nothing to stop us describing these three objects in terms of their real or imagined properties in a formal context. FCA is an intensional logic that uses descriptions of the attributes of objects as a means of classifying and ordering objects. Of course, there may be a concept that can not be found in any object, real or hypothetical, and it would not be possible to include this concept in our subsumption hierarchy unless the properties of that concept are contained in the intersection of other examples. This is a limitation of FCA. As can be seen in the example below it was necessary to treat the higher level concepts (collie, cat, robin and starling) as examples in order to include them in the taxonomy.

Primitive(creature)
Primitive(animal, creature)
Primitive(animal, bird, creature)
Primitive(dog, animal)
Primitive(dog, cat, animal)
Primitive(collie, dog)
Primitive(robin, bird)
Primitive(collie, spaniel, dog)
Primitive(robin, blackbird, bird)
Primitive(robin, blackbird, starling, bird)
Primitive(male)
Primitive(person)
Primitive(parent)
Primitive(male, female)
Primitive(man, male, person)
Primitive(woman, female, person)
Primitive(father, man, parent)
Primitive(mother, woman, parent)
Individual(Charles, spaniel)
Individual(Tweetie, blackbird)

Figure 8.8a Knowledge Structure Definitions and Assertions Compiled from Figure 8.8b (from Shaw and Gaines 1991b)

For comparison we look at some taxonomies produced by Shaw and Gaines (1991b) which they developed to demonstrate how their visual language reflects linguistic semantic arguments. They make a comparison between taxonomies based on Cruse's (1986) lexical semantics and the use of the disjoint and subsume relationships in their visual language. Gaines and Shaw point out that the visual language and other representations of subsumption hierarchies support many of the distinctions that can be made using Cruse's taxonomic lexical hierarchies such as branching or non-branching, the existence of a number of levels, generalisation at higher levels and differentiation and convergence at lower levels. In this section it is shown how the line diagrams drawn using FCA offer a suitable representation for taxonomies.

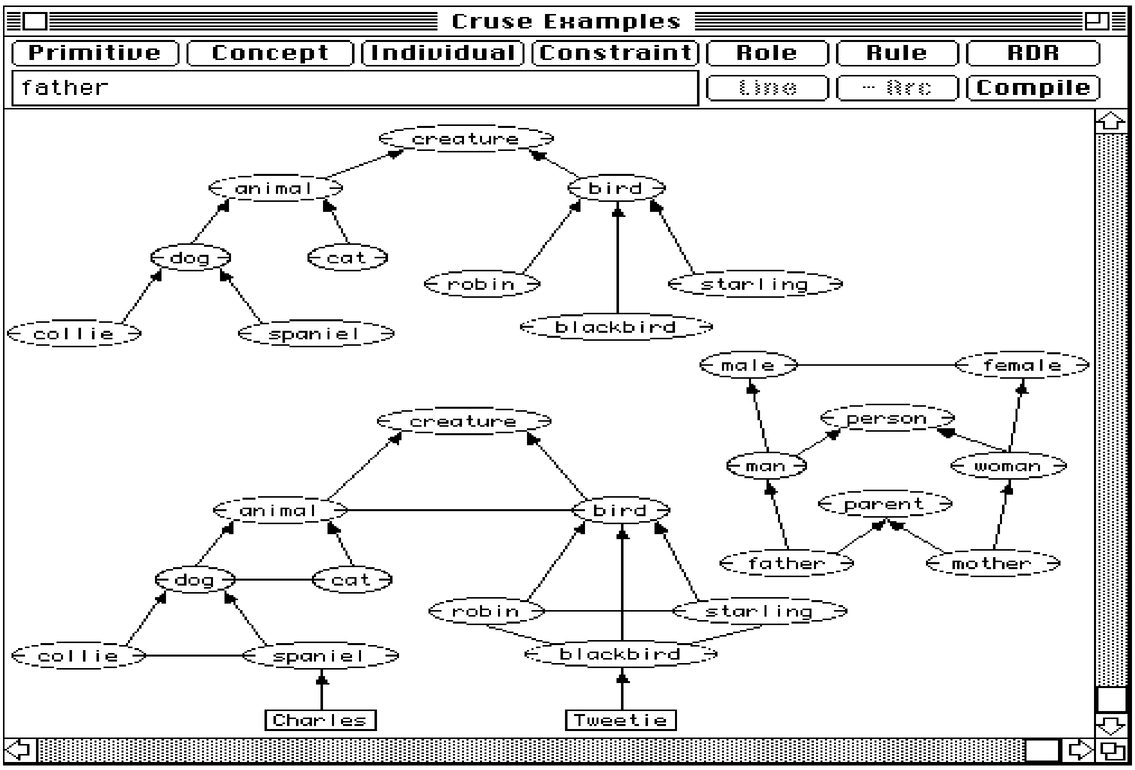


Figure 8.8b Taxonomies in Lexical Semantics in a Tool Based on the Visual Language (from Shaw and Gaines 1991b)

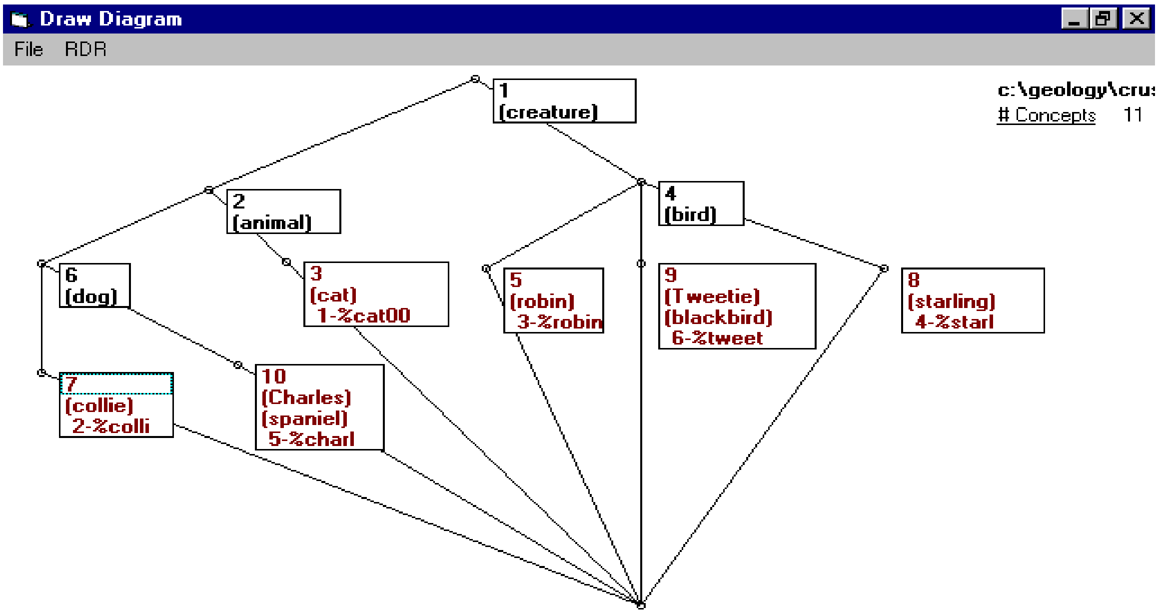


Figure 8.9a: A line diagram to represent the creature taxonomy expanded in the visual language in Figure 8.8a.

1	%cat00	(cat);(animal);(creature)
2	%colli	(collie);(dog);(animal);(creature)
3	%robin	(robin);(bird);(creature)
4	%starl	(starling);(bird);(creature)
5	%charl	(Charles);(spaniel);(dog);(animal);(creature)
6	%tweet	(Tweetie);(blackbird);(bird);(creature)

Figure 8.9b: The knowledge structure used to generate the line diagram in Figure 8.9a.

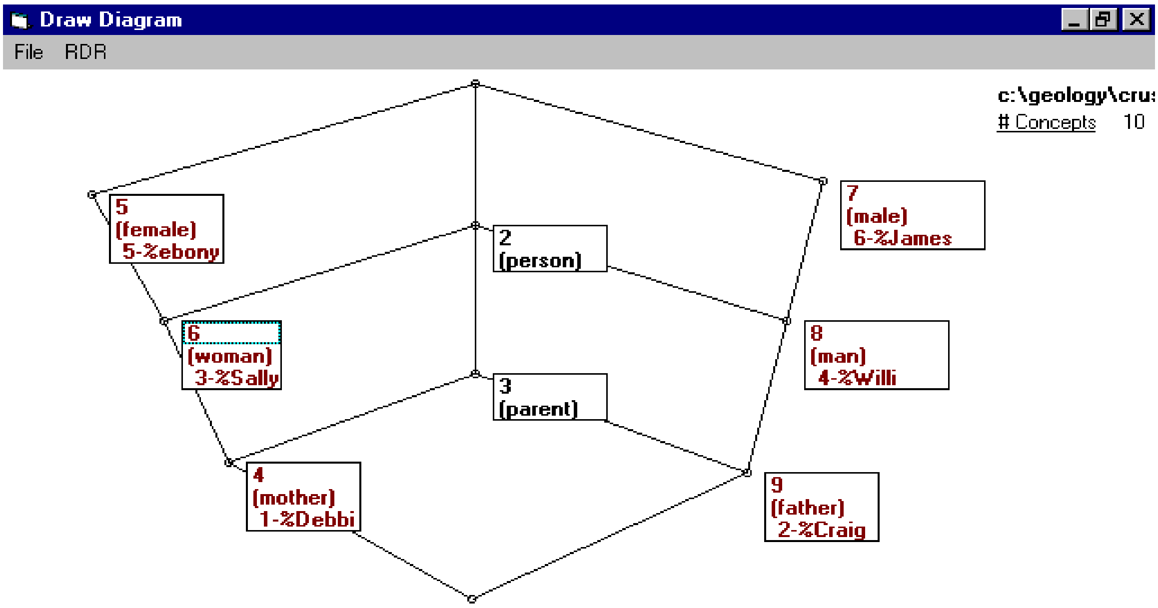


Figure 8.10: A line diagram to represent the father-mother taxonomy in Figure 8.8b.

In Figure 8.8b, the top taxonomy follows the creature taxonomy in Cruse (1986, p. 146) and the centre right father-mother taxonomy is based on the discussion in Cruse (1986, p.262). In the bottom of Figure 8.8b the creature taxonomy is expanded using the visual language. The visual language supports explicit representation of opposites through the disjoint relation which was not obvious in Cruse's taxonomy. The creature and father-mother taxonomy are represented using line diagrams in Figures 8.9a and 8.10, respectively. The key difference between the line diagram in Figure 8.9a and the taxonomy and the visual language graph in 8.8b is the extensional definition associated with each node at or below the *generic level* which is "the ordinary everyday names for things and creatures: cat, oak, carnation, apple, car, church, cup, etc" (Cruse 1986, p.146). In the creature visual language graph, Charles and Tweetie are instances of spaniel and blackbird, respectively. FCA relies on having examples to compute the higher level concepts. If there were instances of each type of animal or bird then the structure shown in Figure 8.9b which includes rules to represent the higher level concepts of cat, robin, starling and collie, would not have been necessary and would have resulted in a line diagram that only had extensional definitions shown at the bottom nodes. The closeness of the structure in Figure 8.9b to the KL-ONE-like structure produced by the visual language shown in Figure 8.8a is notable. Apart from the input being more succinct in the MCRDR/FCA rule representation (six rules compared to twelve based on the visual language), the key difference is that with MCRDR/FCA pathways[4] we started with the rules and derived the diagram and using the visual language the knowledge structures in figure 8.8b were derived from the graph in figure 8.8a.

The similarity to and strength of the visual language and the line diagrams over Cruse's representation is well summarised by Shaw and Gaines (1991b):

> *At one level they are isomorphic visual representations of a taxonomic structure. At another level, the computational representation has the added feature that the underlying data structure exists within the computer and hence is readily subject to graph-theoretic analyses of its fine structure. At another level, it is also an operational knowledge structure within the computer supporting automated deductive inference. At another level, this knowledge structure is available for composition with other knowledge structures to provide a knowledge-based system suitable for testing theories and supporting practical applications. (Shaw and Gaines 1991b, p. 12).*

In the next subsection we expand the comparison beyond taxonomic representations to a number of other graphic representations of the same knowledge. The focus is more on the acceptability and usefulness of the representation for human reasoning.

4 Pathways means the exception structure has been flattened by picking up all rule conditions leading back to the head rule.

8.1.2.2 Various representations of Cendrowska's Lens Data

One of the claims of this thesis has been that the line diagram is a useful representation for understanding the contents of a rule-base. In this section we consider a number of visual representations of the same set of rules and their various merits. We consider such things as clarity and the ability to determine implications as well as the ability to show relationships between concepts and the knowledge structure. The discussion is kept to a minimum as the results of a survey, described in Section 8.2.1.4, act as a good summary of the author's intuitions of these various representations. The six representations are: an MCRDR KBS (Figure 8.11), a formal context or crosstable (Figure 8.12), a concept matrix (Figure 8.13), a line diagram (Figure 8.14), Gaines and Shaw's (1993) visual language (Figure 8.15) and Gaines' Exception Directed Acyclic Graph (EDAG) (Figure 8.16). Rules for Cendrowska's lens data are used in all diagrams for consistency and comparison. Each diagram is shown together with a brief description of its considered strengths and weaknesses.

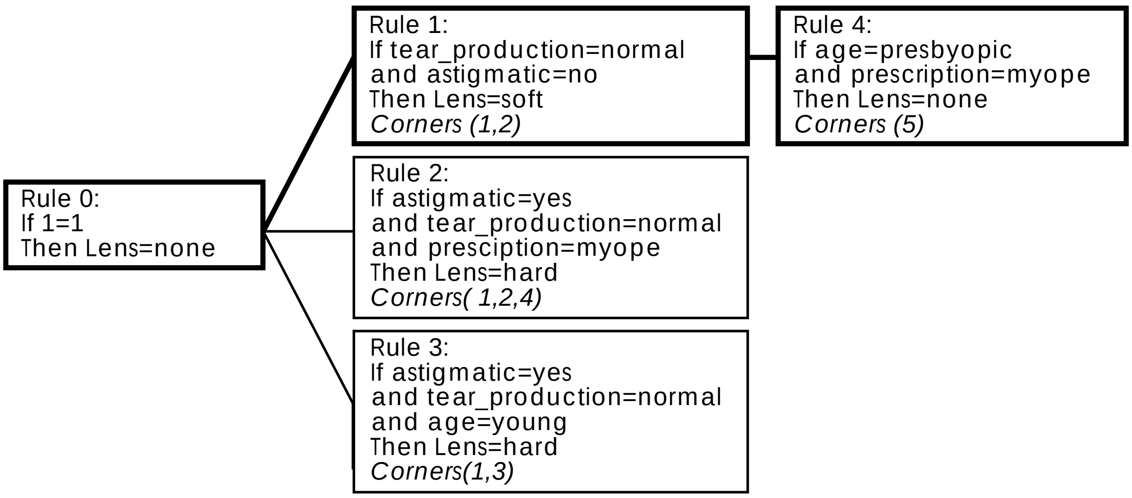


Figure 8.11. An MCRDR KBS for the Contact Lens Prescription Domain.
The highlighted boxes represent rules that are satisfied for the case {age=presbyopic, prescription=myope, astigmatic=no, tear_production=normal}. The classification given is Lens=none. As this domain only deals with mutually exclusive conclusions we only get one conclusion, but if the domain was extended to cover spectacles and bifocals then this case could lead to multiple conclusions being given. The cornerstone case numbers associated with a rule are shown as Corners().

The MCRDR KBS shown in Figure 8.11 shows the exception structure of the representation. An added feature of the MCRDR KBS is the storage of cases associated with each rule which is a valuable aid in providing the context in which the rule applies as well as an aid in KBS maintenance and validation. While the format of the diagram shown is only one possible way that the MCRDR structure can be visually represented, the diagram is adequate since it displays the key aspects that need to be shown which are the exception structure and the associated cases.

In Figure 8.12 the same rule pathways have been flattened into a formal context. The formal context, or crosstable, provides the benefits of a decision table structure. Each rule can be compared using the same features and in the case of rule bases that use intermediate conclusions the decision table format results in a normalisation of the data to the one base level. Decision tables are a useful format for further processing by a large number of techniques including rough set theory and formal concept analysis as described in Chapters 4 and 5, respectively.

	1=1[5]	astigmatic = no	tear_production = normal	age = presbyopic	prescripton = myope	astigmatic = yes	age = young
1-%LENSN	X						
2-%LENSS	X	X	X				
3-%LENSN	X	X	X	X	X		
4-%LENSH	X		X		X	X	
5-%LENSH	X		X			X	X

Figure 8.12: Context of "MCRDR Contact Lens Rules"

As described in this thesis the data/rules in the crosstable may be transformed into a concept matrix, which is shown in Figure 8.13, by using FCA to find the higher level concepts represented in the intersections of sets of attributes and the sets of objects that share them. The concept matrix shows some of the structure of the concepts but because some concepts are children or parents of more than one concept the ordering will affect the appearance of which concepts are related. This limitation is overcome in the line diagram or concept lattice. The researchers at Darmstadt do not seem to consider the concept matrix as a major representation but see it more as an intermediate representation of the preferred concept lattice. This study sees the concept matrix as more than just an intermediate representation and is particularly useful as a more concise representation of the data that in some situations may be more useful than the line diagram. For example, in the Lotus domain a comparison of the concept matrices revealed patterns and differences between the KBS developed by the four agricultural advisors as shown in Figure 5.15. To be able to visually detect differences in the patterns it is necessary to order the attributes so that they are in the same order in each KBS. Similar to the preliminary work done on comparing conceptual models in Chapter Six, the matrix can be further exploited to support automatic comparison of concepts.

As described in Chapter Five, the concepts in the concept matrix can be ordered and drawn as a complete lattice as shown in Figure 8.14. The main strengths of the concept matrix is the ability to see higher level concepts from the primitive rule concepts and the structuring of the concepts into a subsumption hierarchy.

---

[5] 1=1 is the condition in the default rule which give the default conclusion as shown in Figure 8.11.

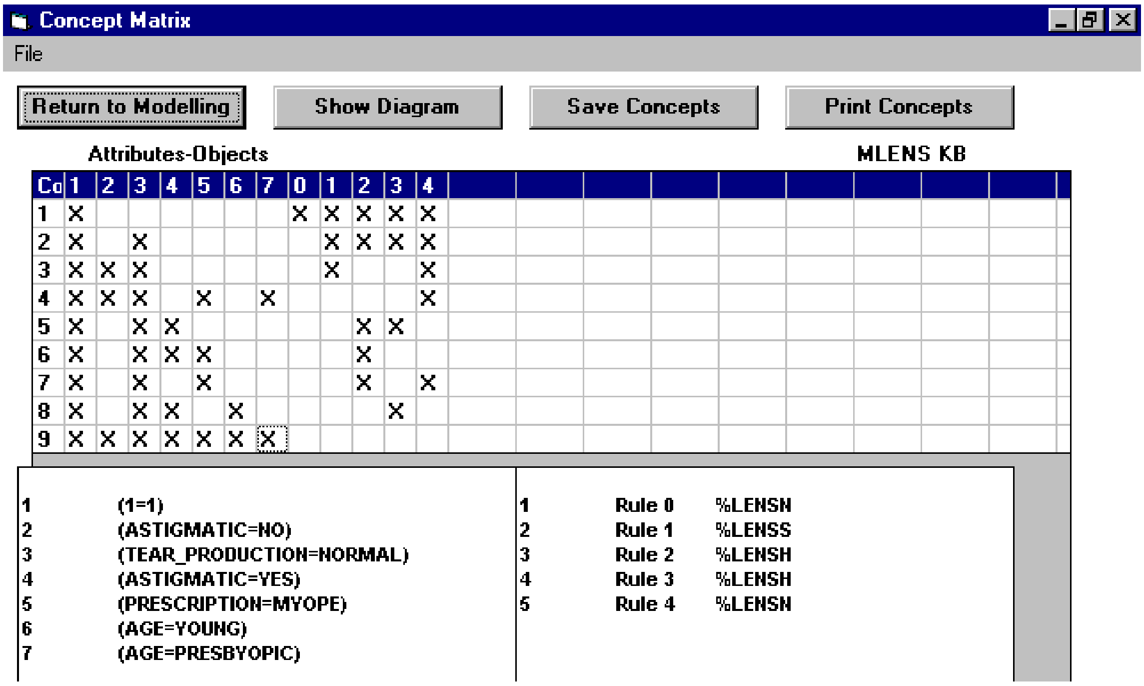


Figure 8.13: The concept matrix screen in MCRDR/FCA for the Formal Context "MCRDR Contact Lens Rules" given in Figure 8.12.

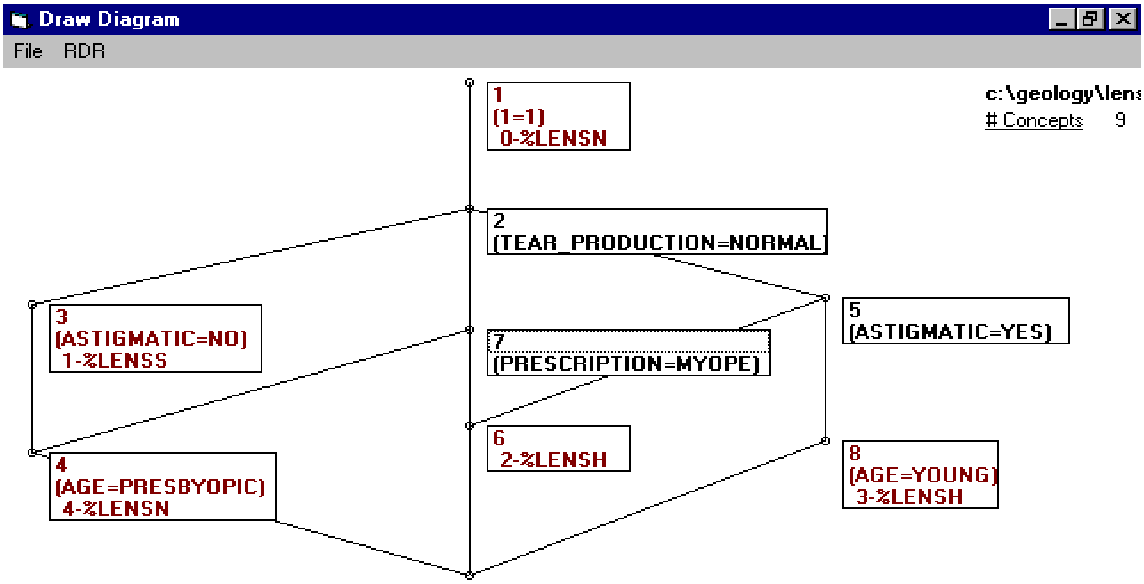


Figure 8.14: The Diagram Screen in MCRDR/FCA which shows the Concept Lattice for the Formal Context "MCRDR Contact Lens Rules" given in Figure 8.12.

The visual language described in Section 3.3.1.3 (Shaw and Gaines 1991b and Gaines and Shaw 1993a) and shown in Figure 8:15 offers a graphic representation of the subsume and disjoint logical relations. The visual language provides a domain model that is not present in the other representations and shows how the rules fit into the model. This provides a rich representation but as to be expected as more information is included the diagram becomes more complex compared to the other representations.

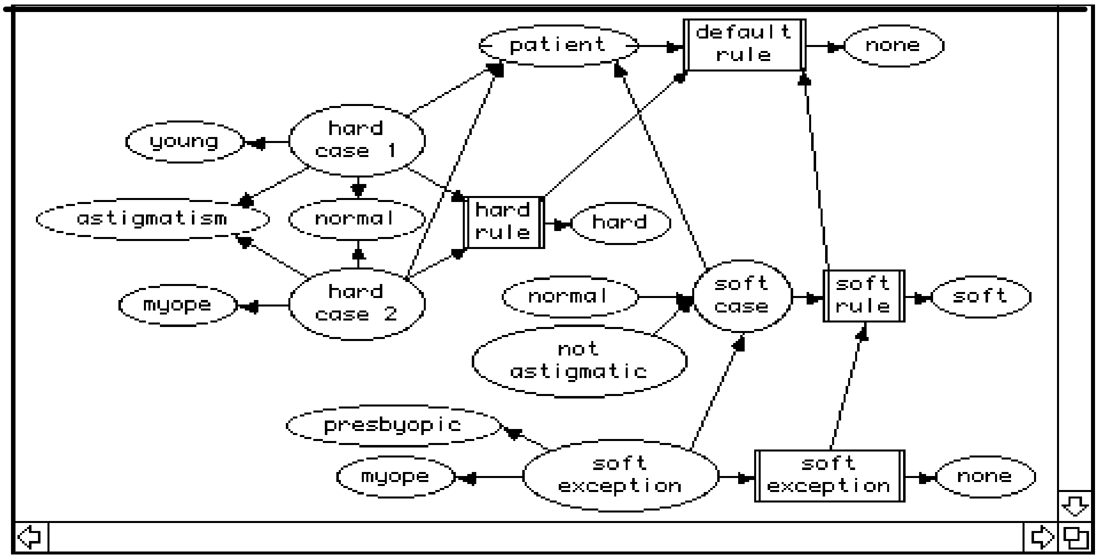


Figure 8.15 Contact Lens Rules Represented in the Visual Language (from Gaines and Shaw 1993a)

The final diagram to be considered is Gaines' (1996) Exception Directed Acyclic Graphs (EDAGS). The similarity between the MCRDR/FCA diagram and the EDAG in Figures 8.14 and 8.16, respectively, warrant a more in depth discussion of the EDAG and how they compare to the FCA line diagram. The development of the EDAG was motivated by the desire to develop knowledge representations that encode knowledge in a similar way to the expert (Gaines 1991a) and to make knowledge comprehensible (Gaines 1996). As a warning that the objectives stated by Miche and Quinlan may not always be achievable, Gaines notes:

> *"It is not obvious that an excellent performance system necessarily implies the existence of a comprehensible knowledge structure" (Gaines 1996, p. 206).*

Instead, compactness, coherence of the structure and the use of familiar concepts are all relevant when determining comprehensibility (Gaines 1996). Gaines considers the use of default reasoning and rules with exceptions to be a closer representation to the way experts reason than decision trees or production rules. Gaines (1996) describes how by generalising parts of the tree or generalising rules with exceptions it is possible to develop more compact knowledge structures. In the case of a tree representation, generalisation into partial orders results in branches rejoining and avoids the replication of sub-trees. The similarity with RDR is noted by Gaines with the difference identified as the inability of the RDR structure to allow multiple conclusions. This difference, however, does not exist with the newer MCRDR representation. Using the features of an

EDAG described in Gaines (1996) we now compare them to the MCRDR rule pathways and the MCRDR/FCA line diagrams.

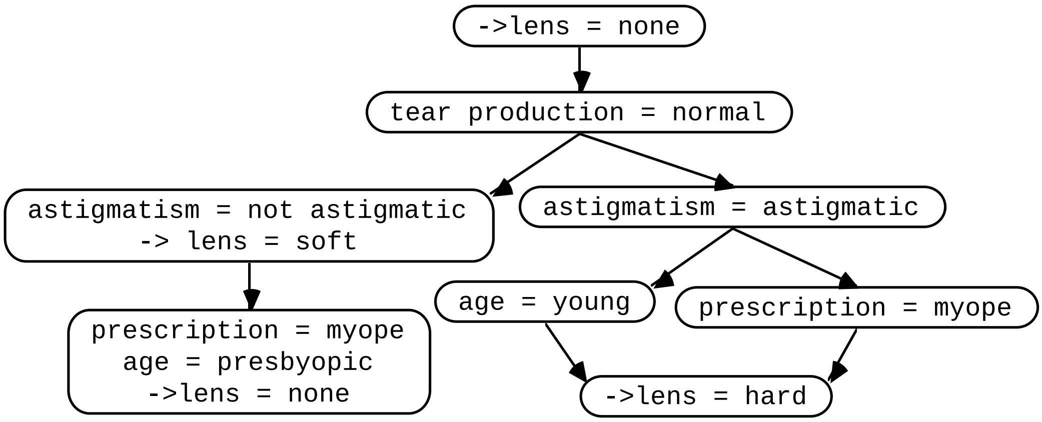


Figure 8.16 Induct rules with exceptions, factored as EDAG (from Gaines 1996)

1. An EDAG is not rooted and may have several disconnected parts. If a default rule is not used, which is not needed in many domains, an MCRDR is not rooted and each pathway at the first level could be treated as an individual disconnected part. A concept lattice is a complete network with a top and bottom root and is not disconnected.
2. Concepts in an EDAG and an MCRDR do not have to be mutually exclusive and multiple conclusions can occur. In both an EDAG and an MCRDR, a rule that has no exceptions will be concluded unconditionally if the premise is true.
3. The structure of an EDAG and an MCRDR is not binary but may have more than two nodes branching from a node.
4. The structure of an MCRDR is a tree, no branches may rejoin. However, the structure of a concept lattice which uses the intersections of sets of attributes and objects from the MCRDR rules is not a tree and, therefore, like an EDAG.
5. An MCRDR is made up of nodes which contain one or more conditions in the premise and one conclusion. A node in an EDAG or a concept lattice does not have to contain both a premise and a conclusion, but a premise may represent a common condition in a number of rules or a conclusion may represent a default or common conclusion.
6. A null node may be used in an EDAG to avoid crossing of lines. This approach is also used by the Darmstadt group in their drawing of concept lattices. Although this technique has not been used in the MCRDR/FCA diagrams it would be a satisfactory solution to reduce diagram complexity where line crossing can not be removed by repositioning nodes.
7. The notion of a premise as a potentially decidable predicate holds for both an EDAG and MCRDR. Possibly RDR (PRDR) was developed (Mulholland et al 1993) to handle an ion chromatography configuration problem where a premise could fire in a number of ways such as true (T), possibly true (P), unknown (U) or false (F).
8. The relationship between using an EDAG for inferencing by following the arrows and treating the EDAG as a semantic network when the arrows are reversed is compatible with the use of the MCRDR structure for inferencing and the concept lattice as a semantic network. By reversing the arrows in the EDAG or line diagram we can see the 'isa' relationships by following the inheritance of properties. Multiple inheritance pathways are treated as disjunctive concepts (Gaines 1996). In a

pathology system currently under development the MCRDR is more of an EDAG-like structure without an implicit OR.

9. The interpretation of a node in an EDAG, MCRDR and concept lattice all require tracing paths back to the root and consideration of the children nodes.

In this section a number of representations of the contact lens data have been shown including the MCRDR rules in Figure 8.11, a formal context based on the MCRDR rules in Figure 8.12, the concept matrix and lattice derived from them using FCA in Figures 8.13 and 8.14 respectively, the visual language of the PCP concepts and rules in Figure 8.15 and an EDAG in Figure 8.16. The similarity between the concept lattice and the EDAG was noted, with the concept lattice providing not only the inference paths but also an extensional definition as does the visual language in Figure 8.15. The MCRDR rule structure provides the inference paths more clearly, shows the cases associated with each rule and is the simplest representation but does not contain any of the higher-level concepts shown in Figure 8.15, which is the most complex diagram.

To evaluate my findings of the various strengths of these representations a number of computer science and engineering students and academics completed a survey of these six representations. This evaluation is found in part four of the survey which is described together with the results and discussion in section 8.2.1.4.

Section 8.2 Using Line Diagrams for Learning

The purpose of this section is to evaluate the usefulness of the line diagram for teaching and instruction. Two approaches to testing this claim have been taken. One is from the viewpoint of the novice the other is from the viewpoint of the expert. In the first evaluation a survey was used to discover how well a novice can use the line diagram for learning. In the second evaluation experts are used to comment on what has been learnt by a novice.

8.2.1 The Survey

A survey was conducted using computer science and engineering postgraduate students and academics that were attending a postgraduate workshop. Most of the participants were students from the Artificial Intelligence Department but all participants, except for one, did not know anything about line diagrams, formal concept analysis or the domains that were being used. Sixteen questionnaires, as can be found in Appendix B, were given out but only 12 were answered. In some cases questions were missed. The questionnaire was divided into four sections, each testing a different aspect of the line diagram. I piloted the survey on a Master of Applied Science (Information Studies) student who identified parts of the questionnaire that needed to be reworded and which parts needed a verbal explanation. Each section is now discussed. The survey was handed out as part of a computer science workshop for postgraduate students and academics. A slot of 25 minutes was allocated and, although the time was extended to 40 minutes over the lunch break, each section was rushed and insufficient time was provided to explain each section adequately, possibly affecting the quality of the results obtained.

8.2.1.1 Learning to read a line diagram - Section One of the survey

The first section of the survey was concerned with giving the subject some familiarity with the line diagram as well as testing how quickly the line diagram could be learnt to be read. I showed a line diagram on the overhead projector and explained briefly what a concept in formal concept analysis was and how the diagrams could be read. I selected two different concepts on the diagram to explain how the set of attributes and the set of objects for a concept could be found. Each subject was then required to look at their questionnaire giving the set of attributes and the set of objects for two concepts on one diagram and two concepts on another line diagram. On the questionnaire itself it was again stated how attributes and objects were found. Also an example using one of the concepts on the first line diagram was given which listed the set of attributes and the set of objects for that concept. About 8 minutes was spent on this section.

	WV	CS	MK	PC	SL	LK	JWK	AEL	MH	XYZ	MP	KD
Q1	6/6	5/6	6/6	6/6	6/6	6/6	6/6	6/6	5/6	2/6	4/6	6/6
Q2	5/5	5/5	4/5	5/5	5/5	5/5	4/5	5/5	4/5	3/5	4/5	5/5
Q3	6/6	5/6	5/6	6/6	6/6	6/6	5/6	6/6	6/6	3/6	4/6	6/6
Q4	7/7	5/7	5/7	7/7	7/7	7/7	4/4*	7/7	5/7	4/7	5/7	7/7

Table 8.1. The results for section one of the survey.

The denominator is the total number of attributes and objects. The numerator is the total number of correct attributes and objects listed by the subject. If extra attributes or objects were added they were subtracted from the correct number. Subjects 10 and 11 did not grasp what was to be done or how to read the diagram since extra information was given that indicated they were guessing.

* This subject missed the second part of the question which asked to list the set of objects.

As shown in Table 8.1, from the twelve subjects, six (6) got all of the descriptions correct. Another four (4) got most of the descriptions correct and two (2) obviously hadn’t grasped how to read the line diagrams because they wrote down extra things or left things out. So 10/12 gained a reasonable to good understanding of how to read a line diagram after a one minute verbal introduction and a brief written description and example in the questionnaire. In the pilot study I had been able to correct a misunderstanding which I was not able to do with this large group. I realised that misunderstandings at this point may jeopardise the other results. Two options were to correct the individual answers and discuss any misconceptions or show the group the corrects answers. However both options would reduce the time allowed on the other more interesting sections and the second option ran the risk of subjects changing their original answers which would alter the results.

8.2.1.2 Comparing Line Diagrams with Rule Traces - Section Two of the survey

The purpose of Section Two of the survey was to determine whether the line diagram provided information in a more concise and understandable form than browsing a rule trace. The questions to be asked had to be simple and answerable from one diagram since the subjects were novices in the domain and with the representation and the subjects did not have access to the tool to perform their own inquiries. Since the line diagrams were based on the rules in the knowledge base it was obvious that with investigation most questions could be answered from the rules. The higher level concepts would be harder to find by browsing the rules but even this information could be gleaned by a skilled person. This meant that the speed with which a question could be answered was the key aspect to be measured. It was not possible to ask someone to first solve the problem using a rule trace and then solve the same problem with a line diagram, or doing these tasks in reverse order, since they would already know the answer after they solved the problem the first time. It was decided to split the subjects into two equal groups, each group using either the line diagram or rule trace to answer the question. Two sets of questions were asked and each group was assigned to a different source for each question. For example if group B used a line diagram for question one they used the rule trace for question two.

Group A were given a set of sheets which included printouts of the rules for questions 1.1 and 1.2 in Section Two and a line diagram for answering questions 2.1 and 2.2. The other group, Group B, were given a line diagram to answer questions 1.1 and 1.2 and printouts of the rules for answering questions 2.1 and 2.2. Both groups were given the chance to use the different sources (but only one source for each question) so that there was no particular bias that some people were better at using one source over another.

I had some reservations over the nature of the rule trace that should be given. I decided that I would provide the rule trace that is available in MCRDR/FCA which is what is provided in the basic MCRDR for Windows system. I realised that having to look at five screen dumps as compared to one line diagram had an obvious disadvantage. To minimise the amount of paper shuffling I put two screens on a page so that there were three pages at which to look. This may also be criticised but the point is that it does take time to browse through rules and I had already cut down the user time since they did not have to search for the relevant rules and they had all the information in front of them at one time. In the real system you only got one rule at a time. Perhaps to reduce the paper shuffle I should have provided a dump of the rules in the format shown in Figure 8.17.

```
21 1 0 19 %PL000 : (GRAIN_SIZE = COARSE)
22 1 0 21 %VC000 : (GRAIN_SIZE = FINE)
23 1 0 22 %PL000 : (GRAIN_SIZE = MEDIUM) & (SILICA = INTERMEDIATE)
24 1 0 23 %VC000 : (GRAIN_SIZE = MEDIUM) & (SILICA = LOWISH)
25 1 0 24 %PL000 : (GRAIN_SIZE = MEDIUM) & (SILICA = VERY_HIGH)
```

Figure 8.17 A dump of the rules in the SISYPHUS III KBS that used the conclusion plutonic or volcanic.

However, the user does not have access to the knowledge in the format shown in Figure 8:17 as part of the MCRDR system so I could have also been criticised for not comparing against the existing facilities in the tool. Two different participants felt that the rule trace I had provided did not show the structure of the rules. This was what I was trying to demonstrate. The point I was trying to make is that by structuring the knowledge into a line diagram, showing the hierarchical relationships between rules and higher level concepts, we are better able to understand the rules and the relationships between rules. The assumption by those that criticised the nature of the rule traces offered was that the rules themselves had structure which provided information on the relationships between them. As can be seen in Figure 8.17, this was not the case with the examples used and none of the rules were exceptions to other rules besides the default rule. Another point is that I would have assumed greater familiarity by most subjects with the use of a rule trace than the use of a line diagram for browsing rules. I would not have thought that the page layout would have made a substantial difference especially since I showed an overhead of the rule trace and pointed out where to find the conditions, the conclusion and conclusion description before anyone commenced, even though each of these were labelled on the screen printout they received.

As noted earlier, 16 questionnaires were taken. Care was taken to give out equal numbers of Group A and Group B type questionnaires. However, only 12 were filled out and of these, 7 were in group B and 5 were in group A. This made comparison a little harder and the results are quite spread. Some people obviously had a reasonable grasp of the line diagram and others hadn’t worked out how to use it properly. With proper understanding of the line diagram these questions can be answered in seconds. For example question 2.2 which asks which attributes/conditions are shared by all concepts can be answered immediately by looking at the top concept.

The results for each question are given in Tables 8.2(a - f). The numbers shown in the tables are the number of minutes and seconds it took to answer the question. An ‘X’ marked in front of the time indicates the answer was wrong. A question mark shows that the question was not answered or the answer was incomprehensible. The results for each subject are shown in the same order in each figure for comparison. A brief synopsis of the results under each figure are given.

Question 1.1	Group A 0 Rule Trace						
	3:00	0:52	?2:13	1:00	1:45		
	Group B - Line Diagram						
	3:30	2:00	X2:12	X3:00	0.15	0.20	?2:00

Table 8.2(a): The time in seconds taken to answer Question 1.1 in Section Two.
The two fastest times and correct answers used the line diagram. Three of the long times using the line diagram got the wrong answer indicating a poor understanding of how to read the diagram. There were three reasonable medium times using the Rule Trace.

Question 1.2	Group A -Rule Trace						
	1:30	1:07	?1:36	2:00	1:15		
	Group B - Line Diagram						
	?1:30	?1:00	1:00	0:30	1:00	1:45	?4:00

Table 8.2(b): The time in seconds taken to answer Question 1.2 in Section Two.
This question required listing the conditions under which a medium-grained rock should be classified as plutonic or volcanic. The substantially shortest time is again using the line diagram, but most times using either method are similar. Note that it is a different person to the two people with the shortest times in question 1.1. Questions 1.1 and 1.2 both required searching through most of the problem space in either technique so that may account for inconclusive results.

Question 2.1	Group A - Line Diagram						
	0:45	0:48	0:17	1:00	0:45		
	Group B - Rule Trace						
	4:20	?	3:05	1:30	1:00	1:45	?4:00

Table 8.2(c): The time in seconds taken to answer Question 2.1 in Section Two.
The times are definitely the best using the line diagram.

Question Four	Group A - Line Diagram						
	0:25	0:38	X0:32	0:20	0:30		
	Group B - Rule Trace						
	0:45	?	0:37	X0:60	1:00	0:40	X4:00

Figure 8.2(d) The time in seconds taken to answer Question 2.2 in Section Two.
Again, the times are definitely the best using the line diagram. Questions 2.1 and 2.2 are more concerned with relationships between rules/concepts which may be why we see a dramatic improvement. The lowish times using the rule trace may occur because by the time you get to 2.2 the subject has possibly already worked out which attributes are shared when they were looking for the closest concept. It probably would have been better to put the questions in the reverse order.

As described in the commentary with each table, the results for the first question were inconclusive and may have been due to the nature of the questions which related directly to the primitive concepts and did not require any type of reasoning or abstracting to be performed. The second set of questions involved relationships between concepts and the structure of the rules and the results strongly favour the line diagram over the rule trace.

In addition to answering the above questions the subjects were asked to make optional statements regarding the rule traces and the line diagrams. These comments are included for interest in Figure 8.18.

General comments
The questions are ambiguous I feel. Does 2.2 mean that the conditions are shared in value or just both shared in the rule antecedent ? What is closer ?

Might have been fairer comparison with better description of the use of the MCRDR method. Your results should still work, I'd think.
What do you mean by a closer conclusion

Comments regarding rule traces -
No implied notion of closeness
Familiarity bound - takes time to learn how to read. Still harder to make easy comparison
Pretty incomprehensible
Impossible to read, but more due to the layout than rule structure.

Comments regarding line diagrams -
Not clear how to derive conclusions
Restricted to planar concepts (for humans) Note consider effect of line diagram practice versus no rule traces practice. Rule trace seems to have higher learning curve.
Seem to prompt intuition a bit better, though still not very understandable.
Poor layout of rules may make this test unfair. Also these line diagrams are simple. What happens if you need several pages ?
How do I know when something is a conclusive classification or just an intermediate step ?
I like the line diagrams more.

Figure 8.18: The optional comments regarding rule traces and line diagrams in Section 2.

The comments in Figure 8.18 tend to favour the line diagram over the rule traces but no conclusions or claims are made based on the comments. The comment that the rules appear to have a higher learning curve than the line diagram is interesting as it was assumed that most computer science graduates, in particular AI postgraduates, were familiar with similar representations.

8.2.1.3 Line Diagrams for Learning - Section Three of the Survey

This section was the hardest for the subjects and the most subjective to evaluate because it has the least structure or guidance and required some deeper understanding of how to read the line diagram and how to reason about it. Only eleven (11) of the 12 subjects answered this section.

Firstly, the subject was shown a line diagram using the Lotus agricultural domain and a description of what I could see from the diagram. They were asked, optionally, to comment on my analysis and then they were shown a different diagram from the same domain which they were asked to analyse and comment on. The comments regarding my description are shown in figure 8.19. My annotations are shown in italics. Generally, there is agreement with what I have said and in some cases there seems to be understanding of why I had made a particular statement. There is obvious difficulty with understanding the representation. In particular the labelling for objects (the rule number and conclusion code) was found to be problematic. These difficulties are to be expected as the line diagrams do take some time to understand and as was noted in Chapter Five, there are limitations on the appropriateness of the extensional definition. The hierarchical and semantic net nature of the line diagram is the hardest to grasp but is also what gives the representation its strength. It may have been helpful to replace the conclusion code with a brief description of the conclusion in the extension labelling although the description of each code was provided on the page that it was used.

Do the observations seem reasonable ?

Yes, but not entirely. For instance, you said that concept 2 is an indicator of poor conditions for Maku. Specifically that Low-PH and not Waterlog aren't good for Maku. But on the right, Maku is recommended in concept 6, but negated in 7, when the WATERLOG condition is added. This might mean that in concept 2, the waterlog condition is not important but the low PH is the actual indicator of poor conditions for Maku ?

Sort of, except that I don't understand why concept 4 is linked to concept 5 (*doesn't understand the subsumption hierarchy*)- we already know it shouldn't be Maku Lotus. Also it doesn't cover all situations.

Makes a lot of assumptions.

Underlined sentence - "this hypothesis could be tested further by looking at all rules that use these two conditions" - may also want to see whether there are other rules not recommending Maku.

XYZ- Initially it is hard to say (what distinguishes a concept from a conclusion). Even after a few minutes this is hard to understand for a lay person. Arcane notation (such as 13-%CLOVR) is well-high impossible to remember (*but I have it on the same page ?*), very distracting and off-putting and requires learning. Of course, once learnt, it can be used to judge the reasonableness of the conclusions (I imagine) without much difficulty. But there is little incentive to learn it.

MH - yes

AEL - Does it follow that given 2 (not 3) implies S?
LM- Yep
SL - The last statement may not be true, as you have explicitly filtered for rainfall, but low precipitation areas may still be fertile, to be independent of rainfall.
PC- okay but very hard to tell. There is a potential conflict between 2 3 5 and 6 - can subsume

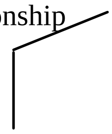
Figure 8.19: The comments regarding my analyses of a line diagram in section 3.

It was interesting to see if others could follow my reasoning based on the line diagrams and the comments in Figure 8.19 indicate that this was possible. The comments showed that the line diagrams could be used to initiate discussion about a domain and to suggest further aspects of the domain knowledge to be considered. This is consistent with the claim that the line diagrams can be used to assist in the negotiation process required during conflict resolution of requirements as suggested in Chapter Six. Further comments from the subjects which allowed for free comment are shown in figure 8.20 which primarily contain comments regarding interpretation of the line diagram.

Further comments ?

Why do things have to connect to the dot at the bottom ? It just makes the diagram more confusing (same person that didn't understand the lattice as a subsumption hierarchy).
What if I get stuck midway in the tree eg rainfall < 1200 but > 1000 ?
Should the bottom node be labelled %NC000.

AEL - on surface may many missing cases. However, some of these may be impossible
Guessing 

'or' relationship 

'and' relationship

is this mutually exclusive ?

SL - Why is the box '5' there at all ? 4 already concludes Shalw, 2 conclude nothing, so you could move the conclusion from 5 to 2?? This, if true, would also mean that '2' is not a high level concept. Actually what is a primitive concept ?

PC- These probably only become useful if there was a motivation for learning. Otherwise pretty hard.
WV- Much of it seems to be reasonable. However, there seems to be insufficient respect given tot he symmetry between the two ancestors of concept 5 - the suggestion is that concept 2's application has priority over the (possibly redundant) concept 4; I imagine this is a mistake. I would certainly agree with the last paragraph above- that the information is incomplete.

Figure 8.20: The optional comments regarding the use of line diagrams for learning in section 3.

However, it was the next part of section three that was of greater interest to this survey. In the next part of section three of the survey the subject describes their understanding of the domain from the line diagram. To me the most important thing is not how correct the analysis is but the fact that all 10 of the 11 subjects were able to make some comments regarding the domain knowledge. The comments are reproduced in Figure 8.21. From the comments it can be seen that eight of the subjects seem to have gained some knowledge of the domain and four of these have got a feel for the structure and order in which decisions should be made. A major benefit is not just the answers it can give but the questions that it raises.

KD- Well this graph blows my last analysis out of the water, since concept 2 explicitly says that Low-PH is good for Maku. Low-Ph =no indicates the conclusion %Salin, I guess, but further expansion of the graph to cover the not-waterlogged case should be considered. (*good for hypothesis testing*)

CS- Concepts above subsume concepts below.
If low-ph = yes then Maku except
  if waterlog=no then Sharn except
    if rainfall >= 1000 then Sharn=no then Kenya
    else if rainfall <=1000 and soil then Shalw
  else if flooding >=3 then flood except
else if low-ph and waterlog=yes then Salin

MP- There are no higher level concepts. Ph does not affect Maku. (*This was a subject who had difficulty reading the line diagram in part one of the survey)*

XYZ- I'm not sure what this tells me.

MH- All groups have a classification associated.
Ph is a high-level attribute
- with low_ph=no and waterlog, %Salin - Maku Ok, but others should be considered.
- missing knowledge - low-ph (=no) and not-waterlogged.
- low_ph is okay for maku
  unless not waterlogged - in which case Sharnae or Goldie
   unless
    rainfall > 1000 and not saline - Kenya
    rainfall < 1000 and soil_depth <=30 - Shalw
  flooding >=3 then Flood.

AEL - Low-Ph is good for Maku. Flooding is bad but the soil should be waterlogged. If not, Sharnae is a good alternative, except in low-rainfall and shallow soil. With higher rainfall and non-saline soil, Kenya grass should be considered. In Non-low_ph soil that is waterlogged, Maku can be used, but other legumes may be better.

LM - Maku is suitable in Low-Ph areas that do not flood/waterlog i.e. it's better in dryer areas unless there is shallow soil. (left hand-side).
Maku may however be okay in wetter soils with high_ph if used in combination with other legumes.

SL - Ph seems to have no bearing on Maku, No conclusions based on Ph suggest to plant Maku. (*didn't understand the diagram).*

PC- Maku is useful for low-ph areas, but there are plenty of exceptions - hard to get consistent the theory.
MWK - If there is not a low ph and the land is waterlogged, then consider strawberry clover and other legumes. Otherwise, in general Maku is recommended. However, if there is flooding , don't use maku lotus. Care must also be taken if the area is not waterlogged, in which case Sharnae or Goldie should be used, unless it's not saline and it rains a lot in which case you should use Kenya Clover. Alternatively, if it doesn't rain and the soil is deep don't use Maku Lotus.

WV- Obviously if acidity is low (high_ph) we would hope for waterlogging - otherwise we would not know what to do.
Otherwise, if there's no waterlogging - go for Sharn else -----

Figure 8.21: The subjects observations from the line diagram they were asked to analyse and comment on from the LOTUS agricultural domain.

8.2.1.4 Comparing Representations - Section Four of the Survey

Six different representations of the rules which cover Cendrowska's lens data, which were presented in Section 8.1.2.2, were shown to the subjects using an overhead projector. A printed copy of each diagram was taped to the top of the table and when the subjects were moving off to lunch they were asked to circle a number between 1-5 for each diagram along four dimensions. The completed surveys were then to be left on the table. Note that only 11 subjects filled in this section of the survey and the last two aspects were missed by one subject (these questions were on the overleaf page). The six representations were:

Diagram 1 - MCRDR KBS (Figure 8:11)
Diagram2 - Formal Context (crosstable/decision table) (Figure 8:12)
Diagram 3 - Concept Matrix (Figure 8:13)
Diagram 4 - Concept Lattice - Line diagram (Figure 8:14)
Diagram 5 - Gaines' Visual Language (Figure 8:15)
Diagram 6 - Exception Directed Acyclic Graph (EDAG) (Figure 8:16)

A brief description of how to read each diagram was given when the diagram was shown on the overhead projector. The scores for each representation are shown individually by feature and an averaged score is given and shown in Tables 8.3(a-f). An overall rating is given which is the sum of the averaged scores. Using the overall score, the diagrams in order of preference are the EDAG, the MCRDR, the line diagram, Gaines and Shaw's Visual language, the crosstable and the concept matrix. Table 8.4 provides a summary of each representation by feature. It is noteworthy that the line diagram scored the highest for structure, which has been the claimed strength of the line diagram in this thesis. The poor score for the concept matrix (Diagram 3) was expected because the representation required more description to really understand, but there was insufficient time, and because the ability to reveal patterns which can be compared was not being evaluated here.

Diagram 1												
Pleasing appearance	2	3	2	4	5	5	2	2	4	4	5	3.45
Comprehensibility	4	3	4	4	5	5	3	2		4	4	3.80
Good Structure	-	3	4	4	-	4	3	3	4	5	3	3.66
Able to determine Implications	-	4	5	4	-	5	5	2	4	5	4	4.20
												15.11

Table 8.3(a): The results for the 11 subjects for MCRDR KBS by feature.

Diagram 2												
Pleasing appearance	1	2	2	2	1	3	2	3	4	2	2	2.18
Comprehensibility	2	5	3	1	1	4	3	4	-	2	1	2.60
Good Structure	-	4	3	4	-	3	3	3	4	3	2	3.20
Able to determine	-	2	2	1	-	3	3	4	4	3	1	2.55
Implications												10.53

Table 8.3(b): The results for the 11 subjects for the crosstable by feature.

Diagram 3												
Pleasing appearance	1	1	3	1	1	2	3	2	1	1	1	1.27
Comprehensibility	1	3	4	1	1	2	3	3	1	1	1	1.91
Good Structure	-	4	4	4	-	3	3	2	1	3	2	2.80
Able to determine	-	1	3	1	-	3	3	2	1	3	1	2.00
Implications												7.98

Table 8.3(c): The results for the 11 subjects for concept matrix by feature.

Diagram 4												
Pleasing appearance	4	3	3	4	4	3	4	3	3	3	4	3.45
Comprehensibility	3	3	4	4	4	3	4	2	3	2	3	3.18
Good Structure	-	4	5	4	-	4	5	3	3	4	3	3.88
Able to determine	-	4	3	4	-	3	4	3	3	4	3	3.44
Implications												13.95

Table 8.3(d): The results for the 11 subjects for line diagram by feature.

Diagram 5												
Pleasing appearance	1	3	4	1	4	3	4	4	1	3	3	2.80
Comprehensibility	1	4	4	1	2	2	4	3	1	1	2	2.27
Good Structure	-	3	4	4	-	4	3	3	1	5	1	3.10
Able to determine	-	4	3	1	-	4	4	3	1	2	2	2.66
Implications												10.83

Table 8.3(e): The results for the 11 subjects for the visual language by feature.

Diagram 6												
Pleasing appearance	5	5	5	4	5	3	4	4	3	4	2	4.00
Comprehensibility	5	5	5	4	5	5	4	4	3	3	2	4.09
Good Structure	-	3	5	4	-	4	4	4	4	5	1	3.77
Able to determine	-	5	4	2	-	4	5	4	4	4	4	4.00
Implications												15.86

Table 8.3(f): The results for the 11 subjects for the EDAG by feature.

	MCRDR	Crosstable	Concept Matrix	Line Diag	Visual Lang.	EDAG
Pleasing appearance	3.45	2.18	1.27	3.45	2.80	4:00
Comprehensibility	3.80	2.60	1.91	3.18	2.27	4:09
Good Structure	3.66	3.20	2.80	3.88	3.10	3.77
Determine to	4.20	2.55	2.00	3.44	2.66	4:00
Implications	15.11	10.45	7.98	13.95	10.83	15.86

Table 8.4: The combined averaged results for the 11 subjects.

8.2.1.5 Discussion and Conclusion

A comprehensive discussion on the use and value of visual languages and representations is offered in Kremer (1998). Kremer (1998) remarks that visual representations have been found to offer noteworthy advantages over linear textual representations. He cites Nosek and Roth (1990) who have found semantic networks to be more "computationally efficient" (comprehensible) than predicate logic. Visual representations have also been useful for visual programming, decision making, knowledge representation, database queries and more. Even the use of indentation in programs has found to be an important aid in program comprehensibility (Smith 1977). As shown in the small survey described in this chapter, reading graphical representations is not always easy and can be considered an "acquired skill" (Petre and Green 1993). In the Petre and Green study it was found that their subjects could use the textual representations more quickly but the graphical representations offered a much richer representation once the subject had passed through the learning curve. Kremer (1998) lists the arguments of Smith (1977) which describe the psychological benefits of visual languages for programs and data. These arguments include:

- abstract reasoning is pictorial and thought may involve manipulation of images,
- linear verbal language only provides one dimension but visual reasoning uses two or three dimensions;
- visual images assist "chunking" and short-term memory;
- images tend to be analogical.

Charles Sanders Peirce, who developed a visual logic language known as *existential graphs* on which Sowa (1984) based his Conceptual Graphs, states:"

> *I do not think I ever reflect in words: I employ visual diagrams, firstly because this way of thinking is my natural language of self-communion, and secondly, because I am convinced that it is the best system for the purpose {Ms 620, p. 8] (Roberts 1973, p.126).*

Kremer (1998) also makes the point that to date there is little hard evidence of the benefit of visual representations and that most interest is due to an intuitive belief that visual representations offer many benefits over textual or linear representations. The survey reported in this section has attempted to gain some evidence of the value of the line diagram representation over the linear textual rule trace. As noted by Kremer (1998) and evidenced in Petre and Greens's (1993) study, use of a visual language requires time and effort and this makes evaluation of the line diagram with novices a difficult task. Despite the difficulty, in this section domain and line diagram novices have been used to interpret and evaluate line diagrams from a number of viewpoints and the line diagrams have been compared with other representations and rule traces. The number of subjects does not warrant the application of statistical techniques to determine if the results are significant and at what level of confidence.

In summary, the survey found that 10 out of the 12 subjects were able to learn to read a line diagram within a few minutes, that the line diagram was easier and faster to use than a rule trace in answering questions about the knowledge base and that even novices could reason about the knowledge using the line diagram allowing the tool to be used for such purposes as hypothesis testing and tutoring. Given the time restriction for conducting the survey the results can be considered promising and generally supportive of the claim that the line diagram is a reasonable representation of the knowledge base and that they can also be used for explanation and teaching.

8.2.2 An evaluation by experts

In the previous section we can see that the line diagram can be used to gain some understanding of the domain. The purpose of evaluation with experts was to see if the type of knowledge that could be discovered was accurate and useful. The four domains introduced in Chapter Five have been used in this evaluation. Thus, four different experts have been involved in these evaluations. Access to all four experts has been difficult mostly due to availability and geographical separation and meant that the evaluations had to be conducted in one meeting or discussion. In each subsection the nature of the interaction is described.

8.2.2.1 The Blood Gases Domain

The 105 Blood Gases KBS was developed by a pathology expert over two years prior to this evaluative study. Since development of the KBS, the expert had moved to Perth, some 3,000 kilometres from the author. The opportunity for a meeting occurred while the author attended the AI'97 conference which was being held in Perth. This evaluation was the first performed and to a large extent the nature of the outcome and the best presentation to the expert was unclear. As preparation for the meeting 11 line diagrams were drawn based on selections the novice, myself, thought might be useful. In a real teaching system the student would be more directed and probably be given questions to pursue or have questions that had arisen from other learning situations such as reading or a lecture. I made a brief synopsis of each diagram in one to ten sentences. I then removed any reference to particular parts of the diagram to be left with one and a half pages of written description of what I had learnt about the domain. This one and a half pages was given to the expert for comment. This format was also used in the classification domains - Lotus and Sisyphus III.

The blood gases KBS I had been given did not include a description of the conclusion codes. I did not worry about this because I thought the expert would know what the codes stood for since he had created them. I also thought that since the description often includes the reasons for the conclusion I would be less biased or influenced by the description and my ability to learn would be based on the rules only. The observations I made and handed to the expert are shown below in Times New Roman single-spaced in 10-pt. The verbal comments of the pathologist are interleaved with each different observation and are shown in a different font (Arial). I have added further comment regarding the expert's response in italics. Each excerpt has been numbered for easier reference.

1. For the %MC002 conclusion there appears to be an inverse relationship between the value of BLOOD_PH and BLOOD_BIC and the two appear to be causally connected. For this conclusion when BLOOD_PH is HIGH, BLOOD_BIC is LOW. Conversely, when BLOOD_PH is LOW, BLOOD_BIC is HIGH. Also if the BLOOD_PH level is increasing then the BLOOD_BIC level must be decreasing. It also appears that if the BLOOD_PH level is NORMAL then the value of BLOOD_PC02 needs to be checked. If BLOOD_PC02 is high then the %MC002 conclusion should be given but if it is low then the value of BLOOD_BIC must also be low to receive this conclusion.

The expert could not remember what the conclusions stood for which made it difficult to say if my analysis was correct. The analysis I gave seemed unusual and he was trying to make sense of it. After a while, he thought that %MC002 was metabollic compensation. This was a weird conclusion that could occur in two quite different circumstances and he had previously decided that if he built the KBS again he would have treated this conclusion differently. This was why it was using High Blood_Ph with Low Blood_Bic or Low Blood_Ph with High Blood_Bic.

It was unexpected that the expert did not remember what the codes were for. I had assumed that he had not bothered with a description of them because he knew them so well. The comment that I had picked a weird conclusion was interesting because it shows the value of the tool for validation in detecting anomalous parts of the KBS. I had not been aware of the anomaly. The expert seemed reasonably impressed with my being able to analyse the rules and said I was a "human induction engine". The expert suggested that I was making life too hard for myself (and him probably) by not knowing the meaning of the conclusion codes. He suggested that I try again to learn using the conclusion descriptions. However, the opportunity to meet again did not seem likely so we carried on with the material available. When I later found the conclusion codes descriptions, it turned out that %MC002 stood for "Metabolic compensation.2" so that the comments by the expert was appropriate.

2. From the previous discussion, I was interested in finding out what conclusion would be given if BLOOD_PH was normal, BLOOD_PC02 was LOW and BLOOD_BIC was HIGH. I did a search on this combination but it was not contained in any rule. I then did a search on HIGH(BLOOD_BIC)=TRUE but there were no rules in addition to rule 15 which only considered LOW(BLOOD_PH)=TRUE and HIGH(BLOOD_BIC=TRUE and gave the %MC002 conclusion. This made me consider what was the effect of LOW(BLOOD_PC02)=TRUE. So I have entered that as the selection.

The expert thought that it was unusual that there would not be a rule to cover alternative situations when Blood_Bic was high.

It can also be argued that the tool was useful for determining missing knowledge.

3. It can be seen that if LOW(BLOOD_PC02)=TRUE the next most important attribute to consider is BLOOD_PH. If BLOOD_PH is abnormal it should be given the conclusion %RK002 or %RC002 depending on whether the value is HIGH or LOW, respectively. A number of rules which are part of the %MC conclusion family and include the conditions NORMAL(BLOOD_PH)=TRUE and LOW(BLOOD_BIC)= TRUE. The actual conclusion assigned depends on the actual PH values and if BLOOD_BIC is increasing.

The expert agreed with this paragraph and gave it two ticks.

Selected conclusion family %RK. The key features of the case to consider for this class of conclusions are BLOOD_PH and BLOOD-PC02. All rules are concerned with BLOOD_PH being high or having been high (over 7.45) except for rule 30 which concludes %RK002 when BLOOD_PH is NORMAL, BLOOD_BIC is LOW and BLOOD_PC02 is LOW. As was noted earlier there appears to be an inverse causal relationship between BLOOD_BIC and BLOOD_PH and the fact that BLOOD_BIC is LOW may indicate that BLOOD_PH is expected to become HIGH. If BLOOD_PH has been high at any stage then an increase in BIC and decrease in PH and BIC will give the conclusion %RK004.

The expert thought this paragraph was okay but had a problem with the last sentence. While he was seeing a patient, I looked again at the diagram and what I had written and saw that “then an increase in BIC” in the last sentence should have read “then an increase in PC02”. When I changed this the expert was satisfied but was not completely sure as he could not remember what the meaning of conclusion code %RK004 which I later found meant “Resolving respiratory alkalosis”.

The expert said that I had a misunderstanding of the relationship between Blood-Bic and Blood_PH. He commented that if you told an expert that Blood_Ph was low they would say the Blood_Bic should be low. The conclusion %MC002 for metabollic compensation resulted in the situation being reversed and unusual so it was misleading me into thinking that this is a common occurrence.

This was useful in identifying and correcting a misconception that I had. I had formed a hypothesis of an inverse causal relationship between Blood_Ph and Blood_Bic but this was generally not so and had only occurred because of an anomalous situation.

5. The conclusion family %MK was selected. The attributes to consider for these conclusions are BLOOD_PH, BLOOD_BIC and BLOOD_PC02. The two rules for %MK003 use the conditions DECR(BLOOD_PH)=TRUE and MAX(BLOOD_BIC)>=27. If BLOOD_BIC is decreasing then the conclusion %MK003 should be given otherwise the value of BLOOD_PC02 should be considered. The concept “HIGH(BLOOD_PH)=TRUE and NORMAL(BLOOD_PC02)=TRUE” may represent a higher concept that could be labelled as indicative of a particular type of patient condition.

The expert said that HIGH(BLOOD_PH)=TRUE and NORMAL(BLOOD_PC02)=TRUE did represent a higher concept which could be described as there is something wrong that needs further investigation.

6. The values used by the %RK and %MK conclusion families are similar and indicates that there is a close relationship between these two families

The expert pointed out that the use of K in the classification code meant that the PH was high. Whether the conclusion was R or M depended on a refinement of the initial rule/analysis based on Ph.

It was therefore natural that these two conclusion families were related since they were both based on a High PH. I had not realised such a naming convention. This also takes us back to point 4 where it was noted that %RK002 was the only rule that didn't require BLOOD_PH to be HIGH when all other similar rules did have this A-V pair. However, the coding used by the expert for the conclusion code has implied that this conclusion typically applies when BLOOD_PH is HIGH even though it was not present in the case that prompted the rule to be added.

7. Selected clause MIN(BLOOD_PO2). Only the rules which conclude %OX006 use this attribute and function. The key to this conclusion is a low BLOOD_PO2 level. The only other attribute that is considered is seen in rule 42 where the condition AGE<=70 is included. MIN(BLOOD_P02) <=80 is the highest value given, but the values range from 59 to 80 with 6 different values used.

The expert asked me if I was able to determine the purpose of the age attribute. I looked for a while at the diagram but not knowing what the conclusion codes were and without the tool to make various inquiries I could not determine the effect of age.

As the expert pointed out, if I had the meanings of the code I could have seen how age changed the conclusion given. After returning to Sydney I tried to see if I could answer this question. Since age was only considered in conjunction with the attribute BLOOD-PO2 I selected all the rules that used BLOOD_PO2 which gave me all of the rules for the %OX conclusion family. Looking at the description of the conclusion codes I could see that the %OX conclusion family all dealt with conditions related to oxygen and thus assumed that BLOOD_PO2 is some type of measurement of oxygen content in the blood. My observations regarding age are given in point 11 and are followed by the experts comments.

8. The %AB conclusion family first considers whether NORMAL(BLOOD_PC02)=TRUE and then whether BLOOD_PH is NORMAL or HIGH. It appears that %AB001 may indicate that everything or a certain condition is normal and %AB002 may indicated that the condition is abnormal.

The expert said this was plausible but he couldn't confirm this without the conclusion code descriptions.

When I later found the code descriptions it turned out that my assumptions were correct. %AB001 stood for "No disturbance of acid base status" and %AB002 meant "Mild respiratory alkalosis".

At a later date I sent the following email to the expert to clarify a couple of the outstanding issues raised.

This first comment concerns point four and the conclusion %RK004. The expert's comment are again interleaved.

9. If BLOOD_PH has been high at any stage then an increase in PC02 and decrease in PH and BIC will give the conclusion %RK004-""Resolving respiratory alkalosis".

Yes that seems right. Although I would qualify it that if the PH has been high AND THE PCO2 WAS LOW AT THE SAME TIME and THE BIC WAS NOT HIGH (at least this would be the usual case)

10. When I saw you, you asked me if I could see how the age affects the conclusion given. The only attribute that was affected by the age attribute was BLOOD_PO2. I therefore selected all the rules that used BLOOD_PO2 which gave me all of the rules for the %OX conclusion family. Looking at the description of the conclusion codes I could see that the %OX conclusion family all dealt with conditions related to oxygen and thus assumed that BLOOD_PO2 is some type of measurement of oxygen content in the blood.

Yes.

11. If BLOOD_P02 was normal the conclusion %OX005-"Normal Oxygenation" should be given except where MIN(BLOOD_PO2)<=75 (rule 37) or <=74 (rule 42) which results in the conclusion %OX006-"resolving hypoxaemia". However, if MIN(BLOOD_PO2) is between 74 and 80 then the age must be taken into consideration. If the age is <=70 then the conclusion %OX007-"hypoxaemia for this age" is given. It appears that even though the BLOOD_PO2 level is within the normal range it is a bit low for patients under the age of 71.

Exactly right. Would you like a job as an anaesthetist?

The comments of the expert in this domain provided some support that the line diagrams were useful for learning about the blood gases domain. Terminology, including the absence of conclusion code descriptions, appeared to be my main impediment to learning and it seems that in such a domain some initial training at least regarding the domain terms would be needed to learn further from the KBS. The value of knowledge learnt is not clear although it is obvious that the line diagrams provided a focal point for discussion which otherwise would not have existed.

8.2.2.2 The LOTUS Domain

The KBS for this domain were kindly supplied by Dr. Zvi Hochman, an agricultural scientist working for CSIRO in Toowoomba. I had had no discussions with Dr. Hochman regarding Lotus but I had met him on a couple of occasions and he agreed to review my observations. Since we were geographically separated I sent the following email message to him which is shown in Times New Roman - 10 point. His comments are shown in Arial 12 and my remarks on his comments are shown italicised in *Times New Roman 12 point*. I have included some introductory remarks for Dr Hochman since we had not had previous opportunities to discuss the background to this work. In the discussion on comparing viewpoints I have removed the names of the individual KBS and replaced them with more generic labels, that is, Lotus 1-4. Dr. Hochman had requested previously that the identities of the experts used in the four KBS remain unknown.

The following comments are observations made from line diagrams which show the hierarchical structure between concepts in a knowledge base. The key benefits of this structure are the higher level concepts and abstractions that are uncovered from the lower level rules. In the Lotus domain there are not many higher level concepts as the rules tend to be self-contained with few intersections of rule conditions with other rules. Using different views of the knowledge, however, we are able to ask questions such as: how does rainfall or PH affect the decision; what other conclusions are close to a specified conclusion and what are the differences between the different KB.

The reason for asking you to add your comment to my observations is to offer some evaluation of the claim that I can use line diagrams to assist me in learning about a domain. Obviously much of what is stated below can be found in the rule traces but these are difficult to structure and relationships between concepts are not easily found. I do not expect to have uncovered any new insights into Lotus but would like to know if my comments appear to be consistent with the domain knowledge. Put as much or as little comment as you feel necessary along the way and perhaps add at the end your comments regarding this whole exercise.

If you would like to pose a question regarding this domain which you feel I should be able to uncover in the KBS then please do so.

I begin with comments for the combined Lotus KBS with some comments regarding the individual KBS at the end. I must add that the conclusion code long descriptions provided in the combined KBS were so comprehensive I could possibly just have used them. In my analysis I did not look at these descriptions but later looked at them and generally found them to echo what I had found. However, the fact that I could get such information without the use of the diagrams is not a strong point in terms of internal validity of this evaluation.

OBSERVATIONS

1. The important feature in choosing the conclusion %SALIN- "Maku OK. Consider Strawberry Clover" is the condition "waterlog = yes" as both rules which give this conclusion share this condition. Additional considerations in choosing this conclusion are the soil must be saline OR the PH must not be LOW.

2. The importance of WATERLOG=YES prompts me to consider what other rules use this condition. Using this as my basis of selection I obtain an additional rule - rule number 21 - which gives a null conclusion if the RAINFALL>=1200 and the FERTILITY=LOW. This would suggest that strawberry clover does not do well if the ground is highly waterlogged and the ground only has low fertility.

3. The conclusion - %SHALW - Maku is unlikely to persist through a prolonged dry period" depends on two different sets of criteria as shown in rules 12 and 22. Rule 22 considers that the rainfall must be below 1000mm and the soil depth below 30mm. Rule 12 looks at a similar feature to low rainfall by specifying that the ground should not be waterlogged. However rather than considering soil depth rule 12 considers the soil PH, which should be low. As is shown in the conclusion description the key to this conclusion is the amount of moisture in the soil which is identified in both rules.

4. Next we look further at the impact of LOW-PH on the conclusion. When PH is low a number of conclusions are possible. Regardless of other features, low PH is an indicator that the soil is suitable for MAKU- with the conclusion %MAKUU -Maku lotus is recommended" being given in rule 2. Other recommendations are also offered when additional conditions exist. If the soil is not waterlogged then the conclusion "%SHARN- Maku OK. Consider Sharnae or Goldie" should be given which indicates that Sharnae or Goldie are other possibilities in dry conditions. A further exception to this is the situation where the soil is not waterlogged but there is still plenty of rain. If in addition the soil is not saline then the recommendation "%KENYA- Lotus is suitable but you may consider Kenya Clover too" may be another alternative crop to plant. From the long description offered for this conclusion - "In high rainfall and on the better soils Kenya clover may provide a higher quality pasture" it appears that if these conditions exist it would be preferable to plant Kenya clover over Sharnae, Goldie or Maku. Where the soil is not waterlogged but the rainfall is <= to 1000mm and the soil-depth is <=30 Kenya would not be recommended and a warning regarding the dry conditions is issued in the conclusion %SHALW- "Maku is unlikely to persist through a

prolonged dry period".Alternatively if the soil is waterlogged (as indicated by Flooding>=3), Maku will not survive and the conclusion %FLOOD- Lotus cannot tolerate 3 or more days of flooding in summer" will be given.

5. It would therefore appear that if the ideal conditions of low-ph, not-waterlogged, adequate-rainfall and non-saline-soil all existed Kenya clover should be planted otherwise plant Maku., Sharnae or Goldie but watch out for either too much or too little rain.

6. If the soil PH is not low and there is plenty of water the recommendation %SALIN - Maku OK. Consider Strawberry Clover" suggests that Maku and Strawberry Clover are both suitable. Alternatively, regardless of the PH, if the rainfall is adequate, but not resulting in the soil being waterlogged, it would be worth considering white clover. If the soil-fertility is low, fertilizer should be added for white clover. One might also assume that if the soil FERTILITY was not LOW then a different conclusion which also recommended to consider white clover without the need for fertiliser should exist. If the soil is waterlogged then the options offered are MAKU or Strawberry Clover if the soil is either saline or the PH is low. If in addition to being waterlogged, the rainfall is high and the soil has low-fertility then no recommendation is given.

A number of attributes such as (RYEGRASS>=15); (SOWING_DEPTH>=15); (SOW_MOISTURE<=75);
(WINTER=TRUE); (SPRING=TRUE); (SCARABS=YES); (RENOVATE=YES); (GRAZ_MANAG=NO)
(TROPIC_GRASS>=3); (SPRING=TRUE); (SUMMER=TRUE)
are used to provide a conclusion but there is no interaction between attributes and the conclusions are found by going directly from the attribute to the conclusion. These types of facts are not terribly interesting. To increase the depth of knowledge in the KBS, it may be worthwhile review the factors that affect these various conclusions to see if some factors have been missed and if relationships between the rule conditions and/or conclusions exist.

Comparing Viewpoints

The Lotus1 viewpoint appears to be the foundation used by the other three viewpoints since it is contained within all other three viewpoints.

The additional concepts found in each of the viewpoints are:

Lotus2
If Soil-type=light then %Light-Maku unsuited to light quick drying soil
If Rainfall=yes then %Maku2-Maku should persist
If Flooding>=3 then %Flood-Flood. Long description =Maku is killed by 3 or more days of flooding if water is still and high temp

Lotus3
If summer-flood >=3 then %Maku3-Maku Ok,but flooding with persistence
If Lotus-Rate>=3 then %Maku4-Maku Ok only at higher sowing rates
If scarabs=yes then %SCARA-Persistence of Maku limited by Scarabs

Lotus4
If moisture<=1250 then %Maku5-Maku at rainfalls above 1250mm
If companion-grass>-1.5 then %SUMM1-Keep grass down in summer. Long description-grass competition is excessive for Lotus - keep grass down in summer

For the purpose of dicussion the following concepts appear related and should be discussed using the cases to assist in reconciliation of differences.

CONCEPTS RELATED TO MOISTURE CONTENT
L2- If Rainfall=yes then %Maku2-Maku should persist
L2-If Flooding>=3 then %Flood-Flood. Long description =Maku is killed by 3 or more days of flooding if water is still and high temp

L3 -If summer-flood >=3 then %Maku3-Maku Ok,but flooding with persistence
L4 -If moisture<=1250 then %Maku5-Maku at rainfalls above 1250mm - This concept appears to be contradictory. The rule condition indicates that the moisture must be below 1250mm but the conclusion comment indicates the situation occurs above 1250mm. If this is an error in the rule condition then these four concepts appear to all consider the affect of rainfall on the persistence of Maku. Resolution of differences in terminology and concepts should be considered.

CONCEPTS RELATED TO COMPETITION FROM WEEDS
L3 -If scarabs=yes then %SCARA-Persistence of Maku limited by Scarabs
L4-If companion-grass>-1.5 then %SUMM1-Keep grass down in summer. Long description-grass competition is excessive for Lotus - keep grass down in summer

CONCEPTS RELATED TO SOIL TYPE
L2- If Soil-type=light then %Light-Maku unsuited to light quick drying soil

Your discussion on this domain reads as though you have some knowledge of it (which I think I am safe in assuming you did not possess before examining the KBs)! You make only one nonsensical statement. It is about a situation in which soil depth is <0. If there is no soil, then soil depth=0 for soil depth to be less than 0 it would have to antimatter or something....A less important error, but one that would make you look silly with clients, was your assumption that scarabs are weeds (they are insects). The common thread here is that weeds and scarabs have an impact on the population/survival of lotus. High densities of ryegrass and ungrazed (poorly managed) tropical grasses have a similar effect.

It was pleasing that it appeared I had some knowledge of this domain. I could not find the nonsensicaI statement mentioned when I checked back to the email I had sent. The above transcript is identical to what was sent. I can only assume that some of the line breaks had messed up the flow of text and that where I refer in point 3 to "soil depth below 30mm" has been segmented so that it reads as 0mm. My assumption that scarabs were weeds instead of insects is amusing and demonstrates my lack of expertise in this domain besides the knowledge I was provided through the KBS. As noted by the expert, at a higher conceptual level the two can be viewed similarly as weeds and pests both pose threats to the survival of Lotus.

Your observations in the last para before "Comparing Viewpoints" are valid. This is to be expected as the rules are being added as expertise is gained. Deep knowledge should follow.

The author believes that the MCRDR/FCA tool can assist in developing and uncovering this deeper knowledge but this belief is not tested here.

On the moisture issue the situation is that lotus requires reasonably wet conditions. This can be mitigated towards drier conditions if the soil has a reasonable storage capacity (depth and light soils are used as proxies for this concept in some rules). Unlike many other species (but in common with strawberry clover lotus can tolerate waterlogged conditions. However conditions must not get so wet that there is a high risk that lotus ends up flooded (still water) for more than 3 days in hot conditions. I think you got close to covering this one!

I also believe my analysis supports these statements by the expert. In fact in the first two points I have already identified the tolerance of Maku (and strawberry clover) for waterlogged soil. In point four I note that while Maku tolerates waterlogged conditions rainfall for longer than 3 days will not be tolerated particularly on hot summer days.

The results in the Lotus domain were more encouraging that those obtained in the blood gases domain. It had only been my goal to determine if I was able to gain an accurate understanding of the domain knowledge and it appears that this was achieved.

8.2.2.3 The SISYPHUS III Geology Domain

The geological experts associated with Sisyphus III KA material were not available for consultation. It was therefore necessary to find another geological expert who could comment on what I had learnt about igneous rocks from the MCRDR KBS that was built. An inquiry was made with the Head of the School of Applied Geology to find if an expert could be found. At the meeting with the geological expert a verbal introduction to the purpose of the meeting was given and the following text was shown to the expert. As in the previous evaluations the expert's comments are shown in Arial 12 and my remarks on his comments are shown italicised in *Times New Roman 12 point*. The numbering system relates to the numbers on the diagrams I had worked from and were not shown to the expert. I used them on my copy of the observations in case I needed to refer to a diagram. They are shown here for reference purposes.

1. There appears to be a typical set of values for the attributes calcium, potassium, iron and iron. When calcium=high, potassium=low,iron=rich. When calcium=medium, potassium=medium, iron=low. When calcium=low, potassium=high and iron= inter-mediate.

3. If a rock is granodiorite, granite or microgranite the COLOUR of the rock will be L_GREEN_WHITE and the value of OLIVINE will be NEVER. These two conditions provide a good starting point and assist the expert in pruning the search space for rock that can be classified as a type of granite.The three types of granite are further distinguished from one another by other attributes. Granodiorite shares the attribute GRAIN_SIZE=COARSE with granite but that they differ in their QUARTZ content, with granodiorite having QUARTZ=SATURATED and granite having QUARTZ=OVERSATURATED. Granite is similar to microgranite in the QUARTZ content but differs in GRAIN_SIZE which is MEDIUM. Granite also takes into account the FELDSPAR content and tbe rule for microgranite includes PYROXENE=POOR. Is there a relationship between PYROXENE and FELSPAR ? If there is these two rules may be considering the same feature of a rock.

8. The density of the rock is determined by the content of dark minerals.

4. Density of the rock is dense if percentage of dark minerals is from 56% to 90%. We reach a conclusion that the density of the rock is very dense if percentage of dark minerals is greater than 90%. A fuller description of the conclusion code reveals that dark minerals includes fe-ag,hornblende,labradorite and opaques.

5. We reach the conclusion that the density of the rock is very light if percentage of dark minerals is less than 30% . We reach a conclusion that the density of the rock is light or medium if percentage of dark minerals from 30% to 56%.

13. We can see the four conclusions %DE000, %DE001, %LI000 and %LI001 are mutually exclusive. As the four attribute-values cover the full range of possible values every rock that has had the DARK_MINERAL content identified can be assigned one of these four conclusions.

6. Grain size is the key factor when considering rock formation. If grain size is coarse then the form of formation of the rock is major intrusions or small slivers. If grain size of the rock is medium then its form of formation is small intrusives. If grain size of the rock is fine then its form of formation is extrusive flows (larva).

7. All rules that conclude Basic Rock-%BS000 have QUARTZ=UNDERSATURATED. The SILICA content is then the next most important feature as it is used in all of the rules with values ranging from very little to intermediate.

9. All rules that conclude Acidic Rock-%AC000 have SILICA=VERY_HIGH. The QUARTZ content is then the next most important feature as it is used in all of the rules with values oversaturated or saturated.
10. From looking at the %AC000 and %BS000 it appears that these two conclusions are mutually exclusive and based on alternate values for the same attributes. Acidic rock have a high silica content. Quartz may be oversaturated or saturated. If saturated the colour tends to be light-green-white. Basic rocks have medium to very little silica content and quartz is undersaturated. The colour may tend towards grey.

11. The closeness of Basalt to Dacite was considered. These two rocks were picked at random as an example of how two or more rocks can be compared. The student would probably be more directed and pick two rocks of particular interest that may be similar but which need to be diffentiated..

    Both rocks have GRAIN_SIZE=FINE. The other distinguishing features are unrelated except for colour. If they had been distinguished using the same attributes then the differences would be alignable and can be seen as more meaningful. The diagram prompts the question - What other rocks have a fine grain_size and may be similar to these two rocks ?

12. As a follow on from 11, the condition (GRAIN_SIZE=FINE) was selected. Firstly it can be seen that a fine grain-size is sufficient to support a number of conclusions - the rock is volcanic (%VC000), extrusive (%EX000) and (%MJ000). Five specific rocks use this conclusion and can be identified based on a number of other features. The rocks are Basalt, Dacite, Rhyolite, Andesite and Trachyte. The attributes COLOUR, OLIVINE and QUARTZ are sufficient to identify two of these rocks (Andesite and Dacite). The additional attribute SILICA classifies Basalt and Trachyte and FELDSPAR is used to classify Rhyolite.

Regarding point 8 the expert commented that it is also necessary to consider the content of light minerals.

Information about light mineral was not contained in the KBS. The identification of missing knowledge is an important outcome of using the line diagrams for learning.

The expert found the comments reasonably correct but that the knowledge only covered igneous rocks. He said that it was well accepted within the geology community that all igneous rocks can be classified based on grainsize and silica (quartz) content. It appeared to him that the KBS had simply developed rules from a firm set of rules accepted worldwide. He felt that classification of igneous rocks was a stable field of knowledge and straightforward and didn't involve too much variation.

These comments seemed somewhat inconsistent with the impression of the domain I had gathered from the KA material due to the inconsistencies between sources and from the findings of others such as Jansen, Schreiber and Wielinga (1998). However, this expert *also stated that if a particular rock was shown to four different experts there would be at least five classifications offered. It appears that much of the problem is in the perception of different attributes and their values by different experts. Since rocks are potentially comprised of many types of minerals a classification of a particular rock is difficult even though there may be consensus at a higher level of abstraction on the important features in classifying different types of igneous rocks.*

What I had learnt he considered to be what a geology student would learn in their first year. What I had told him was simply the accepted rules and classification system for igneous rocks. He commented that at this higher level if there was disagreement between experts it was because they had forgotten their first year training or were trying to change the accepted rules to fit their own research goals. Since I had not uncovered anything new (to him that is) about the domain his comment was also “so what”.

The expert appeared disappointed that he was unable to say that I had learnt something exciting. I was pleased that he considered what I had learnt was what a first year geology student would be taught. I did not expect to uncover something unknown about the domain but hoped that as a novice I would be able to gain some understanding of what was already known by experts.

The expert commented that my observations were a “Reliable reflection of accepted rules for igneous rocks”.

I thought this comment was worth quoting since I felt it provided confirmation of my ability to learn about the igneous rock domain from the line diagrams.

He asked if I had any questions and I asked him if there was a relationship between feldspar, plagioclase and orthoclase. He explained that if the mineral was calcium feldspar it belongs towards the orthoclase end and if the mineral was potassium feldspar it belongs towards the plagioclase end. This explained why some rocks were classified using one or the other.

It was interesting to clarify some of the issues that I had pondered when looking at the relationships between various concepts. It seems if the line diagrams could be used for learning in conjunction with occasional discussion with a domain expert the student could greatly increase their knowledge and that the expert would also be driven to update the knowledge as misunderstandings or missing knowledge became apparent.

Although the expert seemed somewhat unimpressed with what I had learnt I was satisfied by his comments. I think the expert was expecting that I may have uncovered something new and exciting and so repeating basic first year knowledge seemed trivial. To me this was not the case as I was hoping that what I had learnt was in keeping with accepted domain knowledge.

8.2.2.4 The Ion Chromatography Domain - a configuration task

As described in Section 5.3.4, the concepts derived by MCRDR/FCA for the ion chromatography configuration KBS revealed a flat structure with little variation. It was difficult to know if this “poor” (as opposed to rich and interesting) structure was due to the way the KE had entered the rules, the nature of the domain or the nature of the configuration task. Expert evaluation of what I had learnt from this KBS was pointless as there was little of any interest. After discussion with the supervisor (Mary Mulholland) of the PhD student who developed the KBS it appears that the structure was probably due to the way the KE developed the rules (tending to develop rules with one condition and many rules to cover a case) together with the nature of the subdomain that was covered. The set of approximately 300 cases from the full set of 4000 cases could be considered a standard subdomain of ion chromatography that showed little variation in solutions with there only being a couple of ways that various substances could be combined and was largely controlled by a few manufacturers with a limited set of possible configurations. Thus the KE’s method and the nature of the cases can be seen as the reasons why the line diagrams developed are not terribly interesting. A new KBS is planned to be built to cover a wider set of cases over the next 12 months and it would be interesting to find the concepts for the new KBS. However, since this time frame is too long for this study, the question of if and how a configuration domain differs structurally to a classification domain must be left to future work.

8.2.3 Discussion of the evaluation by experts

In all of the classification domains that were evaluated by experts the results were satisfactory to excellent. In the Sisyphus III- geological and Lotus- agricultural domains the results were very encouraging and it appears from the feedback that the experts primarily agreed with my observations and felt that I had gained a reasonable understanding of the domains. The results were not so clear with Blood Gases- medical domain but these results were partly affected by the absence of conclusion code descriptions during the meeting and the terminology used was quite foreign. For example, I do not know the meaning of attributes such as BLOOD_PC02 and BLOOD_BIC describe and conclusion descriptions such as “metabollic compensation”. These domain terms can be seen as basic knowledge that needs to be acquired first before a student can reason within the domain. The other two classification domains used terms that were more comprehensible such as “waterlogged=yes” and “grainsize=coarse” and the more familiar terminology may account the greater ability to learn. The results with the Ion Chromatography - configuration domain were inconclusive as it was not possible to learn much about the domain and the reasons for the lack of models or abstractions in the rules may be due to a combination of limitations associated with the cases chosen, the KE's skill or the nature of the domain. Despite the poor results in the latter domain, the overall conclusion is that the MCRDR/FCA line diagrams are useful for learning.

8.3 Chapter Summary

This chapter has sought to evaluate some of the claims made regarding MCRDR/FCA. In particular the value of the FCA line diagrams has been considered from a number of perspectives including generality for rule-based KBS, compatibility with other graphic rule-based representations, acceptability and comprehensibility by users and usefulness for explanation and learning. A number of different methods of evaluation have been offered including comparison, demonstration, a survey and expert interviews/comments. The evaluations have been subjective and have offered little opportunity for the use of statistical means of determining significance. Despite these limitations it is considered that the line diagram as used in MCRDR/FCA does satisfy the criteria of generality, compatibility, acceptability, comprehensibility and usefulness.

In the next and final chapter we return our focus to the work in this thesis as a whole considering the various contributions, limitations and possible future work.

Chapter 9

9 Thesis Conclusion

In this chapter we review the work described in this thesis, its contribution, the limitations and possible future work.

9.1 A Summary of the Work Presented

The work reported in this thesis is motivated by the belief that we need to focus more on users' needs and cater for the various decision situations in which users will find themselves. To build individual systems that cater for all the activities that may be needed is not feasible or desirable. The problems associated with capturing knowledge are well known and the ability to capture knowledge once and access and manipulate the knowledge in multiple ways adds value to the original knowledge and offers all the benefits associated with the reuse of resources. Thus the problem becomes one of knowledge reuse. The research question pursued in this thesis was "could knowledge captured for one activity be reused to support a wide range of alternative activities allowing the user to answer different types of questions" ? Further, this goal was to be sought in a situated cognition, dynamic knowledge framework.

The scope of this work has been broad rather than deep as the number of *reuse-activities* considered was extensive and the goal of this work has been to see how many different activities could be supported using the same knowledge rather than focusing on a good or best solution to any one particular activity. The use of RDR has allowed greater attention to be paid to the reflective activities that require some understanding and explanation of the knowledge since in RDR the reflexive activities of KA, inferencing and maintenance were already well established within the one system. The strengths and shortcomings of RDR were presented in Chapter Three together with a comparison of RDR with other similar techniques, notably personal construct psychology and case-based reasoning. Also presented in Chapter Three was some work that was done on rule-based reasoning which resulted in the development of a nearest-neighbour algorithm to determine the closeness of concepts within the knowledge base. The limitations of RDR seen as relevant to *activity-reuse* were repetition, lack of higher-level models and the handling of multiple sources of expertise and have been addressed in Chapters Four, Five and Six.

Chapter Four addressed the problem of repetition of knowledge in RDR KBS. Some experiments were described that sought to compact RDR KBS by removing redundancy and repetition. It was expected that the cleaned-up knowledge would facilitate reuse of the knowledge and particularly improve the quality of the explanations provided. The experiments reported had mixed results and the benefits of removing repetition were not clear. A major limitation of the experiments was that they only considered single-classification RDR and it is conjectured that the ambiguity of the false branches was a major cause for the poor results. It also appears that the Garvan data used in the study was not statistically distributed which is inappropriate for a statistical method such as Induct. A proposal for the removal of repetition from MCRDR KBS was offered but further exploration of that proposal was left to future work. The fourth chapter also offered an extension to the RDR approach to verification of MCRDR KBS through a novel usage of rough set theory. The use of RST for verification was also offered as a general approach applicable to any KR that could be converted to decision table format.

Chapter Five described a solution to the lack of higher level modelling available in RDR which was important for many of the reflective activities. This work, together with the development of the nearest neighbour algorithm described in Chapter Three, allowed the relationships between concepts to be uncovered. Through the incorporation of FCA into MCRDR it was possible to develop an abstraction hierarchy from the performance rules. This allowed abstractions and the underlying structure of the rules, which were not previously visible, to be found.

Chapter Six reported on an extension of the approach in Chapter Five to support requirements engineering. This RE work addressed a shortcoming of RDR not considered in any depth in other RDR research which was “how to handle multiple sources of expertise ?”. Also, this work offered an additional *activity-reuse* of the knowledge which was “how does my knowledge compare to knowledge from one or more other sources ?” and “how can differences between the viewpoints be reconciled ?” Chapter Six demonstrated the extendability and possible widespread application of the use of FCA for KBS analysis by building terminological KBS from assertional KBS. The RE approach developed has also proposed some new strategies for the maintenance of MCRDR KBS based on the identified changes in the terminological KBS. As part of these strategies, the paths that ended with stopping rules were not included in the terminological KBS but if the action required a modification rather than a deletion, such as removing one condition, a new rule with the correct conclusion would be added at the top of the MCRDR tree with all conditions bar the dropped one.

The aim of Chapter Seven was to give the reader a feel for the total product of the work in the prior chapters and to demonstrate that the original goal of this thesis, to build a multi-purpose system that uses and reuses knowledge for different activities, had been achieved. The chapter was not designed to be a comprehensive guide to each activity or to be a definitive description of the system but to offer a view of how a number of activities could be included in one system without the need for changes to the knowledge or the inference strategy. The chapter embodies what Stelzner and Williams (1988) had envisaged when they said that KBS should be structured so that changing the usage of the knowledge only required changes to the user interface. The original MCRDR system could support a number of activities already and FCA provided the link between what was already available and those activities such as critiquing and teaching that needed abstractions and structure to be made explicit.

Chapters Three, Four, Five, Six and Seven included sections which evaluated the work reported in the chapter. Chapter Eight concentrates on an evaluation of the FCA line diagram as its value is pivotal to much of this thesis and from discussion with the developers of FCA in Darmstadt such an evaluation had not previously been performed.

In this final chapter, the contribution of this thesis is considered including not only what has been delivered but what may be possible.

9.2 The Contribution of this Thesis to KBS Research

This thesis has made a number of contributions. Some of them relate directly to the original goal and some have been outgrowths of the work. Two of these outgrowths are the use of rough set theory for verification and the development of a requirements engineering framework, both of which apply beyond RDR KBS. While both of these explorations were detours from the original goal of reusing knowledge for different activities and have therefore not been pursued as far as possible, the techniques described in this thesis are comprehensive and supported by implemented systems and examples. These detours were natural consequences from the investigation of the use of RST for finding relationships between concepts and the use of MCRDR/FCA for conflict resolution and are evidence of the usefulness of the approaches developed beyond their intended use in addressing the concerns of this thesis.

The main contribution of this thesis that is seen as more directly applicable to the research question is the incorporation of FCA into MCRDR to build abstraction hierarchies from the primitive rules. This contribution has been significant because it has demonstrated that it is possible to start with a performance system and derive an explanation system. This finding is in stark contrast to the thrust of most KBS research which has concentrated on complex modelling of the domain and problem solving knowledge before KA could even commence. It has been argued in this thesis that the imprecise nature of models and the situated nature of knowledge has resulted in approaches that have done little to alleviate the KA bottleneck. While the general problem solving method and ontological approaches based on the knowledge level are superior to the earlier ‘transfer-of-knowledge’ approaches, they still seem to indicate an Platonic view that knowledge is an artifact and that if we can only model it correctly the rest will fall into place.

We are beginning to a see a change in this view. At the recent EKAW’97 in Barcelona different research groups were starting to view different problem solving methods as variations of a limited number of search strategies. This change is welcome. The focus on building the right model isolated the expert/user even further from the system and has resulted in systems based on the KE’s model rather than the experts. Less focus on these models and a greater focus on the data and the development of performance systems seems more realistic and practical. The RDR philosophy is itself undergoing change as it presents itself as a family of problem solving methods where the PSM is weak (that is, it is a general-purpose strategy) and relies on enough and properly-structured data to be thrown at it to support tasks of classification and construction. It appears that both camps are moving to more middle ground.

This thesis has argued that while explanation systems are difficult to capture and even harder to validate they are useful and necessary for many of the activities that a user will need to perform. This thesis offers a system that includes both performance and explanation knowledge by starting with an assertional KBS, that can be captured and validated with minimal effort, and later deriving the terminological system. FCA is used to provide automated reflection on the primitive concepts in the performance system by uncovering some of the higher level concepts that experts often use but have trouble articulating. The models built by FCA can be presented to the expert and used as a further source of validation and to assist the expert with KA where reflective action is necessary. This thesis has also argued that by making explicit the relationships between concepts in the KBS and by structuring them into an abstraction hierarchy the user is better able to use the knowledge in the KBS for explanation, critiquing and learning purposes.

Another, but less explored, contribution is the use of the nearest neighbour algorithm offered in the MCRDR/FCA system which provides a way of determining the closeness of concepts. Its reliance on comparison of strings within a rule is a limitation but this limitation appears unavoidable unless some other type of symbol or number is used to encode the relationships. In MCRDR/FCA the nearest neighbour algorithm is used to determine which other rule pathways are closest by providing a proximity score between 0-1. This is seen to be useful for providing KA as a reflective critiquing activity and could also be used in learning more about a domain such as in the example in Chapter Three where the higher the score the closer the animal listed was to the man concept selected. Some potential uses of this algorithm, which depending on the situation may require modification, are suggested in the following section. While the nearest neighbour algorithm is part of the MCRDR/FCA toolkit there is no claim that this algorithm is original or optimal. This would require more thorough investigation of other similar algorithms and more support from theoretical semantic analysis. The algorithm was developed from a brief review of cluster analysis research and appeared to be suitable for the task for which it was designed. For example, a comparison between the algorithm and other work by Mehrotra (1995a and 1995b), who has developed a multi-viewpoint clustering analysis methodology, revealed that the algorithm developed in Section 3.3.2.1 provided most of the features offered in the clustering analysis methodology.

Of equal importance is the contribution of this thesis in bringing together a number of theories and techniques. In many cases comparisons and evaluations have been offered in this thesis. The theories and techniques include: Induct, Rough Set Theory, Personal Construct Psychology, Case-Based Reasoning, Semantic Networks and Formal Concept Analysis. The closeness of many of the techniques was interesting to the author particularly their use of differences and/or similarities to find concepts. I was intrigued whether consideration of differences or similarities was the stronger approach or what were the various merits of each approach. This question was clearly and simply answered following a keynote speech by Dedre Gentner at the 4th Australasian Cognitive Science Conference who referred me to a number of papers and pointed out that differences only become important once some similarity has been established. When I considered RDR, PCP, RST or FCA I could see that all approaches either explicitly or implicitly accepted this theory. Only when a case is similar is it and a difference list shown to the user in RDR. In PCP, triadic elicitation requires the user to state what two elements are alike and what third element is different. In RST sets of objects which share the same features are found first and then used to determine differences. An FCA formal context is essentially a grouping of related objects which are being differentiated by the attributes that describe them.

It is noteworthy that no new theory has been developed in this thesis. It appeared to me that there are many ways to solve a particular problem and also that many theories are very similar. Pawlak (personal communication) has commented on the similarity between the use of set theory in RST and FCA with the main difference being what they were trying to achieve. In this thesis I have tended to consider existing theory and to determine how it can be used or adapted to solve the problem at hand. I think that as a community we should work together borrowing from one another rather than reinventing the wheel if one was already available. As a community we suffer when we are inundated with a myriad of alternative approaches with no clear way of determining which one is the best. We also have a chance at achieving some standards if we stabilise the approaches we are using.

Many of the lessons learnt in the MYCIN experiments are confirmed in the work performed as part of this thesis. These lessons included the need for abstracted and more knowledge relevant to the particular activity to be performed. What differentiates the work in this thesis from Clancey's work in GUIDON is that the relationships between concepts and the higher level abstractions, which is some of the extra knowledge that is needed, can be derived automatically using FCA. In Clancey's approach additional knowledge was coded into rules which had different roles, such as the T-rules used to guide the tutoring process. This difference constitutes a substantial benefit because of the reduction in effort required and, as already noted, the capture of performance knowledge seems easier and more reliable than the capture of explanation or terminological knowledge. There is also high fidelity between the performance and explanation systems and ontological commitment between the KBS and FCA built ontology since one is based directly on the other.

This section would not be complete without some analysis of whether a multi-purpose system that allowed the user to reuse knowledge in a wide range of activities had been achieved. The MCRDR/FCA system supports the activities of KA, maintenance, inferencing, critiquing, explanation, tutoring, 'what-if' analysis and hypothesis testing. While it is not claimed that these activities are fully functional or ideally presented, the key concern was whether the knowledge content and structure were adequately robust to allow reuse for different activities. Initial examination of RDR showed that while many activities could be supported to some extent, it was necessary to find a method for finding concepts, both primitive and abstracted, and the relationships between them to provide a full range of activities. The use of FCA, and to a lesser extent the nearest neighbour algorithm, have addressed this need.

The activities offered in MCRDR/FCA are designed to exhibit transparency of interaction (Winograd and Flores 1986) or *readiness-to-hand* (Heidegger 1962), meaning that the system should appear as an extension to the human assisting them in the task they are performing. The user may switch between activities without a conscious change, being driven by their circumstances to ask a different type of question of the system. The activities supported by MCRDR/FCA covers reflexive and reflective activities because humans will at different times behave in these different ways.

9.3 Limitations and Future Work

While this thesis provides a proof-of-concept that one can use and reuse knowledge for multiple purposes and that we can start with a performance system and derive an explanation system there is still much more work to be done particularly in the area of HCI. The user evaluation of the MCRDR/FCA line diagram performed in Chapter Eight was favourable but inconclusive. To a large extent the decision of what should be included in a formal context, the use of the concepts derived, the use of the score provided by the nearest neighbour algorithm and the required features of the line diagrams need to be user and application driven. Without greater user involvement the system developed can not be more than a prototype that can act as a tool for communication of ideas. Nevertheless, some improvements are suggested. Some of the features that would greatly increase the utility of MCRDR/FCA would be the ability to scroll off the screen to other parts of the diagram, to zoom in and out at specified nodes or the ability to drop or add nodes as desired. This would particularly enhance the activities that require explanations such as 'what-if' or scenario analysis. A simple improvement would be to display the short description of the conclusion instead of the conclusion code as the object name. The code makes comprehension of the meaning of implications and 'isa' relationships difficult. As suggested in Section 8.1.1.2, an interesting extension to the MCRDR/FCA tool would be to add a run-time feature that selected which parts of the KBS to add to the formal context based on the current case or a user-response. It is also desirable to allow a split screen so that more than one diagram could be viewed at a time for comparison, although this would require larger screens in many cases. A comparison of the layout algorithm to other algorithmic and/or deductive approaches (Eades 1996) should be made with the possible facility for the user to select from a range of layout methods. Currently the layout algorithm works fairly well as a first approximation but depending on the size and shape of the labels, nodes may need to be moved to avoid overlapping.

More theory-related considerations of the limitations of the approach offered is the simple way that MCRDR KBS, or other rule-bases for that matter, are converted into a formal context. As described in Chapter Five, the formulation of concept lattices from many-valued contexts requires their interpretation into a binary formal context. The interpretation of a rule condition as an attribute in the crosstable has proven straightforward and sufficed for the purposes of the prototype developed. However, a rule that should be part of the context for a selected focus of attention may be missed if the condition does not match on a conclusion or attribute already selected. The use of different conceptual scales may provide a solution and needs further investigation. Perhaps the distance-weighted nearest neighbour algorithm described in Chapter Three could be used to assign a score of relative proximity to indicate which clauses are related and to what extent. This score could then be used to determine which rules should be added to a context. Another option is to generate concepts for the whole KBS and use the FCA concept hierarchy to determine the distance between nodes. Those within a specified distance would be added to a second formal context from which a smaller set of concepts could be derived.

The removal of repetition from MCRDR is still an open question and one that should be pursued. In particular the strategy for the handling of stopping rules needs to be investigated further. From the work that I have done using FCA to draw line diagrams of MCRDR KBS, the stopping rules were often hard to interpret or explain. In some cases it appeared that a stopping rule meant if a case has certain additional conditions then the previous conclusion should no longer be given otherwise the old conclusion was still valid. In other cases the stopping rule may mean that the previous rule, and the entire pathway associated with it, should be ignored. This is a similar problem to the ambiguity of the false branches. With the false branches you do not know if the case fails because it was a ‘don’t care’ situation or because the condition should be false. For the purposes of explanation it is important to know the purpose of the stopping rule and what the appropriate handling should be. If the previous rule is to be stopped then the pathway should be ignored. If the previous rule still holds but not under certain exceptions then it should be retained together with its exceptions. Recent discussions with the developer of MCRDR indicate similar reservations. It would have been good to address in this thesis the shortcomings of the stopping rule strategy and the MCRDR repetition problem. However, the MCRDR representation needs time to mature and more MCRDR KBS to be built before these issues can be adequately addressed.

In Chapter Two, ontologies and some of their limitations were presented. The main problems were that they are time consuming to capture, they are difficult to validate and they can be difficult to change. The abstraction hierarchy built by MCRDR/FCA can be seen as an ontology as was demonstrated in the examples in Chapter Eight which used RDR to build taxonomies. However, in the case of MCRDR/FCA none of the limitations mentioned exist because the ontology is built retrospectively based on the performance knowledge that is easier to capture and validate. This feature, however, means that the descriptiveness of the ontology derived is limited by the assertions and can be considered as a first step. A descriptive model (T-box) may require deeper knowledge to be captured and represented. As was mentioned in section 7.3.2, causal knowledge is an example of deeper knowledge that needs to be acquired but this is still consistent with the approach as the abstraction hierarchy acts as a causal model based on the causal assertions.

There may be merit in the addition of higher level concepts as new rules but since this information can be derived it would add to the incidence of repetition and redundancy in the KBS. Of greater concern is how the higher level knowledge would be maintained since changes to the primitive rules can result in a change in the higher level concept. Maintenance of abstractions is not a concern when only the primitive rules are kept in the assertional knowledge base and the terminological knowledge base is regenerated each time it is needed or when changes are made to assertions.

To support the reuse of abstractions, it is possible with MCRDR/FCA to attach a name to a higher level concept which is stored in a file and whenever a concept is found which shares the same set of attributes, the concept name is shown on the line diagram and can also be accessed via the concept matrix screen. In this way higher level concepts can be identified and reused. This is a similar goal to that sought by NRDR which was described in Section 3.1.3. However, as noted in the previous paragraph, maintenance of abstractions can pose a problem as is the case for NRDR. Taking usage of abstractions a step further, Compton (personal communication) has suggested that these intermediate and high level concepts could be used to support reasoning at different levels. As commonly advocated in the literature, this would be particularly useful for explanation purposes but could also be useful in providing a recommendation depending on the type of information the user was after. The main problem then becomes how to decide what level should be offered. To avoid the problems associated with user-modelling, the simplest, and quite possibly the best, alternative is to let the user specify the level of reasoning and to allow the user to change the level.

Menzies (personal communication) has proposed two further extensions of the use of MCRDR/FCA. The first one he calls Ripple-Down Ontologies. The idea is that we take the higher level concepts found in a T-box derived from an MCRDR/FCA A-box and see if we can reuse these in a new application that appears to have similar concepts. For example if the lower concept “cat” in the first KBS has the features: suckles young and warm-blooded which results in the higher concept “mammal” we may find a lower concept “dog” in a second KBS that has these two attributes which prompts us to assume that we are dealing with a similarly structured domain. We could then copy over the structure developed in the first KBS and modify the copy incrementally to suit the second application. This would result in a bottom-up initial approach followed by a subsequent top-down approach to system development. This top-down approach, like that used by Martinez-Bejar, Benjamins and Martin-Rubio (1997) can be used to assist the user with the population and validation of the KBS and could also be a way of combining different sources of expertise. The first step of identifying similar concepts can be easily performed through the use of the concept name file which was mentioned above. Bringing over all of the higher level concepts (not just the ones that match) has not been tried and would require the objects (the rules) from the first KBS to be added as there must be at least one example of every higher level concept. This will introduce lower level concepts that may not be of relevance to the second application.

Another interesting suggestion by Menzies is the use of MCRDR/FCA for evaluating ontologies. Menzies points out that there is an assumption within ontologies that reusable terms exist. To test this we can give the same cases (which ensures consistent terminology), in the same order, to different experts and compare the RDR rules and FCA T-boxes that they build. If the T-boxes developed are different and cannot be resolved using requirements engineering we can reject the use of ontologies. This would be an interesting exercise which has not been performed due to the unavailability of multiple experts in the same domain, to conduct such an experiment.

A number of uses of the MCRDR/FCA tool have been suggested by Compton (personal communication). One is the use of the abstraction hierarchy to determine which conclusions are similar or the same which could then be combined to reduce the complexity of the KBS. Alternatively the nearest neighbour algorithm could be used on the rule conditions and/or the conclusion descriptions to find how close the conclusion strings are to one another. The use of MCRDR/FCA to reduce the KBS complexity seems feasible but has not been performed due the size limitations of Visual Basic in the handling of files, grids and strings. There were two KBS available, one with 600 the other with 1300 rules that could have possibly been reorganised for this exercise but the limitation of Visual Basic meant the current tool was unsuitable. A reimplementation in another language was not desirable at this stage.

Another problem being faced by a pathology MCRDR KBS currently under construction (3,000 rules to date) is the situation where the same conclusion is being reached but through different pathways. This is satisfactory for inferencing as duplicate conclusions are edited out but for the purposes of explanation it is unclear which path should be reported and how to explain the different reasons. Multiple pathways also make KA difficult as there are numerous situations to consider. The MCRDR/FCA hierarchy could provide a way of ordering the rule pathways to reveal what were the main concepts and what variations existed.

It is apparent that the MCRDR/FCA tool has opened up the way to addressing a number of problems concerning the use and understanding of knowledge in a KBS. These issues have not been addressed in this thesis and those mentioned are only a few examples of the potential value of such a tool. The realisation of this potential is left to future work.

9.4 Final Words

As described, many KBS approaches assume the domain must be fully described before we can begin to acquire knowledge and seem to take the view that "there is no perception without prior learning" (Rosenfield 1988, p.7). However, the focus of KBS should be on systems that support change and evolution of the knowledge since new perceptualisations of categories are continually reconstructed. This also means that it is not necessary or feasible to try to encode all the knowledge before it can be put into use as most approaches to KBS development require (Kang, Gambetta and Compton 1996). Systems can be put into routine use with a minimal set of rules and developed online, as was the case with PEIRS. It also means that rather than using KBS methods that rely on identification of the task structure or problem solving method/s before the appropriate method can be applied it is desirable to have a KA and representation technique that allows a system to be built incrementally and "gradually evolve into whatever type of ... system is best suited to the problem as new cases are seen" Kang, Gambetta and Compton 1996, p.267, which was first suggested in Compton et al (1993).

Just as RDR stresses the incremental nature of KBS development, MCRDR/FCA systems are not expected to be complete but able to evolve. This evolution allows new activities to be added according to user requirements. The key concern is whether the MCRDR/FCA framework is sufficiently robust so that new activities only require user interface enhancements or more knowledge. As was found by Lee and Compton (1995), causal modelling required additional knowledge but it was not necessary to change the structure of the knowledge or the way that inferencing was performed. Causal modelling was possible due to the causal positive or negative relationships between nodes. These relationships could be derived automatically or captured manually. The recent work (Ramadan et al 1997a) in configuring for ion chromatography supports the reuse of the MCRDR algorithm for classification and configuration tasks. MCRDR can handle well the reflexive activities of KA, maintenance and inferencing. By adding FCA tools to MCRDR, MCRDR/FCA is also able to support reflective activities such as explanation, critiquing and 'what-if' analysis. Thus, MCRDR/FCA supports the reuse of knowledge for multiple activities. In the case of PSM and activity reuse, data is the key. It is anticipated that if sufficient data of the right type is captured, RDR, including MCRDR/FCA will be able to handle whatever task or activity is required. This is a big claim and one that is not substantiated in this thesis. But this is the direction in which RDR research is heading and MCRDR/FCA also pushes RDR in this direction.

The system described is an attempt to build a system that does not force the system developer's prejudices onto the user by restricting interaction to a particular type of activity. Accepting the limitations and developer-imposed restrictions in the user interface, the idea is that the user can ask their own questions rather than having to reframe their question into the type of activity that is offered by the system. By offering a wide range of ways of utilising the knowledge the user has the flexibility to interact in the way that best suits their current situation.

The work and tool described in this thesis were aimed at building a system that would allow the user to view, explore, analyse, maintain, manipulate and consult the knowledge in a knowledge based system. While the tool itself has many limitations, the key aspect to be emphasized is the freedom on the part of the user to decide how to use this tool and is well expressed in the following idea of Winograd and Flores:

> *"the use of the tool [referring to the computer] shapes the potential for what those actions are and how they are understood....its power does not lie in a single purpose."*
> *Winograd and Flores 1986, p.170."*

It is also believed that the system described embodies the vision that:

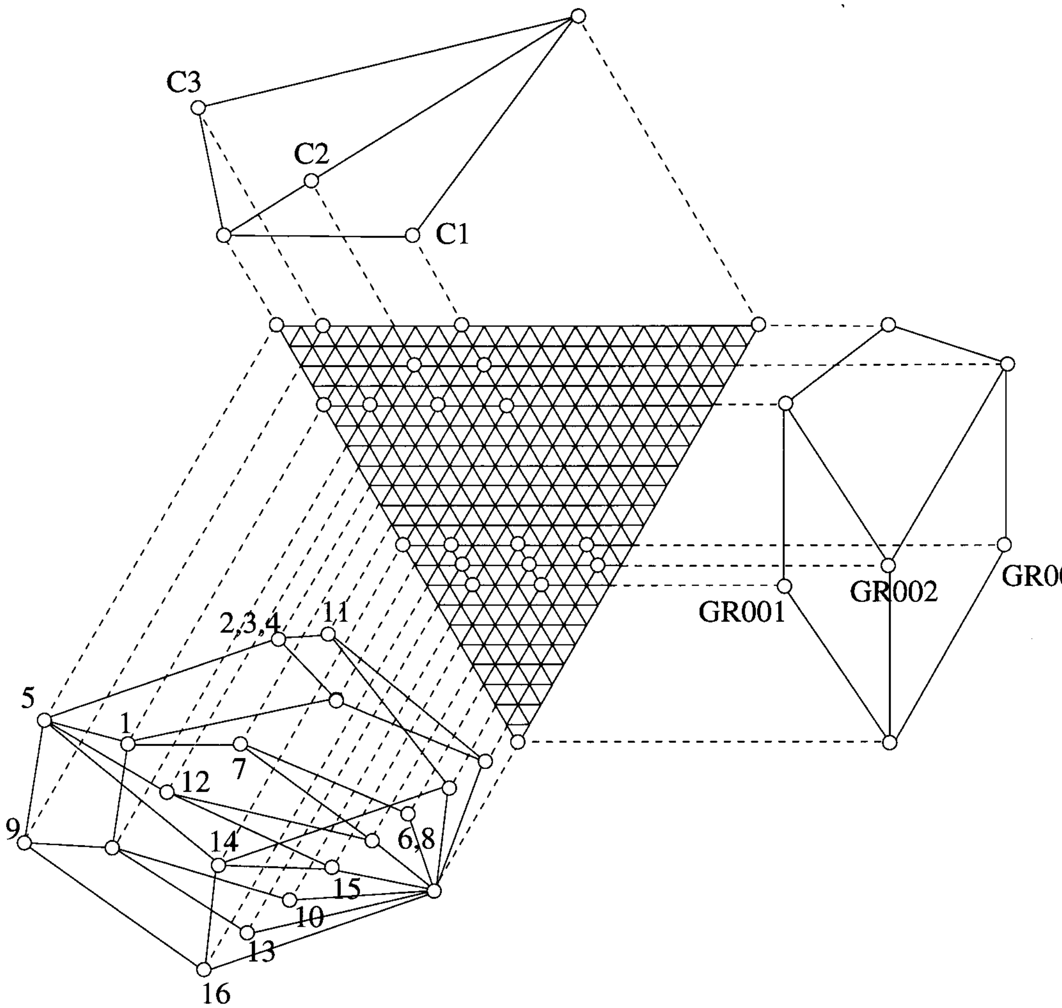
> *"The most successful designs are not those that try to fully model the domain in which they operate, but those that are 'in alignment' with the fundamental structure of that domain, and that allow for modification and evolution to generate a new structural coupling." Winograd and Flores 1986, p.53.*

RDR allows the user to capture a simple observable model of their world using attribute-value pairs and conclusions. By allowing the expert to retrospectively view and explore their underlying conceptual models using FCA, together with a situated system style as described above, it is expected that the system may be a vehicle for transformation of tradition (Winograd and Flores (1986, p.170). We all act according to a tradition based on a communal view. By helping the expert understand their domain we help them to understand, question and possibly change their tradition. Much of the failure of ES is due to the improper view of what they are and should be able to achieve. An ES must be seen as a tool and the emphasis must be on support not on replacement. In this thesis the emphasis has been on giving the user a flexible tool that the user controls by choosing how to use and reuse the knowledge according to their situation and personal preferences.

Appendix A

Triadic Diagram using the SISYPHUS III data by
Klaus Biederman

Appendix A - Triadic Diagram using the SISYPHUS III data by Klaus Biederman



1: Grain Size: Coarse
2: Colour: Leucocratic
3: Olivine: Never
4: Quartz: Always
5: 1=1
6: Quartz: Saturated
7: Olivine: Absent
8: Feldspar: Never
9: Dark Minerals: LT30
10: Feldspar: GT2 3Plagioclase
11: Silicia: GT68
12: Quartz: Oversaturated
13: Feldspar: GT2 3Orthoclase
14: Grain Size: Medium
15: Colour: Light Green
16: Quartz: Present

The above diagram has been developed by Klaus Biederman using the technique described in Biederman (1997). The attributes are listed below the diagram as numbers 1-16. The three viewpoints C1, C2 and C3 are from Card Sorts 1, 2 and 3 from the Sisyphus III KA material. The objects are the rocks Granodiorite - GR000, Granite - GR001 and Microgranite - GR002. The appropriate rows from the KA material were extracted by me and sent to Klaus to see what a triadic representation of this data would look like. The diagram is included here as it offers another visualisation of the knowledge that may be useful for conflict detection and resolution. Currently the automated tool which produced this diagram is limited to handling a combined total of sixteen (16) attributes to cover all objects and viewpoints

Appendix B

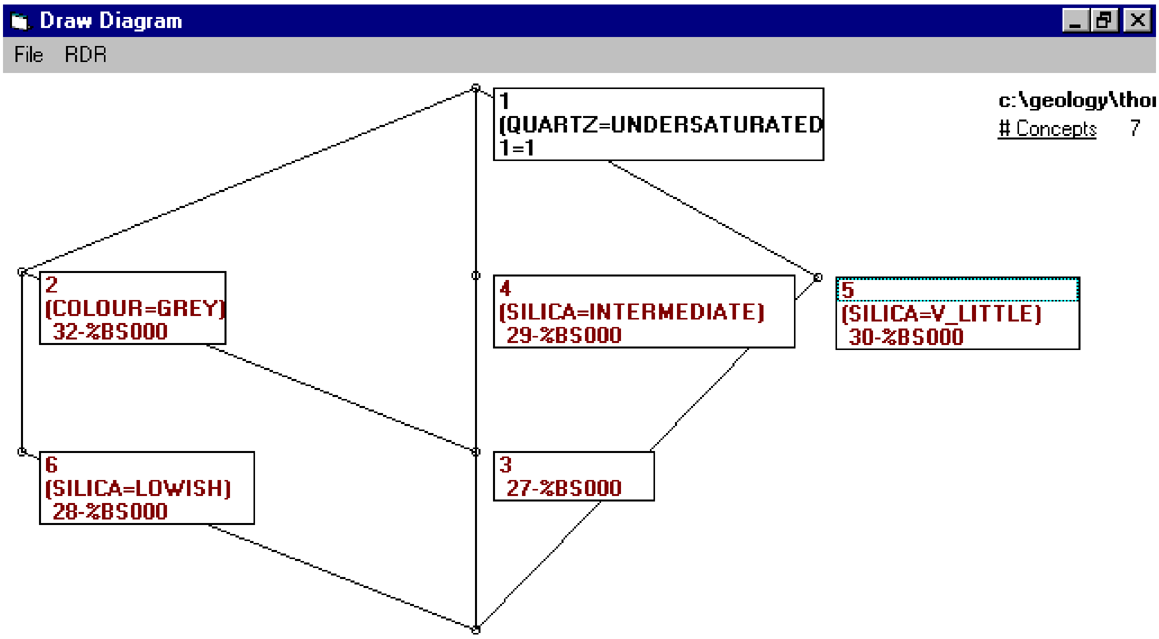
Survey used in Chapter Eight to
Evaluate Line Diagrams with Novices

Questionnaire

Section One: Line Diagrams

In this section I am interested to find out how quickly you can learn to read a line diagram. The diagrams use a geological KBS built from the Sisyphus III data. I have included two line diagrams. Remember that attributes (the intension) are reached by ascending pathways and objects (the extension) are reached by descending paths. The objects are labelled with the rule number and the rule's conclusion. I have completed the first question as an example in the first diagram.

Do you feel you would know something about this domain ? If yes, explain.



What is the full description of concept number 4 -
Set of attributes - Quartz=undersaturated, 1=1, Silica=intermediate

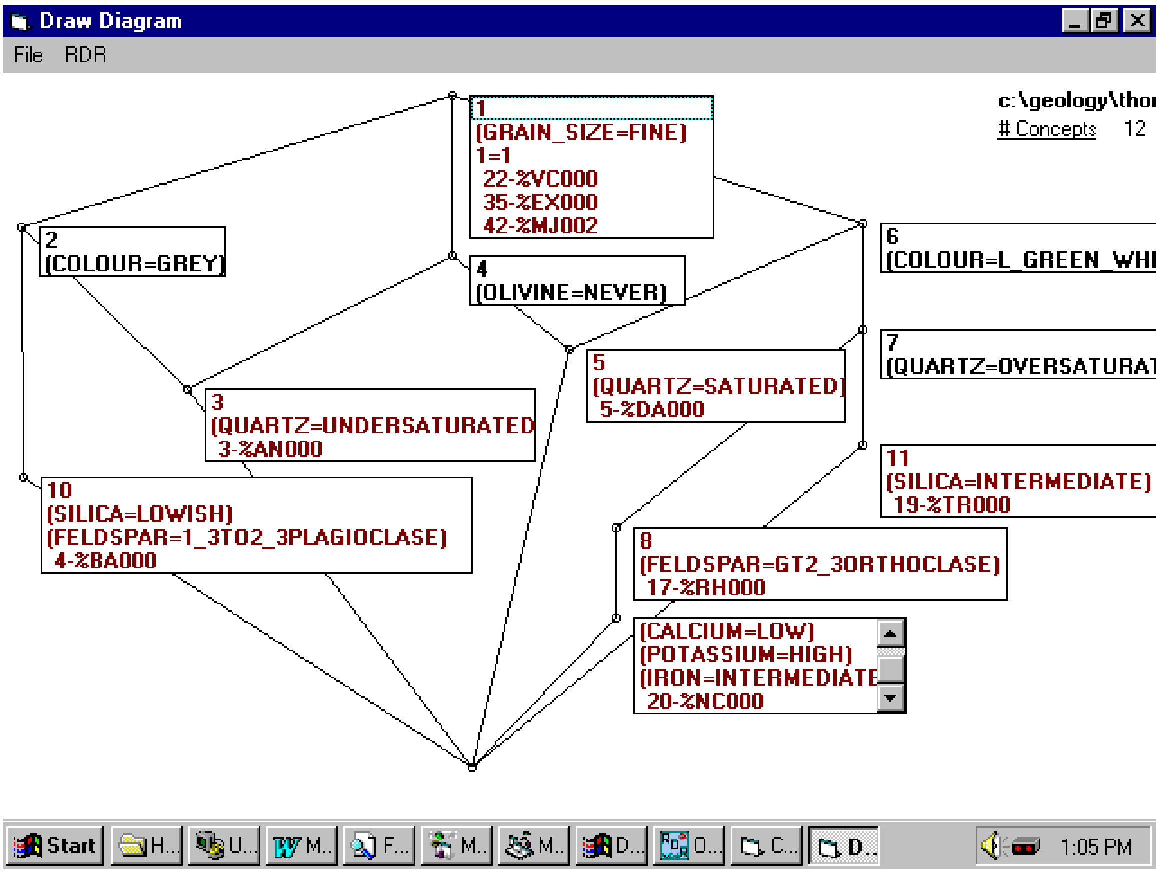
Set of objects:- 29-%BS000, 27-%BS000

What is the full description of concept number 2 -
Set of attributes -

Set of objects:-

What is the full description of concept number 3 -
Set of attributes -

Set of objects:-



What is the full description of concept number 5 -
Set of attributes -

Set of objects:-

What is the full description of concept number 7 -
Set of attributes -

Set of objects:-

Section Two: Comparing using the line diagrams with rule traces

Half of the group are shown a line diagram of a set of rules and the other half are shown the rule traces. The speed with which you can answer the following questions using the source you have been given is being evaluated.

1 - The VC/PL Question
%PL000=plutonic, %VC000-volcanic

Question 1.1: **Can a rock be classified as plutonic and volcanic ?**
Start time: End/stop time: Total time:
Answer 1.1
Yes/No/Sometimes/Don't know

Question 1.2: **When should a rock be classified as plutonic and/or volcanic if the grain_size is medium?**
Start time: End/stop time: Total time:
Answer 1.2

2 - The GR Questions

Question 2.1 Is the conclusion %GR000 closer to the conclusion %GR001 or %GR002
Start time: End/stop time: Total time:
Answer 2.1:

Question 2.2 **What conditions are shared by all the rules belonging to the %GR family?**
Start time: End/stop time: Total time:
Answer 2.2:

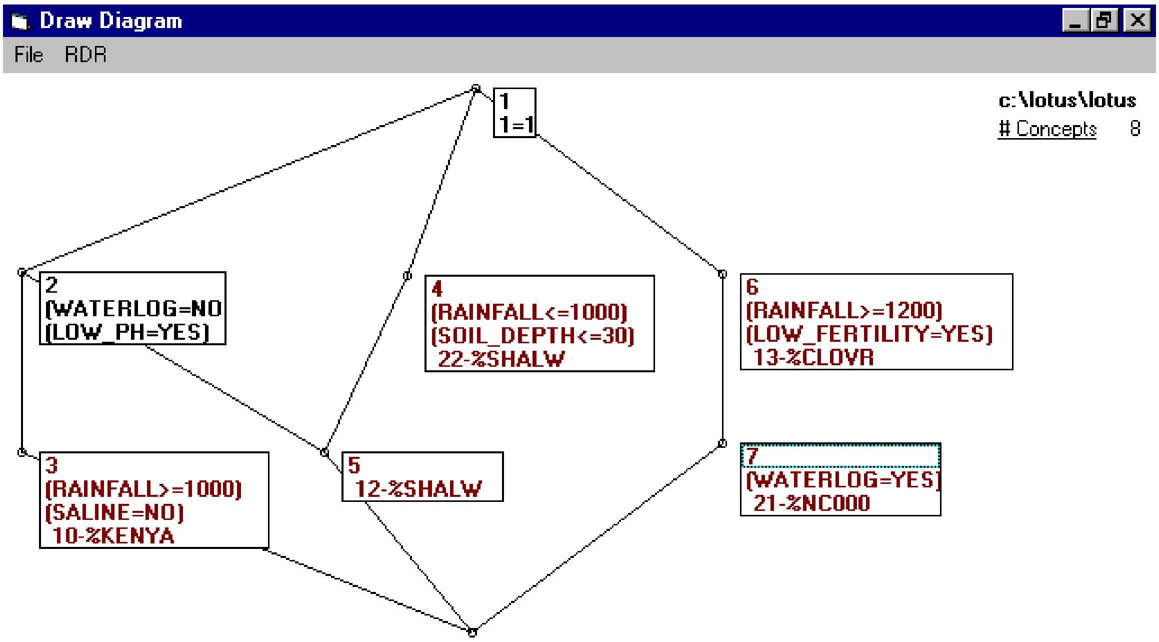
General comments regarding rule traces (optional)

General comments regarding the line diagrams (optional).

Section Four: Line Diagrams for Learning

We now move to an agricultural domain - MAKU LOTUS a new grasslands crop. I have tried to learn something about this domain using the line diagrams. Below is the screen shown when I select all rules that use the attribute RAINFALL. I have written a summary of observations I have made.

Do you feel you would know something about this domain ? If yes,e xplain
The conclusions used:
%CLOVR-Maku OK, also consider white clover plus fertiliser
%FLOOD-Not Maku Lotus
%KENYA- Kenya Clover
%MAKUU-Maku Lotus.
%NC000- No Conclusion.
%SALIN-Maku okay but other legumes such as strawberry clover should be considered.
%SHALW-Not Maku lotus.
%SHARN-Sharnae or Goldie



Concept 2 may represent a higher concept as it is shared by a two primitive concepts and does not have a conclusion associated with it. It could be conjectured that the conditons LOW-PH=YES and WATERLOG=NO are initial indicators of poor conditions for MAKU, since the rules that use this concept do not recommend MAKU. This hypothesis could be tested further by looking at all rules that use these two conditions. Although unsuitable for MAKU, in Concept No. 3 Kenya Grass is recommended in this situation if the soil is not saline and the rainfall is over 1000mm otherwise the conclusion is %SHALW- not Maku Lotus in Concept No. 5. This last conclusion is further reinforced if there is low rainfall and shallow soil (Concept No. 4). These extra conditions may be useful further tests or may be redundant knowledge that should be removed.

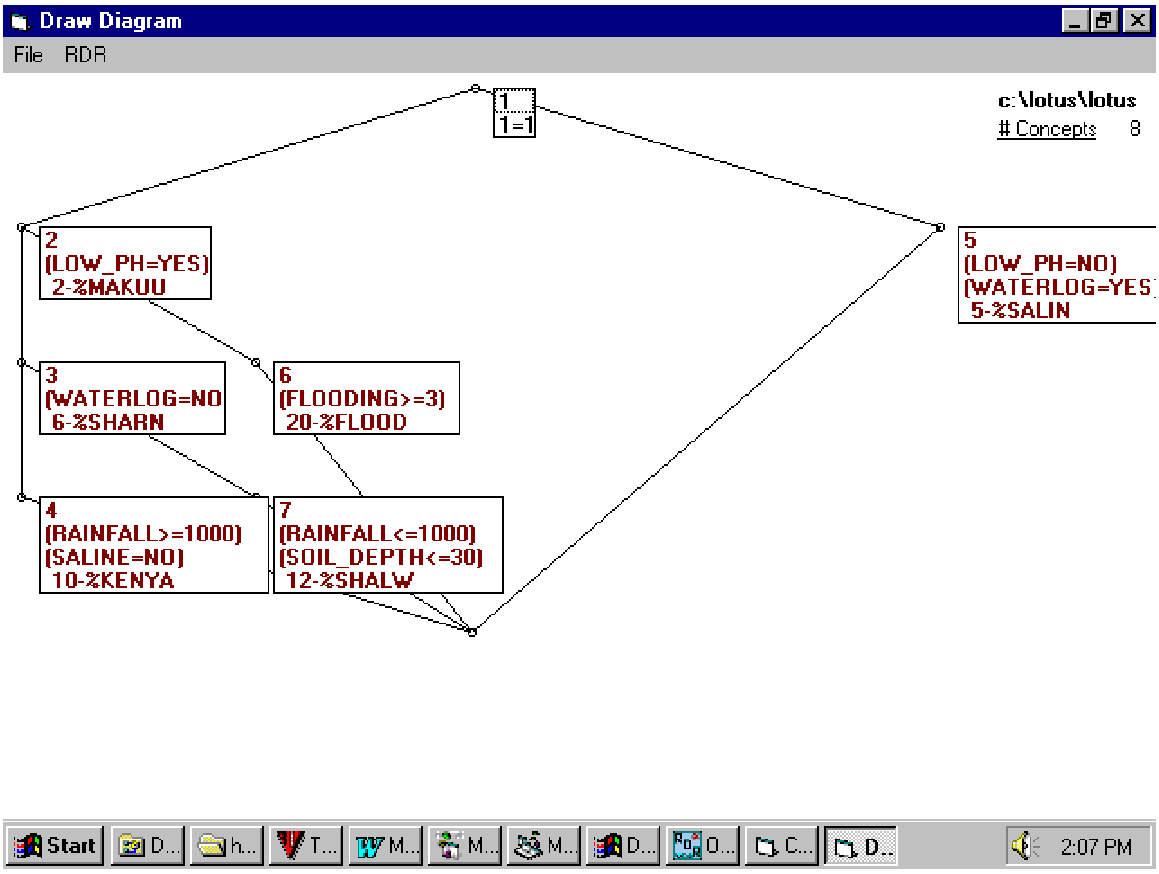
On the right hand side in concept 6 we can see that if the rainfall is over 1200mm and the soil FERTILITY is LOW then “%CLOVR-Maku OK, also consider white clover plus fertiliser” is recommended except if the ground is waterlogged and then clover should not be recommended and

%NC000 -No Conclusion is given, which can be seen in concept 7. One might also assume that if the soil FERTILITY was not LOW then a different conclusion which also recommended to consider white clover without the need for fertiliser should exist. If it existed it would have been picked up in this query. This may have identified some missing or incomplete knowledge ?

Do these observations seems reasonable ?

Any further comments ?

Now it's your turn. Look at the diagram below. It includes the rules that use the partial attribute PH. A text description of the conclusions are given below. Write a few words/lines on what you can learn about it.



Section 4: Comparing different representations of the Lens Data Rules

Diagram 1 - MCRDR KBS
Diagram 2 - Formal Context (crosstable/decision table)
Diagram 3 - Concept Matrix
Diagram 4 - Concept Lattice - line diagram
Diagram 5- Gaines' Visual Language
Diagram 6 - Exception Directed Acyclic Graph (EDAG)

Give each representation a score between 1-5 (1 is the lowest - 5 is the highest) on each of the following points by circling the score:

Pleasing appearance

Diagram 1:	1	2	3	4	5
Diagram 2:	1	2	3	4	5
Diagram 3:	1	2	3	4	5
Diagram 4:	1	2	3	4	5
Diagram 5:	1	2	3	4	5
Diagram 6:	1	2	3	4	5

Comprehensibility

Diagram 1:	1	2	3	4	5
Diagram 2:	1	2	3	4	5
Diagram 3:	1	2	3	4	5
Diagram 4:	1	2	3	4	5
Diagram 5:	1	2	3	4	5
Diagram 6:	1	2	3	4	5

Appendix B - Survey used in Chapter Eight to Evaluate Line Diagrams with Novices

Diagram 1 - MCRDR KBS
Diagram 2 - Formal Context (crosstable/decision table)
Diagram 3 - Concept Matrix
Diagram 4 - Concept Lattice - line diagram
Diagram 5- Gaines' Visual Language
Diagram 6 - Exception Directed Acyclic Graph (EDAG)

Good Structure

Diagram 1:	1	2	3	4	5
Diagram 2:	1	2	3	4	5
Diagram 3:	1	2	3	4	5
Diagram 4:	1	2	3	4	5
Diagram 5:	1	2	3	4	5
Diagram 6:	1	2	3	4	5

Ability to determine implications

Diagram 1:	1	2	3	4	5
Diagram 2:	1	2	3	4	5
Diagram 3:	1	2	3	4	5
Diagram 4:	1	2	3	4	5
Diagram 5:	1	2	3	4	5
Diagram 6:	1	2	3	4	5

Bibliography

Agnew, N.M., Ford, K.M. and Hayes, P.J. (1997) Expertise in Context: Personally Constructed, Socially Situated, and Reality-Relevant ? In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 219-244.

Agre, P.E. and Chapman, D. (1987) Pengi: An Implementation of a Theory of Activity, *Proceedings of the Sixth National Conference on Artificial Intelligence,* Menlo Park, CA, American Association for Artificial Intelligence, 268-272.

Agre, P.E. (1993) The Symbolic Worldview: Reply to Vera and Simon *Cognitive Science* 17, 61-69.

Anderson, J.R. (1978) Arguments concerning representations for mental imagery *Psychological Review, 85*:249-277.

Baader F. (1995) Computing a Minimal Representation of the Subsumption Lattice of all Conjunctions of Concepts Defined in a Terminology. In *Proceedings of the International Symposium on Knowledge Retrieval,Use, and Storage for Efficiency,* KRUSE 95, Santa Cruz, USA, 1995.

Balzer, R. (1991) Tolerating Inconsistency *Proceedings of 13th International Conference on Software Engineering (ICSE-13)* Austin, Texas, USA, 13-17th May 1991, IEEE Computer Society Press, 158-165.

Bannis, Joseph (1990) *An Intelligent Decision Support System for Public Sector Investment Programming.* Phd Thesis, University of Missouri, Rolla, USA.

Bareiss, E.R (1989) *Exemplar-Based Knowledge Acquisition: A Unified Approach to Concept Representation, Classification, and Learning* Academic Press, Boston.

Baroff, J., Simon, R., Gilman F. and Schneiderman, B. (1988) Direct Manipulation User Interfaces for Expert Systems in Hendler, J. A.(ed), *Expert Systems: The User Interface* Ablex Publishing Corp.

Barr, Valerie (1997) Applications of Rule-Base Coverage Measure to Expert System Evaluation In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 411-416.

Beail, N. (1985) *Repertory Grid Technique and Personal Constructs: Applications in Clinical and Educational Settings* Croom Helm, Sydney.

Bennett, John L., (1983) *Building Decision Support Systems.* Addison-Wesley Publishing Company Inc., Reading.

Beydoun, G. and Hoffmann, A. (1997) Acquisition of Search Knowledge, In Enric Plaza and Richard Benjamins (eds), *Knowledge Acquisition, Modeling and Management.* 10th European Workshop, EKAW'97, Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 1-16.

Beydoun, G. and Hoffmann, A. (1998) Building Problem Solvers Based on Search Control Knowledge, In Gaines, B. and Musen, M. (eds) *Eleventh Workshop on Knowledge Acquisition, Modeling and Management (KAW'98),* Banff, Alberta, Canada, 18-23 April, 1998, Vol 1:SHARE.3.

Biederman, K. (1997) How Triadic Diagrams Represent Conceptual Structures In Dickson Lukose, Harry Delugach, Marry Keeler, Leroy Searle, and John F. Sowa, (eds) (1997), *Conceptual Structures: Fulfilling Peirce's Dream, Proceedings of the Fifth International Conference on Conceptual Structures (ICCS'97)*, August 3 - 8, University of Washington, Seattle, USA, Lecture Notes in Artificial Intelligence, Springer-Verlag, Number 1257, Berlin, 304-317.

Boose, J.H., J.M. Bradshaw, C. M. Kitto and D.B. Shema. (1989) From ETS to Aquinas: Six Years of Knowledge Acquisition Tools in *Proceedings of the 4th AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Canada, October, 1989, 5.1-5.17.

Boose, J.H., J.M. Bradshaw, J.L. Koszareck, and D.B. Shema. (1992) Knowledge Acquisition Techniques for Group Decision Support in *Proceedings of the 7th Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Canada, 2.1-2.22.

Booth, P., (1989) *An Introduction to Human Computer Interaction* LEA Publishers: Hillsdale.

Boswell, Robin, Craw, Susan and Rowe, Ray (1997) Knowledge Refinement for a Design System, In Enric Plaza and Richard Benjamins (Eds), *Knowledge Acquisition, Modeling and Management*. 10th European Workshop, EKAW'97, Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 49-64.

Brachman, R.J. (1979) On the Epistemological Status of Semantic Networks In Findler, N.V. (ed) *Associative Networks: Representation and Use of Knowledge by Computers* Academic Press, New York, 3-50.

Brachman, R.J. (1990) The Future of Knowledge Representation *Proceedings Eighth National Conference on Artificial Intelligence* AAAI Press / MIT Press, Menlo Park, 1083-1092.

Bradshaw, J.M., S.P. Covington, P.J. Russo and J.H. Boose, J.H., (1989) KWAQing with DDUCKS: Knowledge Acquisition Techniques for Decision Analysis Using Axotl and Aquinas in *Proceedings of the 4th Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Canada.

Bradzil, P. Gama, J. and Henery, B. (1994) Characterising the Applicability of Classification Algorithms Using Meta-level Learning *Technical Report LIACC University of Porto*, Rua Campo Alegre 823, 4100 Porto, Portugal.

Breuker, J. (1994) Components of Problem Solving and Types of Problems In Steels, L. Schreiber, G. and Van de Velde, W. (eds) *A Future for Knowledge Acquisition Proceedings of the 8th European Knowledge Acquisition Workshop, EKAW'94,* Lecture Notes in Artificial Intelligence 867, Springer Verlag, 118-136.

Brown, R., (1988) *Group Processes: Dynamics within and between Groups* Basil Blackwell Ltd, Oxford.

Buchanan, B.G. and Shortliffe, E.H. (1984) *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project* Addison-Wesley Publishing Co. Reading, Massachusetts.

Burmeister, P. (1996) Formal Concept Analysis with ConImp: Introduction to the Basic Features A translation of ConImp -Ein Programm zur Formalen Begriffanalyse In G.Stumme, R.Wille (eds.) *Begriffliche Wissenerarbeitung:Methoden und Anwendungen*, Springer Verlag.

Buvac, Sasa and Mason, Ian A. (1993) Propositional Logic of Context *Proceedings of the 11th National Conference* or URL http://sail.stanford.edu/buvac/propositional-logic-of-context.ps.

Carbonell, J.G., Knoblock, C.A. and Minton, S. (1989) Prodigy: An Intelligent Architecture for Planning and Learning *Architectures for Intelligence* Lawrence Erlbaum Associates, Hillsdale, NJ.

Catlett, J. (1992) Ripple-Down-Rules as a Mediating Representation in Interactive Induction *Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop*, Kobe, Japan, Nov 9-13 1992, 155-170.

Cawsey, A. (1993) User Modelling in Interactive Explanations *Journal of User Modelling and User Adapted Interaction.*

Chander, P.G., Shinghal, R. and Radhakrishnan, T. (1997) Performance Assessment and Incremental Evaluation of Rule-based Systems In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation of Knowledge-Based Systems Workshop*, July 28, 1997, Providence, Rhode Island, Technical Report WS-97-01, AAAI Press, Menlo Park, California, 40-46.

Chandrasekaran, B. (1986) Generic Tasks in Knowledge-Based Reasoning: High Level Building Blocks for Expert System Design *IEEE Expert* Fall 1986, 23-30.

Chandrasekaran, B. and Johnson, T. (1993) Generic Tasks and Task Structures In David, J.M., Krivine, J.-P. and Simmons, R. (eds), *Second Generation Expert Systems* Springer, Berlin, 232-272.

Chandrasekaran, B., Tanner, M.C and Josephson J.R., (1988) Explanation: The Role of Control Strategies and Deep Models in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corporation.

Chapman, D., (1989) Penguins can make cake *AI Magazine*, 10: 45-50.

Chappel, H.R., Wilson, M.D. and Cahour, B., (1992) Engineering User Models to Enhance Multi-modal Dialogue In J.A. Larson and C. Unger (eds), *Engineering for Human-Computer Interaction* Elsevier Science Publishers, Amsterdam, 297-313.

Chellen, Vijaletchmee (1995) Induct/RDR and Knowledge Base Compaction *BE Computer Engineering Honours Thesis*, University of New South Wales.

Clancey, W.J., (1984) Methodology for Building an Intelligent Tutoring System, Kintsch, W, Miller, J. and Polson, P. (eds) *Methods and Tactics in Cognitive Science* Lawrence Erlbaum Associates, New Jersey, 51-83.

Clancey, W.J., (1985) Heuristic Classification *Artificial Intelligence 27:*289-350.

Clancey, W.J. and Bock, C (1985), *Representing Control Knowledge as Abstract Task and Metarules* Technical Report Stanford Knowledge Systems Laboratory, Department of Computer Science, Palo Alto.

Clancey, W.J., (1986) From GUIDON to NEOMYCIN and HERACLES in Twenty Short Lessons: ORN Final Report 1979-1985 *AI Magazine***,** August 1986, 40-60.

Clancey, W.J., (1988) Acquiring. Representing and Evaluating a Competence Model of Diagnosis In M.T.H. Chi, R. Claser and M.J. Farr, (eds), *The Nature of Expertise* Lawrence Erlbaum Associates, New Jersey, 343-418.

Clancey, W.J., (1991a) The Frame of Reference Problem in the Design of Intelligent Machines In K. VanLehn, ed. *Architectures for Intelligence,* Erlbaum, Hillsdale.

Clancey, W.J., (1991b) Book Review - Israel Rosenfield, The Invention of Memory: A New View of the Brain *Artificial Intelligence*, 50:241-284.

Clancey, W.J., (1992) Model Construction Operators *Artificial Intelligence 53*:1-115.

Clancey, W.J., (1993) Situated Action: A Neurological Interpretation Response to Vera and Simon *Cognitive Science*, 17: 87-116.

Clancey, W. (1997) The Conceptual Nature of Knowledge, Situations and Activity In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 247-291.

Clark, Peter and Porter, Bruce (1997) Building Concept Representations from Reusable Components In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 369-376.

Cleal, D.M. and Heaton, N.O. (1988) *Knowledge-Based Systems: Implications for Human-Computer Interfaces* Ellis Horwood Ltd.

Cohen, R., Jones, M., Sanmugasunderam, A., Spencer, B. and Dent, L. (1989) Providing Responses Specific to a User’s Goals and Background *International Journal of Expert Systems* 2(2):135-162.

Cohen, P. R. and Loiselle, C. L. (1988) Beyond isa: Structures for Plausible Inference in Semantic Networks In H. Levesque and R. Brachman (eds) *Readings in Knowledge Representation* Morgan Kaufman, San Mateo, California, 411-431.

*Application of formal concept analysis to information retrieval using
a hierarchically structured thesauris , R.J. Cole and P.W. Eklund,
International Conference on Conceptual Graphs, ICCS '96, Sydney. **Referred in C5 - look up on web not in ICCS proceedings.**

Collins, H.M. (1997) RAT-Tale: Sociology’s Contribution to Understanding Human and Machine Cognition In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds)

Expertise in Context: Human and Machine AAAI Press/MIT Press, Cambridge, Massachusetts, 293-311.

Colomb, R.M. (1989) Representation of Propositional Expert Systems as Decision Tables *Technical Report TR-FB-89-05* Paper presented at the *Third Joint Australian Artificial Intelligence Conference* (AI'89) Melbourne, Victoria, Australia 15-17 November, 1989.

Colomb, Robert.M. (1992) Computational Stability of Expert Systems *Expert Systems with Applications* 5(2/3): 411-419.

Colomb, Robert.M. (1993) Decision Tables, Decision Trees and Cases: Propositional Knowledge-Based Systems *Technical Report No. 266* Key Centre for Software Technology, Department of Compter Science, The University of Queensland, Australia.

Colomb, R.M. and Sienkiewicz, J. (1995) Analysis of Redundancy in Expert Systems Case Data Xin Yao (ed) *Proceeding of the Eighth Joint Conference on Artificial Intelligence AI'95* 13-17 November 1995, Canberra, World Scientific, Singapore, 395-402.

Compton, P. and Jansen, R., (1988) Knowledge in context: a strategy for expert system maintenance, *AI'88: Proceedings of the second Australian Conference in Artificial Intelligence*, Adelaide, 283-297.

Compton, P. and Jansen, R., (1990) A Philosophical Basis for Knowledge Acquisition. *Knowledge Acquisition* 2:241-257**.**

Compton, P., Horn, R., Quinlan, and Lazarus, L (1989) Maintaining an Expert System In R. Quinlan (ed.), *Applications of Expert Systems***,** Addison Wesley, 366-385.

Compton, P., Edwards, G., Kang, B., Lazarus, L., Malor, R., Menzies, T., Preston, P., Srinivasan, A. and Sammut, C. (1991a) Ripple Down Rules: Possibilities and Limitations *6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems Workshop*, Banff, 6.1 - 6.18.

Compton, P., Srinivasan, A., Edwards, G., Malor, R. and Lazarus, L., (1991b) Knowledge Base Maintenance Without a Knowledge Engineer *Proceedings World Congress on Expert Systems*, Pergmon Press, Orlando, 1:668-675.

Compton, P., Edwards, G., Kang, B., Lazarus, L., Malor, R., Preston, P. and Srinivasan, A. (1992). Ripple down rules: Turning knowledge acquisition into knowledge maintenance *Artificial knowledge in Medicine* 4:463-475.

Compton, P., Kang, B., Preston, P. and Mulholland, M., (1993), Knowledge Acquisition without Analysis, *European Knowledge Acquisition Workshop*, Springer Verlag, 277-290.

Compton, P., Preston, P. and Kang, B., (1994) Local Patching Produces Compact Knowledge Bases *A Future in Knowledge Acquisition* (eds) L. Steels, G. Schreiber and W. Van de Velde, Berlin, Springer Verlag, 104-117.

Compton, P., Preston, P. and Kang, B. (1995) The Use of Simulated Experts in Evaluating Knowledge Acquisition, *Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. Feb 26 - March 3 1995, Vol 1:12.1-12.18.

Compton, P., Ramadan, Z., Preston, P., Le-Gia, T. Chellen, V., Mulholland, M. Hibbert, D.B., Haddad, P.R and Kang, B. (1998) A Trade-off Between Domain Knowledge and Problem-Solving Method Power In Gaines, B. and Musen, M. (eds) *Eleventh Workshop on Knowledge Acquisition Modeling and Management (KAW'98)* Banff, Canada, April 18-23, 1998, SRDG Publications, University of Calgary, Calgary, SHARE-7.

Coombs, M. and Alty, J. (1984) Expert Systems: an Alternative Paradigm *International Journal of Man-Machine Studies 20*, 21-43.

Cragun, B.J. and Steudel, H.J. (1987) A Decision-Table-Based Processor for Checking Completeness and Consistency in Rule-Based Expert Systems *Int. Journal of Man-Machine Studies* 26(5):633-648.

Cruse, D.A. (1986) *Lexical Semantics* University Press, Cambridge.

David, J.M., Krivine, J.-P. and Simmons, R. (1993) Second Generation Expert Systems: A Step Forward in Knowledge Engineering In David, J.M., Krivine, J.-P. and Simmons, R., (eds) *Second Generation Expert Systems*, Springer, Berlin.

Davis, R. (1977) Meta-Rules: Reasoning About Control *Artificial Intelligence* 15:179-222.

De Greef, P., Breuker, J. and De Jong, T. (1988) *Modality: and Analysis of Functions, User Control and Communication in Knoweldge-Based Systems* Technical Report, University of Amsterdam.

Dimopoulos, Yannis and Kakas, Antonis, (1995) Learning Non-Monotonic logic Programs: Learning Exceptions. In Nada Lavrac and Stefan Wrobel (eds) *Machine Learning: Proceedings of the European Conference on Machine Learning ECML-95* Lecture Notes in Artificial Intelligence 912, Springer Verlag, Berlin, 122-137.

Eades, P. (1996) Graph Drawing Methods *Conceptual Structures: Knowledge Representation as Interlingua* (Eds. P.Eklund, G. Ellis and G. Mann), Springer Verlag, Berlin, 40-49.

Eades, P and Tamassia, R. (1989) Algorithms for Drawing Graphs: An Annotated Bibliography *Technical Report CS-89-90*, Dept. of Computer Science, Brown University., Providence, R.I.

Easterbrook, Steve (1991) Elicitation of Requirements from Multiple Perspectives *PhD Thesis*, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, London SW7 2BZ.

Easterbrook, S. and Nuseibeh, B. (1996) Using Viewpoints for Inconsistency Management *BCSEEE Software Engineering Journal* January 1996, 31-43.

Erdmann, Michael (1998) Formal Concept Analysis to Learn from the Sisyphus-III Material *11th Workshop on Knowledge Acquisition, Modeling and Management,* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:SIS-2.

Edwards, Glenn (1996) Reflective Expert Systems in Clinical Pathology *MD Thesis*, University of New South Wales.

Edwards, G., Compton, P., Malor, R, Srinivasan, A. and Lazarus, L. (1993) PEIRS: a Pathologist Maintained Expert System for the Interpretation of Chemical Pathology Reports *Pathology 25*: 27-34.

Edwards, G., Kang, B., Preston, P. and Compton, P. (1995) Prudent Expert Systems with Credentials: Managing the expertise of Decision Support Systems *Int. Journal Biomedical Computing* 40:125-132.

Edwards, G., Compton, P., Kang, B., Preston, P. and Lazarus, L. (1995) New Paradigms for Expert Systems in Health Care. *Proceedings of the Fourth Health Informatics of NSW Conference*, Primbee, 67-72.

Edwards, G., Kang, B., Preston, P. and Compton, P. (1995). Prudent expert systems with credentials: Managing the expertise of decision support systems. *International Journal Biomedical Computing* 40:125-132.

Ericsson, K.A. and Charness, N. (1997) Cognitive and Developmental Factors in Expert Performance In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 3-41.

Faneuf, R. and Kirk, S., (1988) A UIMS for Building Metaphoric User Interfaces in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corp.

Feldman, B.Z., Compton, P.J., and Smythe, G.A. (1989) Hypothesis testing: an Appropriate Task for Knowledge-Based Systems in *Proceedings of the 4th Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Canada.10.1-10.20.

Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (1997) A Preliminary Tour of Human and Machine Expertise in Context In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, xiii-xviii.

Fibak, J., Slowinski, K. and Slowinki, R. (1986) Rough Sets Based Decision Algorithm for Treatment of Duodenal Ulcer by *HSV Proceedings of 6th Internal Workshop on Expert Systems and their Applications* Avignon, France, 587-599.

Fickas, S., (1988) Development Tools for Rule-based Systems in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corp.

Finin, Tim and Klein, David (1989) Characterizing Knowledge Depth in Intelligent Safety Systems In Horn, Werner (ed), *Causal AI Models* 1990 Hemisphere Publishing Corp. New York.

Finkelstein, A.C.W., Goedicke, M., Kramer, J. and Niskier, C. (1989) Viewpoint Oriented Software Development: Methods and Viewpoints in Requirements Engineering In *Proceedings of the Second Meteor Workshop on Methods for Formal Specification* Springer Verlag, LNCS.

Finkelstein, A., Gabbay, D., Hunter, A, Kramer, J. and Nuseibeh, B. (1994) Inconsistency Handling in Multi-Perspective Specifications *IEEE Transactions on Software Engineering* 20(8):569-578.

Fischer, G. and Girgenshohn, A. (1990) End-User Modifiability in Design Environments In *Conference Proceeding of CHI'90, Human Factors in Computing Systems* ACM, New York (April 1990), 183-191.

Flaaten, Per O., McCubbrey, Donald J., O'Riordan, P. Declan and Burgess Keith. (1989) *Foundations of Business Systems.* The Dryden Press, Orlando.

Frakes, W.B and Fox, C.J.(1995) Sixteen Questions About Software Reuse *Communications of the ACM* June 1995, 38 (6):75-87.

Freidson, E. (1994) Nourishing Professionalism *Professionalism Reborn: Theory, Prophecy and Policy.* University of Chicago Press, Chicago, 199-216.

Gaines, B. R. and Shaw, M.L.G. (1983) Dialog Engineering In *Designing for Human-Computer Communication* Academic Press, 23-53.

Gaines, B. R. (1989a) An Ounce of Knowledge is Worth a Ton of Data: Quantitative Studies of the Trade Off Between Expertise and Data Based on Statistically Well-Founded Empirical Induction *Proceedings of the 6th International Workshop on Machine Learning* Morgan Kaufmann, San Mateo, California, 156-159.

Gaines, B. R. (1989b) Knowledge Acquisition: the Continuum Linking Machine Learning and Expertise Transfer *Proceedings of the 3rd European Workshop on Knowledge Acquisition for Knowledge-Based Systems* Paris, 90-101.

Gaines, B.R., (1991a) Induction and Visualization of Rules with Exceptions In J. Boose & B. Gaines (eds), *Proceedings of the 6th Banff AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.* Banff, Canada,Vol 1: 7.1-7.17.

Gaines, B.R. (1991b) An Interactive Visual Language for Term Subsumption Languages *IJCAI91: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, 24-30 August 1991, Darling Harbour, Sydney, 817-823.

Gaines, B.R., (1993) Modeling Enterprise in Individuals and Organizations in *Proceedings of the 8th Knowledge Acquisition for Knowledge-Based Systems Workshop.* Banff, Canada.

Gaines, B.R. (1995) Structured and Unstructured Induction with EDAGs, Fayyad, U.M. & Uthurusamy, R., Eds. Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95).Cambridge, Massachusetts: MIT Press.

Gaines, B.R. (1996) Transforming Rules and Trees into Comprehensible Knowledge Structures, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. & Uthurusamy, R., Eds. Advances in Knowledge Discovery and Data Mining. Cambridge, Massachusetts: MIT Press, 1996, 205-226.

Gaines, B.R. and Compton, P. (1992) Induction of Ripple-Down Rules, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence (AI'92)*, Hobart, Tasmania, World Scientific, Singapore, 349-354.

Gaines, B.R. and Compton, P. (1995) Induction of Ripple-Down Rules Applied to Modeling Large Databases, *Journal for Intelligent Information Systems* 5(3): 211-228.

Gaines, B. R. and Shaw, M.L.G. (1989) Comparing the Conceptual Systems of Experts *The 11th International Joint Conference on Artificial Intelligence***,** 633-638.

Gaines, B. R. and Shaw, M.L.G. (1993a) Knowledge Acquisition Tools Based on Personal Construct Psychology *Knowledge Engineering Review* 8(1):49-85.

Gaines, B.R. and Shaw M.L.G. (1993b) Eliciting Knowledge and Transferring it Effectively to a Knowledge-Based System, *IEEE Transactions on Knowledge and Data Engineering* 5(1) 4-14.

Gaines, B. R. and Shaw, M.L.G. (1995) Collaboration through Concept Maps *CSCL'95 Proceedings* September 1995.

Gambetta, W., (1995) Obtaining a Compromise View: Verification and Validation Problems in Integrating Knowledge Bases AI Department: School of Computer Science, University of New South Wales.

Ganter, B. (1988) Composition and Decomposition of Data In H. Bock (ed), *Classification and Related Methods of Data Analysis* North-Holland, Amsterdam, 561-566.

Ganter, B. and Wille, R. (1989) Conceptual Scaling In F. Roberts (ed), *Applications of Combinatorics and Graph Theory to the Biological Sciences* Springer, New York, 139-167.

Geertz, Clifford (1993) *Local Knowledge : Further Essays in Interpretive Anthropology* Fontana, London.

Gennari, J.H., Tu, S.W., Rothenfluh, T.E. and Musen, M.A. (1994) Mapping Domains to Methods in Support of Reuse *International Journal of Human-Computer Studies* 41:399-424.

Gentner, D. and Markman, A.B. (1994) Structural Alignment in Comparison: No Difference Without Similarity *Psychological Science* 5(3): 152-158.

Gil, Yolanda and Tallis, Marcelo (1997) A Script-Based Approach to Modifying Knowledge Bases In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 377-383.

Gomez-Perez, A. (1994) From Knowledge Based Systems to Knowledge Sharing: Evaluation and Assessment *Knowledge Systems Laboratory*, Stanford University.

Gonzalez, A.J. and Dankel, D.D. (1993) *The Engineering of Knowledge-Based Systems* Prentice Hall.

Greeno, J.G. and Moore, J.L. (1993) Situativity and Symbols: Response to Vera and Simon *Cognitive Science* 17:49-59.

Grosof, Benjamin N. (1997) Courteous Logic Program: Prioritized Conflict Handling for Rules. *IBM Research Report* RC 20836(92273)
Available from: http://www.research.ibm.com/people/g/grosof.

Grossner, C., Preece, A.D., Chandler, P.G., Radhakrishnan, T. and Suen, C.Y. (1993) Exploring the Structure of Rule Based Systems *Proceedings of the Eleventh National Conference on Artificial Intelligence*, MIT Press, Washington DC, 704-709.

Grossner, C., Gokulchander, P. Preece, A., and Radhakrishnan, T. (1996) Revealing the Structure of Rule-Based Systems *International Journal of Expert Systems: Research and Applications* 9(2):255-278.

Gruber, T. R.,(1993a) A Translation Approach to Portable Ontology Specifications *Knowledge Systems Laboratory*, Stanford University.

Gruber, T. R., (1993b) Toward Principles for the Design of Ontologies Used for Knowledge Sharing *Knowledge Systems Laboratory*, Stanford University.

Guarino, N., and Giaretta, P. (1995) Ontologies and Knowledge Bases: Towards a Terminological Clarification In Mars, N. (ed), *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing 1995,* IO Press, Amsterdam, 25-32.

Guha, T.V., and Lenat, D.B. (1990) CYC:A Mid-Term Report *AI Magazine* 11(3):32-59.

Gunnar, M Von Gnechten (1978) Changing a Frightening Toy into a Pleasant Toy by Allowing the Infant to Control it Actions *Developmental Psychology* 14:157-162.

Hammer, Eric (1995) Peirce on Logical Diagram *Transactions of the Charles S.Peirce Society* Fall 1995, Vol XXXI, 4:807-827.

Hamming, R.W. (1980) *Coding and Information Theory* Prentice-Hall, Englewood Cliffs, NJ.

Harmon, P. and King, D. (1985) *Expert Systems,* Wiley, New York.

Hayes, P.J., (1988) Using a Knowledge Base to Drive an Expert System Interface with a Natural Language Component in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corp.

Hayes-Roth, F., Waterman, D.A. and Lenat, D.B. (eds) (1983) *Building Expert Systems* Addison-Wesley, Reading, Massachusetts.

Heidegger, M (1962) *Being and Time* (Translated by John Macquarie and Edward Robinson), Harper and Row, New York.

Hemmann, Thomas (1992) KADS Models are Reusable -MoMo Allows Sharing In Nicolaas Mars (ed.) *Notes of the ECAI-92 Workshop on Knowledge Sharing and Reuse: Ways and Means* Vienna University of Economics and Business Administration, 3rd August 1992.

Hemmann, Thomas (1993a) Reusable Frameworks of Problem Solving IN Catherine Peyralbe (ed.) *IJACI-93 Workshop on Knowledge Sharing and Information Interchange* Chamberry, France, August 29, 1993.

Hemmann, Thomas (1993b) How Knowledge Engineering Can Benefit from Software Engineering with Respect to Reuse: Towards Reusable Knowledge Models In Panos Constantopolous, Yannis Vassiliou (eds) *Proceedings of ERICM Workshop on Methods and Tools for Software Reuse* Heraklion, Greece, October 29-30 1992, 212-227.

Hemmann, Thomas and Voss, Hans (1993a) A Reusable and Specializable Interpretation Model for Model-Based Diagnosis In Christiane Luckenhoff, Dieter Fensel, Rudi Studer (eds.) *Proceeding 3rd KADS Meeting* Siemens AG, Munich, March 8-9, 1993, 189-205.

Hemmann, Thomas and Voss, Hans (1993b) MoMo in the Large *Notes of AAAI-93 Workshop on Modeling in the Large***,** Washington D.C. July 11-16, 1993.

Helmbold, D., Sloan, R. and Warmuth, M.K. (1989) Learning Nested Differences of Intersection-Closed Concept Classes *Proceedings of the Workshop on Computational Learning Theory*, 41-56.

Hendler, J. A., (1988) *Expert Systems: The User Interface*. Ablex Publishing Corp.

Hender, J.A. and C. Lewis, (1988) Introduction: Designing Interfaces for Expert Systems in Hendler, J. A., *Expert Systems: The User Interface* Ablex Publishing Corp.

Herlea, D.E. (1996) User's Involvement in the Requirements Engineering Process *Proceedings 10th Knowledge Acquisition Workshop*, Banff, Canada.

Hochman, Z., Compton, P., Blumenthal, M. and Preston, P. (1996) Ripple-down rules: a potential tool for documenting agricultural knowledge as it emerges *In Proceedings of the 8th Australian. Agronomy Conference*, Toowoomba 1996, 313-316.

Hollands, J.G., Carey, T.T., Matthews, M.L. and McCann, C.A. (1989) Presenting a Graphical Network: A Comparison of Performance using Fisheye and Scrolling Views *Proceedings of 3rd International Conference on Human Computer Interaction* September 1989, 313-320.

Hopple, Gerald W. (1988) *The State of the Art in Decision Support Systems.* QED Information Sciences Inc., Wellesley.

Ignizio, James, P. (1991) *Introduction to Expert Systems: The Development and Implementation of Rule-Based Expert Systems* McGraw-Hill Inc, New York.

Ikeda, Mitsuru and Mizoguchi, Riichiro (1994) FITS: A Framework for ITS - A Computational Model of Tutoring *Journal of AI in Education* 5(3).

Ikeda, Mitsuru, Seta, Kazuhisa, Kakusho, Osamu and Mizoguchi, Riichiro (1996) An Environment for Building Conceptual Model of Problem Solving Repetition In Compton, P., Mizoguchi, R., Motoda, H. and Menzies, T. (eds) *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia, 210-225.

Jansen, B. and Compton, P. (1988) The Knowledge Dictionary: A Relational Tool for the Maintenance of Expert Systems *Proceeding of the International Conference on Fifth Generation Computer Systems FGCS'88*, 1159-1167.

Jansen, M.G., Schreiber A. Th. and Weilinga, B.J. Rocky III - Round 1 A Progress Report In Gaines, B. and Musen, M. (eds) *Eleventh Workshop on Knowledge Acquisition Modeling and Management (KAW'98)* Banff, Canada, April 18-23, 1998, SIS-5.

Jones, P.L.K. (1986) REVEAL: Addressing DSS and Expert Systems In Mitra, G. (ed.) *Computer Assisted Decision Making.* Elsevier Science Publishers, Amsterdam.

Kang, B. (1996*) Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules* PhD Thesis, School of Computer Science and Engineering, University of NSW, Australia.

Kang, B., Compton, P. and Preston, P. (1995) Multiple Classification Ripple Down Rules: Evaluation and Possibilities *Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. Feb 26 - March 3 1995, Vol 1: 17.1-17.20.

Kang, B., Gambetta, W. and Compton, P. (1996) Verification and Validation with Ripple-Down Rules *Int. J. Human-Computer Studies* 44 :257-269.

Kang, B., Yoshida, K, Motoda, H, and Compton, P. (1997) Help Desk with Intelligent Interface *Applied Artificial Intelligence* 11:611-631.

Kaplan, R and Rock, D. (1995) New Direction for Intelligent Tutoring *AI Expert* February 1995, 31-40.

Karunananda, Asoka (1993) Computer Modelling of Thought Processes, *International Journal of Computer Applications in Technology* 6(2/3):135-140.

Keen, Debby and Rajasekar, Arcot (1993) Rough Sets and Data Dependencies. V.S. Alagar, S. Bergler and F.Q. Dong (eds) *Proceedings of the SOFTEKS Workshop on Incompleteness and Uncertainty in Information System* Concordia University, Montreal, Canada 8-9 Oct. 1993, 87-101.

Kelly, G.A, (1955) *The Psychology of Personal Constructs* Norton, New York.

Kelly, G.A, (1970) A Brief Introduction to Personal Construct Theory. Bannister, D. (ed) *Perspectives in Personal Constructs Theory* London, Academic Press, 1-29.

Kent, R.E. (1993) Rough Concept Analysis In W. Ziarko (ed.), *Rough Sets, Fuzzy Sets and Knowledge Discovery. Proceedings of the International Workshop on Rough Sets and Knowledge Discovery (RSLD'93)* Banff, Alberta, Canada, October 12-15, Springer Verlag, Berlin, 248-255.

Kent, R.E. (1996) Rough Concept Analayis: A Synthesis of Rough Sets and Formal Concept Analysis, *Fundamenta Informaticae* 27:169-181.

Kidd, A.L. (1985) What do Users Ask? - Some Thoughts on Diagnostic Advice In M. Merry, editor *Expert Systems '85* Cambridge University Press, 9-19.

Kidd, Alison L. and Sharpe, William P. (1987) Goals for Expert System Research: An Analysis of Tasks and Domains In *Expert System IV* Cambridge University Press, 146-152.

Kieras, David E. and Bovair, Steven (1984) The Role of a Mental Model in Learning to Operate a Device *Cognitive Science* 8:255-273.

Kivinen, J., Mannila, H. and Ukkonen, E. (1993) Learning Rules with Local Exceptions *Proceedings of the European Conference on Computational Learning Theory COLT'93.*

Klein, Michel and Methlie, Leif B., (1990) *Expert Systems a Decision Support Approach: With Applications in Management and Finance.* Addison Wesley, Cornwall.

Kolodner, J.L and Simpson, R.L. (1984*) Retrieval and Organisation Strategies in Conceptual Memory: A Computer Model* Erlbaum, Northvale, NJ.

Kolodner, J.L and Simpson, R.L. (1989) The MEDIATOR: Analysis of an Early Case-Based Problem Solver *Cognitive Science* 13(4): 507-549.

Kolodner, Janet (1993) *Case-Based Reasoning* Morgan Kaufman Publishers Inc., San Mateo, CA.

Kono, Yasuyuki, Ikeda, Mitsuru and Mizoguchi, Riichiro (1994) A Nonmonotonic Inductive Student Modelling System *Journal of AI in Education* 5(3).

Kowalski, R. and Sadri, F. (1991) Logic Programs with exceptions *New Generation Computing* 9:387-400.

Kremer, Rob (1998) Visual Languages for Knowledge Representation In Gaines, B. and Musen, M. (eds) *Eleventh Workshop on Knowledge Acquisition Modeling and Management (KAW'98)* Banff, Canada, April 18-23, 1998, VKM-6.

Kriegsman, Mark (1993) Building a Case-Based Help Desk Application *IEEE Expert* December 1993, 18-26.

Krysinski, J. (1990) Rough Sets Approach to Analysis of Relationship Between Structure and Activity of Quaternary Imidazolium Compounds. *ArzeneimittelForschung Drug Research,* 40:795-799.

Kuo, F.-Y. (1988) Combining Expert Systems and the Bayesian Approach to Support Forecasting. *Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences.* The Computer Society of the IEEE, Washington.

Laird, J., Newell, A., and Rosenbloom, P. (1987) SOAR: An Architecture for General Intelligence *Artificial Intelligence* 33:1-64.

Lal, Harbans (1989) Engineering Farm Knowledge for a Seamless Decision Support System*. Phd Thesis*, University of Florida, USA, Florida.

Lambiotte, J.G., Dansereau, D. F., Cross, D.R. and Reynolds, S.B. (1984) Multirelational Semantic Maps *Educational Psychology Review* 1(4):331-367.

Langlotz, Curtis P. and Shortliffe, Edward E. (1983) Adapting a Consultation System to Critique User Plans. *International Journal of Man-Machine Studies* 19:479-496.

Larkin, Jill H. and Simon, Herbert, A. (1987) Why a Diagram is (Sometimes) Worth Ten Thousand Words *Cognitive Science* 11:65-99.

Lauderdale, K (1995) URL:http://smi-web.stanford.edu/projects/t-helper/index.html.

Lave, J., (1988) *Cognition in Practice: Mind, Mathematics, and Culture in Everyday Life* Cambridge, UK, Cambridge University Press.

Leake, David B., Kinley, Andrew and Wilson, David (1997) Case-Based Similarity Assessment: Estimating Adaptability from Experience In *Proceedings of the Fourteenth National Conference* *on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts.

Lee, M. (1996) From Multiple Representations to Causal Explanations *PhD Thesis*, University of New South Wales.

Lee, M. and Compton, P. (1995) From Heuristic Knowledge to Causal In X. Yao (ed), *Explanations Proceedings of Eighth Australian Joint Conference on Artificial Intelligence AI'95***,** 13-17 November 1995, Canberra, World Scientific, 83-90.

Lehner, P.E. and M.M. Kralj, (1988) Cognitive Impacts of the User Interface in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corp.

Leigh, William E. and Doherty, Michael E. (1986) *Decision Support & Expert Systems*. Cincinnati: South-Western Publishing Co.

Li, X. (1991) What's so bad about rule-based programming ? *IEEE Software*, September 1991, 103-105.

McCarthy, J. (1991) *Notes on Formalizing Context* Technical Report. Computer Science Department, Stanford University.

Macquarie Dictionary, The Concise (1982) Doubleday Australia Pty. Limited, Lane Cove.

Maes, P (1987) Concepts and Experiments in Computational Reflection. In *OOPSLA'87, Object-Oriented Programming Systems, Languages and Applications. Conference Proceedings*. Ed: Norman Meyrowitz. Special Issue of SIGPLAN Notices. Vol. 22. December 1987. ACM Press, 147-155.

Mahidadia, A., Compton, P. and Sammut, C., (1994) Helping Researchers to Construct Scientific Models: A Tool from Inductive Logic Programming, *Proceedings of the 4th Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop*, Tokyo, Japan, November 1994.

Mani, Inderjeet and Bloedorn, Eric (1997) Multi-Document Summarization by Graph Search and Matching In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts.

Mansuri, Y., Kim, J.G., Compton, P. and Sammut, C. (1991*).* A comparison of a manual knowledge acquisition method and an inductive learning method *Australian Workshop on Knowledge Acquisition for Knowledge Based Systems*, Pokolbin (1991), 114-132.

Markman, A.B. and Gentner, D. (1996) Commonalities and Differences in Similarity Comparisons *Memory and Cognition* 24(2):235-249.

Marques, D., G. Kliner, G. Dallemangne, P. Gautier, J. McDermott, and D. Tung. (1991) More data on useable and reusable programming constructs *in 6th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop* Banff, Canada.

Marques, D., Dallemangne, G., Kliner, G., McDermott, J. and Tung, D. (1992) Easy Programming Empowering People to Build Their Own Applications *IEEE Expert* June 1992, 16-29.

Marr, D. (1982) *Vision: A Computational Investigation in the Human Representation and Processing of Visual Information* W.H. Freeman, San Francisco.

Martinez-Bejar, R., Benjamins, R. and Martin-Rubio, F. (1997) Designing Operators for Constructing Domain Knowledge Ontologies, Enric Plaza and Richard Benjamins (Eds), *Knowledge Acquisition, Modeling and Management*. Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 159-173.

Martinez-Bejar, R., Benjamins. R., Compton,, P., Preston, P. and Martin-Rubio, F. (1998) A Formal Framework to Build Domain Knowledge Ontologies for Ripple-Down Rules-Based Systems. In Gaines, B. and Musen, M. (eds) *Eleventh Workshop on Knowledge Acquisition, Modeling and Management (KAW'98),* Banff, Alberta, Canada, 18-23 April, 1998, Vol 2:SHARE.13.

Martinez-Bejar R., Shiraz G. M., and Compton P. (1988) Using Ripple Down Rules-based Systems for Acquiring Fuzzy Domain Knowledge, In Gaines, B. and Musen, M. (eds) *Eleventh Workshop on Knowledge Acquisition, Modeling and Management (KAW'98),* Banff, Alberta, Canada, 18-23 April, 1998, Vol 1:KAT.2.

Marvin, Lisa Johanna (1990) Intelligent Decision Support Systems: Assisting Decision-makers in Multiple Domains. *MSc Thesis*, University of Texas, USA, Arlingtion.

McDermott, J. (1988) Preliminary Steps Toward a Taxonomy of Problem-Solving Methods In Marcus, S (ed) *Automating Knowledge Acquisition for Expert Systems* Kluwer Academic Publishers, 225-256.

Mehrotra, Mala (1995a) Analyzing Knowledge-Based Systems Using Multi-Viewpoint Clustering Analysis *Journal of Systems and Software, Special Issue on Quality Assurance in Knowledge-Based Systems* June 1995.

Mehrotra, Mala (1995b) Requirements and Capabilities of the Multi-Viewpoint Clustering Analysis Methodolgoy. *IJCAI'95 Workshop on Verification and Validation of Knowledge-Based Systems* August 1995, Montreal, Canada.

Menzies, T.J. (1992) Maintaining Procedural Knowledge: Ripple-Down-Functions, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence (AI'92),* Hobart, Tasmania, World Scientific, Singapore.

Menzies, T.J. (1995) Limits to Knowledge-Level-B Modeling (and KADS) In X. Yao (ed), *Proceedings of Eighth Australian Joint Conference on Artificial Intelligence AI'95***,** 13-17 November 1995, Canberra, World Scientific, 459-465.

Menzies, T.J. and Compton, P. (1995) The (Extensive) Implications of Evaluation on the Development of Knowledge-Based Systems *Proceedings of the 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge Based Systems*. SRDG Publications, Banff Canada, 18.1-18.20.

Menzies T.J., (1996) Assessing Responses to Situated Cognition *Proceedings of the 10th Knowledge Acquisition Workshop,* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada.

Menzies, T.J. (1997a) OO Patterns: Lessons from Expert Systems *Software-Practice and experience* Vol 27 (12) 1457-1478 (December 1997).
Available from http://www.cse.unsw.edu.au/~timm/pub/docs/97probspatt.ps.gz.

Menzies, T. J. (1997b) Knowledge Maintenance: State of the Art. *Knowledge Engineering Review (to appear).*
Available from http://www.cse.unsw.edu.au/~timm/pub/docs/96aim.ps.gz.

Menzies, T. and Compton, P. (1997). Applications of Abduction: Hypothesis Testing of Neuroendocrinological Qualitative Compartmental Models. *Artificial Intelligence in Medicine* 10:145-175.

Menzies, T.J. (1998) Evaluation Issues for Problem Solving Methods *11th Workshop on Knowledge Acquisition, Modeling and Management*, Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:EVAL-5.

Meseguer, P. (1992) Incremental Verification of Rule-based Expert Systems In Neumann, B. (ed) *10th European Conference on Artificial Intelligence,* 829-834.

Michalski, R.S. and Chilausky, R.L. (1980) Learning by being told and learning from examples: An experimental comparison of two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *International Journal of Policy Analysis and Information Systems*, Volume 4.

Miller, P.L. (1986) *Expert Critiquing Systems: Practice-Based Medical Consultation by Computer* Springer-Verlag, New York.

Mittal, S., Bobrow, D.G. and J. de Kleer, (1988) *DARN*: Toward a Community Memory for Diagnosis and Repair Tasks in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corp, 57-63.

Mittal, Vibhu O., (1997) Visual Prompts and Graphical Design: A Framework for Exploring the Design Space of 2-D Charts and Graphs In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts.

Mizoguchi, R., Vanwelkenhuysen, J. and Ikeda, M. (1995) Task Ontology for Reuse of Problem Solving Knowledge, *Second International Conference on Building and Sharing of Very Large-Scale Knowledge Bases, Proceedings of the KB and KS-95,* 46-59.

Mulholland, M., Preston, P., Sammut, C., Hibbert, B. and Compton, P. (1993) An Expert System for Ion Chromatography Developed using Machine Learning and Knowledge in Context *Proceedings of the 6th Int. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems* Edinburgh.

Murphy, Patrick M. and Pazzani, Michael J. (1994) Revision of Production System Rule-Bases In W.W. Cohen and H. Hirsch, (eds.) *Machine Learning: Proceedings of the Eleventh International Conference,* New Brunswick, NJ, Morgan Kaufmann, 199-207.

Musen, M.A. (1993) An Overview of Knowledge Acquisition In David, J.M., Krivine, J.-P. and Simmons, R., editors *Second Generation Expert Systems* Springer, Berlin, 273-298.

Musen, M.A., L.M. Fagan and E.H. Shortliffe, (1988) Graphical Specification of Procedural Knowledge for an Expert System in Hendler, J. A. (ed), *Expert Systems: The User Interface* Ablex Publishing Corp.

Musen, M.A., Schreiner, J.G., Campbell, K.E., Tu, S.W., Shortliffe, E.H., Fagan, L.M. T-Helper I: An electronic medical record supporting the treatment of AIDS. *Proceedings of the Seventeenth Annual Symposium on Computer Applications in Medical Care*, p. 958 (abstract). Washington DC, November 1993.
Available from http://smi-web.stanford.edu/projects/t-helper/index.html

Narayanaswamy, K. and Goldman, N. (1992) “Lazy Consistency”: A Basis for Cooperative Software Development *Proceedings of International Conference on Computer-Supported Cooperative Work (CSCW’92)* Toronto, Ontario, Canada, 31 October- 4 November, ACM SIGCHI & SIGOIS, 257-264.

Nau, D. and M. Gray, (1988) Hierarchical Knowledge Clustering: A Way to Represent and Use Problem-solving Knowledge in Hendler, J. A., *Expert Systems: The User Interface* Ablex Publishing Corp.

Nebel, B. (1991) Terminological Cycles: Semantics and Computational Properties In John Sowa (ed) *Principles of Semantic Networks: Explorations in the Representation of Knowledge* Morgan Kaufmann Publishers, Inc. California, 331-361.

Neches, R., R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator and W.R. Swartout, (1991) Enabling Technology for Knowledge Sharing *AI Magazine* Fall 1991:36-56.

Newell, A. (1982) The Knowledge Level *Artificial Intelligence* 18:87-127.

Newell, A. and Simon, H.A. (1976) Computer Science as Empirical Inquiry: Symbols and Search, *Communications of the ACM*, 19(3), 113-126.

Nguyen, T.A., Perkins, W.A., Laffey, T.J. and Pecora, D. (1987) Knowledge Base Verification *AI Magazine* Summer 1987, 69-75.

Nii, H.P. (1986) Blackboard System (Part 2): Blackboard Application Systems, Blackboard Systems from Knowledge Engineering Perspective *AI Magazine* Summer 1986.

Norman, D.A.(1986) Cognitive Engineering In Norman, D.A. and Draper, S.W. (eds.) *User Centered System Design: New Perspectives on Human-Computer* Interaction Lawrence Erlbaum Associates.

Norman, D.A.(1988) *Psychology of Everyday Things* Basic Books, New York.

Norman, D.A.(1993) Cognition in the Head and in the World: An Introduction to the Special Issue of Situated Action *Cognitive Science* 17:1-6.

Nosek, J.D. and Roth, I. (1990) A Comparison of Formal Knowledge Representation Schemes as Communication Tools: Predicate Logic vs. Semantic Network *International Journal of Man-Machine Studies* 33:227-239.

Nuseibeh, B. (1996) To Be And Not To Be: On Managing Inconsistency in Software Development, *Proceedings of 8th International Workshop on Software Specification and Design (IWSSD-8),*Scloss Velen, Germany, 22-23 March 1996, IEEE CS Press, 164-169.

O'Keefe, R.M., Balci, O. and Smith, E.P. (1987) Validating Expert System Performance *IEEE Expert*, 81-89.

O'Keefe, R.M. and O'Leary, E.E. (1993) Expert System Verification and Validation: A Survey and Tutorial *Artificial Intelligence Review* 66: 273-309.

Partridge, D. (1978) A Philosophy of Wicked Problem Implementation, *Proceedings AISB/GI Conference on Artificial Intelligence,* Hamburg.

Patel, V.L. and Ramoni, M.E. (1997) Cognitive Models of Directional Inference in Expert Medical Reasoning In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 67-99.

Patil, R. S., Fikes, R. E., Patel-Schneider, P. F., McKay, D., Finin, T., Gruber, T. R. and Neches, R., (1992) The DARPA Knowledge Sharing Effort: Progress Report In C. Rich, B. Nebel and Swartout, W., *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference* Cambridge, MA, Morgan Kaufman.

Patil, R.S., Szolivits, P. and Schwartz, W.B. (1984) Causal Undertanding of Patient Illness in Medical Diagnosis In W.J.Clancey and E.H. Shortliffe, eds, *Readings in Medical Artificial Intelligence,* Addison-Wesley, Reading, MA, 334-360.

Pawlak, Zdzislaw (1982) Rough Sets *International Journal of Information and Computer Sciences*, 11:341-356.

Pawlak, Zdzislaw (1991) *Rough Sets: Theoretical Aspects of Reasoning about Data* Kluwer Academic Publishers, Dordrecht.

Perkins, W., Laffey, T. and Pecora, D (1989) Knowledge Base Verification *AI Magazine* 8(2):69-75.

Petre, M. and Green, T.R.G. (1993) Learning to Read Graphics: Some Evidence that 'Seeing' an Information Display is an Acquired Skill *Journal of Visual Languages and Computing* 4(1):55-70.

Pew, R.W. (1988) Human Factors Issues in Expert Systems In Helender, M. (ed) (1988*)* *Handbook of Human-Computer Interaction* Elsevier Science Publishers B.V. North-Holland.

Pew, R.W.and Rollins, A.M. (1975) Dialog Specification Procedures (Report 3129 Revised) Bolt Baranek and Newman, Cambridge, MA.

Philippakis, A.S. (1988) Structured What If Analysis in DSS Models. In *Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences.* Washington: The Computer Society of the IEEE.

Piatetsky-Shapiro, G. and Matheus, C.J. (1994) The Interestingness of Deviations In *Proceedings AAAI'94 Workshop on Knowledge Discovery in Databases,* 25-36.

Pirlein, T and Studer, R., (1994) *KARO*: An Integrated Environment for Reusing Ontologies In Steels, L. Schreiber, G. and Van de Velde, W. (eds) *A Future for Knowledge Acquisition Proceedings of the 8th European Knowledge Acquisition Workshop, EKAW'94*, Lecture Notes in Artificial Intelligence 867, Springer Verlag, Berlin, 200-225.

Pollack, M.E., Hirschberg, J. and Weber, B. (1982) User Participation in the Reasoning Processes of Expert Systems *AAAI-82.*

Popper, K. (1963) *Conjectures and Refutations* Routledge and Kegan Paul, London.

Preece, A.D., Shinghal, R and Batarekh, A. (1992) Principles and Practice in Verifying Rule-Based Systems *The Knowledge Engineering Review 7*:115-141.

Preston, P. (1992) Expert Systems, Knowledge Acquisition and Knowledge Modification *Honours thesis,* School of Electrical Engineering, University of NSW.

Preston, P., Edwards, G., Compton, P. and Litkouhi, D. (1994) An Expert System for Time Course Data with Refinement in Context *Artificial Intelligence in Medicine: Interpreting Clinical Data AAAI 1994, Spring Symposium Series***,** Stanford University, 121-124.

Puerta, A. R, Egar, J.W., Tu, S.W. and Musen, M.A. (1992) A Mulitple Method Knowledge Acquisition Shell for Automatic Generation of Knowledge Acquisition Tools *Knowledge Acquisition* 4(2):171-196.

Puppe, F. (1993) *Systematic Introduction to Expert System Knowledge Representation and Problem Solving Methods* Springer Verlag, Berlin.

Quinlan, J. Ross (1988) Semi-autonomous acquisition for pattern-based knowledge. *Teaching Solving and Learning* The Rand Corporation, 192-207.

Quinlan, J.R. (1990) Learning logical definitions from relations *Machine Learning* 5:239-266.

Quinlan, J. Ross (1992) *C4.5 Programs for Machine Learning* Morgan Kaufman Publishers Inc.

Rademakers, P. and Vanwelkenhuysen, J. (1993) Generic Models and Their Support in Modeling Problem Solving Behaviour In David, J.M., Krivine, J.-P. and Simmons, R., (eds) *Second Generation Expert Systems,* Springer, Berlin, 350-375.

Ram, A. and Cox, M.T. (1994) Introspective Reasoning Using Meta-Explanations for Multistrategy Learning R.S. Michalski and G. Tecuci, (eds), *Machine Learning: A Multistrategy Approach IV* Morgan Kaufmann, San Mateo, CA, 349-377.

Ramadan, Z., Compton, P., Z., Preston, P., Le-Gia, T. Chellen, V., Mulholland, M. Hibbert, D.B., Haddad, P.R and Kang, B.(1997a) From Multiple Classification Ripple Down Rules to Configuration Ripple Down Rules. *Proceedings Australian Knowledge Acquisition Workshop*

(AKAW’97) in conjunction with AI’97, University of Western Australia, Perth, Australia, 5.1-5.17.

Ramadan, Z., Mulholland, M., Z., Hibbert, D.B., Preston, P., Compton, P., and Haddad, P.R (1997b) Towards an Expert System in Ion Chromatography Using Multiple Classification Ripple Down Rules (MCRDR). *Poster presented at International Ion Chromatography Symposium,IICS’97),* September 14-17, Santa Clara, CA. USA.

Ramesh, B. and Dhar, V. (1992) Supporting Systems Development by Capturing Deliberations During Requirments Engineering *IEEE Transactions on Software Engineering* 18(6):498-510.

Rector, A. L. (1989) Helping with a Humanly Impossible Task: Integrating KBS into clinical care In *Proceedings of the Second Scandinavian Conference on AI***,** SCAI’89.

Reubinstein, H.B. and Waters, R.C. (1989) The Requirements Apprentice: An Initial Scenario *Proceedings of the fifth IEEE International Workshop on Software Specification and Design* Pittsburgh, Penn.

Richards, Deborah C. (1994) The Adaptation of Expert Systems to Provide ‘What-if’ Analysis in the Decision Support System Style *MAppSc Thesis,* Charles Sturt University, Wagga Wagga, Australia.

Richards, D. and McDonald, C. (1995) Bridging the Expert System - Decision Support System Gap: Adapting and Reusing an Existing Expert System *Proceedings of Sixth Australasian Conference on Information Systems ACIS'95* 27-29 September 1995, Curtin University, Western Australia.

Richards, D and McDonald, C. (1995) Changing the System Style of Expert Systems In Craig McDonald (ed) *Proceedings of the Workshop on AI and the Environment* 13 November 1995, Australian Defence Force Academy, Canberra, Australia.

Richards, D. and McDonald, C. (1995) Adapting Expert Systems to Behave like Decision Support Systems In Andrew Blair, John Debenham and John Zeleznikow (eds*) Proceedings of the First Australian Workshop on Intelligent Decision Support Systems* 13 November 1995, Australian Defence Force Academy, Canberra, Australia, 36-45.

Richards, D. and McDonald, C. (1995) Bridging the Expert System - Decision Support System Gap: Reuse of Knowledge in an Existing Expert System In Xin Yao (ed) *Poster Proceedings of Eighth Australian Joint Artificial Intelligence Conference AI'95* 13-17 November 1995, Australian Defence Force Academy, Canberra, Australia, 33-40.

Richards, D ., Gambetta, W. and Compton, C (1996) Using Rough Set Theory to Verify Production Rules and Support Reuse In Antoniou, G. (ed) *Proceedings of the Verification, Validation and Refinement of KBS Workshop, PRICAI’96* 26-30 August 1996, Cairns, Australia, Griffith University, 57-66.

Richards, D and Compton, C (1996) Building Knowledge Based Systems that Match the Decision Situation Using Ripple Down Rules, *Intelligent Decision Support IDS’96* 9th September, 1996, Monash University, 114-126.

Richards, D., Chellen, V. and Compton, C (1996) The Reuse of Ripple Down Rule Knowledge Bases: Using Machine Learning to Remove Repetition In Compton, P., Mizoguchi, R., Motoda,

H. and Menzies, T. (eds) *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia, 293-312.

Richards, D and Compton, P, (1997a) Combining Formal Concept Analysis and Ripple Down Rules to Support the Reuse of Knowledge *Proceedings Software Engineering Knowledge Engineering SEKE'97*, Madrid 18-20 June 1997, 177-184.

Richards, D and Compton, P, (1997b) Finding Conceptual Models to Assist Validation In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation Workshop* Technical Report WS-97-01, July 28, Providence, Rhode Island. AAAI Press: Menlo Park, CA, 53-62.

Richards, D. and Compton, P. (1997c) Uncovering the Conceptual Models in Ripple Down Rules In Dickson Lukose, Harry Delugach, Marry Keeler, Leroy Searle, and John F. Sowa, (eds) (1997), *Conceptual Structures: Fulfilling Peirce's Dream, Proceedings of the Fifth International Conference on Conceptual Structures (ICCS'97),* August 3 - 8, University of Washington, Seattle, USA, Lecture Notes in Artificial Intelligence, Number 1257, Springer Verlag, Berlin, 198-212.

Richards, D. and Compton, P. (1997d) Knowledge Acquisition First, Modelling Later, In Enric Plaza and Richard Benjamins (eds), *Knowledge Acquisition, Modeling and Management*. 10th European Workshop, EKAW'97, Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 237-252.

Richards. D and Compton. P. (1997e) Situated Expert Systems *Proceedings 4th Conference of the Australasian Cognitive Science Society* ACS'97, Newcastle September, 1997.

Richards. D and Compton. P. (1997f) Building Adaptable, Reflective Knowledge Based Systems that Address the Problems of First Generation Expert Systems In Abdul Sattar (ed) *Poster Proceedings 10th Australian Joint Conference on Artificial Intelligence* AI'97, Perth 2-4 December, 1997, 50-55.

Richards, D. and Menzies, T. (1997) Extending Knowledge Engineering to Requirements Engineering from Multiple Perspectives In Menzies, T., Richards, D. and Compton, P. (eds) *Proceedings of 3rd Australian Knowledge Acquisition Workshop AKAW'97 in conjunction with AI'97*, University of Western Australia, Perth, Australia, 3.1-3.16.

Richards, D. (1998a) Ripple Down Rules with Formal Concept Analysis: A Comparison to Personal Construct Psychology In Gaines, B. and Musen, M. (eds) *11th Workshop on Knowledge Acquisition, Modeling and Management (KAW'98),* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:KAT-4.

Richards, D. (1998b) Using AI to Resolve the Republican Debate In Slaney, J., Antoniou. G and Maher, M.J. (eds) *Poster Proceedings of Eleventh Australian Joint Artificial Intelligence Conference AI'98* 13-17 July 1998, Griffith University, Nathan Campus, Brisbane, Australia, 109-120.

Richards, D. (1998c) An Evaluation of the Formal Concept Analysis Line Diagram In Slaney, J., Antoniou. G and Maher, M.J. (eds) *Poster Proceedings of Eleventh Australian Joint Artificial Intelligence Conference AI'98* 13-17 July 1998, Griffith University, Nathan Campus, Brisbane, Australia, 121-133.

Richards, D. (1998d) Providing Multiple Interfaces to Support Multiple Uses in Knowledge Based Systems using Ripple Down Rules *Poster Proceedings of the Pacific Rim International Conference on Artificial Intelligence* November 1998 (to appear).

Richards, D. (1998e) Assisting Political Decision Making, In Burstein, F. Linger, H and Smith, H. (eds) *Proceedings of the Australian Workshop on Intelligent Decision Support (AWIDS'98*), Sydney 29th September, 1998, Monash University, 59-71.

Richards, D (1998f) A New Look at Knowledge-Based System Explanation *Proceedings of the Pacific Knowledge Acquisition Workshop (PKAW'98),* Singapore, 22-23rd November 1998, (to appear).

Richards, D (1998g) Reverse Requirements Engineering from Multiple Perspectives *Proceedings of the Australasion Pacific Workshop on Intelligent Software Engineering (APWISE'98),* Singapore, 24th November 1998, (to appear).

Richards. D and Compton. P. (1998a) Taking Up the Situated Cognition Challenge with Ripple Down Rules *International Journal of Human Computer Studies, Special Issue on Situated Cognition.* (To appear).

Richards. D and Compton, P. (1998b) An Alternative Verification and Validation Technique for an Alternative Knowledge Representation and Acquisition Technique, *Journal of Knowledge Based Systems* (to appear).

Richards, D. and Menzies, T. (1998) Extending the SISYPHUS III Experiment from a Knowledge Engineering to a Requirements Engineering Task In Gaines, B. and Musen, M. (eds) *11th Workshop on Knowledge Acquisition, Modeling and Management (KAW'98),* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:SIS-6.

Riel, M.M., Levin, J.A. and Miller-Souviney, B. (1984) Learning with Interactive Media: Dynamic Support for Students and Teachers In R.W. Lawler and M.Yazdani, (eds), *Artificial Intelligence and Education,* Vol 1, Ablex Publishers: Norwood NJ.

Rivest, R.L. (1987) Learning Decision Lists *Machine Learning* 2(2):229-246.

Roberts, D.D. (1973) *The Existential Graphs of Charles S. Peirce* Mouton and Co. N.V. The Hague.

Rosenfield, I., (1988) *The Invention of Memory: A New View of the Brain,* Basic Books, New York.

Roth, E.M., Bennett, K.B. and Woods, D.D. (1987) Human Interaction with an 'Intelligent' Machine *International Journal of Man-Machine Studies* 27:479-525.

Rothenfluh, T.E., Gennari, J.H., Eriksson, H., Puerta, A.R., Tu, S.W., and Musen, M.A. (1994) Reusable Ontologies, Knowledge-Acquisition Tools, and Performance Systems: PROTEGE-II Solutions to Sisyphus-2 *Knowledge Acquisition Workshop,* Banff, Alberta, Canada, 43.1-43.30.

Rumbaugh, James (1994) Getting Started: Using Use Cases to Capture Requirements *JOOP* September 1994, 8-23.

Runkel, J. (1995) Analysing Tasks to Build Reusable Model-Based Tools *Proceedings of the 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop* Banff, Canada.

Salle, J.M. and Hunter, J., (1990) Computer/User Cooperation Issues for Knowledge-Based Systems: A Review, *Technical Report AUCS/TR9003*, Aberdeen University.

Santos, Eugene Jr., Banks, Darwin and Banks, Sheila (1997) BVAL: Probabilistic Knowledge-Base Validation In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation of Knowledge-Based Systems Workshop,* July 28, 1997, Providence, Rhode Island, Technical Report WS-97-01, AAAI Press, Menlo Park, California, 12-22.

Sarker, Manojit and Brown, Marc G. H (1994) Graphical Fisheye Views *Communications of the ACM* December 1994 37(12):73-84.

Salter, W.J. (1988) Human Factors in Knowledge Acquisition In Helender, M. (ed*) Handbook of Human-Computer Interaction* Elsevier Science Publishers B.V. North-Holland.

Schank, R.C. and Leake, D.B. (1989) Creativity and Learning in a Case-Based Explainer *Artificial Intelligence* 40: 353-385.

Scheffer, T. (1996) Algebraic Foundation and Improved Methods of Induction of Ripple Down Rules Repetition In Compton, P., Mizoguchi, R., Motoda, H. and Menzies, T. (eds) *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia, 279-292.

Schmalhofer, F., J.S. Aitken and L.E. Bourne Jr., (1994) Beyond the Knowledge Level: Descriptions of Rational Behaviour for Sharing and Reuse In Steels, L. Schreiber, G. and Van de Velde, W. (eds) *A Future for Knowledge Acquisition Proceedings of the 8th European Knowledge Acquisition Workshop, EKAW'94,* Lecture Notes in Artificial Intelligence 867, Springer Verlag, Berlin, 83-103.

Schneiderman, B. (1980) *Software Psychology: Human Factors in Computer and Information Systems* Winthrop Publishers, Cambridge, MA.

Schon, D.A. (1987) *Educating the Reflective Practitioner* Jossey-Bass, San Francisco, CA.

Schreiber, G., Weilinga, B. and Breuker (eds) (1993) KADS: A Principles Approach to Knowledge-Based System Development *Knowledge-Based Systems* Academic Press, London, England.

Schultheis, Robert and Sumner, Mary. (1992) *Management Information Systems: The Manager's View.* 2nd ed. Irwin Publishers, Homewood.

Schubert, L. K. (1991) Semantic Nets are in the Eye of the Beholder In John Sowa (ed) *Principles of Semantic Networks: Explorations in the Representation of Knowledge* Morgan Kaufmann Publishers, Inc. California, 95-108.

Schwanke, R.W. and Kaiser, G.E. (1988) Living with Inconsistency in Large Systems *Proceedings of the International Workshop on Software Version and Configuration Control* Grassau, Germany, 27-29 January 1988, B.G. Teubner, Stuttgart, 98-118.

Schwartz, Rina and Kraus, Sarit (1997) Negotiation and Data Allocation in Multi-Agent Environments In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 29-35.

Scott, J and Weckert, J. (1995) Explanation in Expert Systems *Poster Proceedings of Eighth Joint Conference on Artificial Intelligence AI'95* In Xin Yao, (ed) Australian Defence Force Academy, 13-17 November 1995, 185-192.

Seifert, C.M., Patalano, A.L., Hammond, K.J. and Converse, T.M. (1997) Experience and Expertise: The role of Memory in Planning for Opportunities In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 101-123.

Shadbolt, N., (1996)URL:http://www.psyc.nott.ac.uk/aigr/research/ka/SisIII

Shalin, V.L., Geddes, N.D., Bertram, D., Szczepkowski, M.A. and DuBois, D. (1997) Expertise in Dynamic, Physical Task Domains In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 195-217.

Shapiro (1981) Inductive Inference of Theories from Facts *Computer Science Department, Technical Report TR 192,* Yale University.

Shapiro, S. C. (1991) Cables, Paths, And "subconscious" Reasoning in Propositional Semantic Networks In John Sowa (ed) *Principles of Semantic Networks: Explorations in the Representation of Knowledge* Morgan Kaufmann Publishers, Inc. California, 137-156.

Shastri, L. (1991) Why Semantic Networks ? In John Sowa (ed) *Principles of Semantic Networks: Explorations in the Representation of Knowledge* Morgan Kaufmann Publishers, Inc. California, 109-136.

Shaw, M.L.G., (1980) *On Becoming a Personal Scientist: Interactive Computer Elicitation of Personal Models of the World* Academic Press, London.

Shaw, M.L.G., (1988) Validation in a Knowledge Acquisition System with Multiple Experts In *Proceedings of the International Conference on Fifth Generation Computer Systems* edited by ICOT, 1259-1266.

Shaw, M.L.G. and Gaines, B.R., (1988) A Methodology for Recognising Consensus, Correspondence, Conflict and Contrast in a Knowledge Acquisition System in *Proceedings of the 3rd Knowledge Acquisition for Knowledge-Based Systems Workshop,* November 1988, Banff, Canada.

Shaw, M.L.G. and Gaines, B.R., (1991a) Using Knowledge Acquisition Tools to Support Creative Processes In *Proceedings of the 6th Knowledge Acquisition for Knowledge-Based Systems Workshop,* Banff, Canada.

Shaw, M.L.G. and Gaines, B.R., (1991b) A Methodology for Analyzing Terminological and Conceptual Differences in Language use Across Communities In Kohler, R. & Rieger, B.B. (eds). Contributions to Quantitative Linguistics. Proceedings of the First Quantitative Linguistics Conference, QUALICO), Trier, 1991, Kluwer, Dordrecht, 91-138.

Shaw, M.L.G. and Gaines, B.R., (1992) Kelly's "Geometry of Psychological Space" and its Significance for Cognitive Modeling, *The New Psychologist*, 23-31, October, 1992.

Shaw, M.L.G. and Gaines, B.R.. (1994) Personal Construct Psychology Foundations for Knowledge Acquisition and In Steels, L. Schreiber, G. and Van de Velde, W. (eds) *A Future for Knowledge Acquisition Proceedings of the 8th European Knowledge Acquisition Workshop, EKAW'94,* Lecture Notes in Artificial Intelligence 867, Springer Verlag, Berlin, 256-276.

Shaw, M.L.G. and Woodward, J.B. (1989) Mental Models in the Knowledge Acquisition Process, In *Proceedings of Fourth Knowledge Acquisition for Knowledge-Based Systems Workshop,* Banff, October 1989.

Shiraz, G.M. and Sammut, C. (1998) Acquiring Control Knowledge from Examples Using Ripple-down ules and Machine learning, *11th Workshop on Knowledge Acquisition, Modeling and Management,* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:KAT-5.

Silverman, Barry G. (1992a) Judgment Error and Expert Critics in Forecasting Tasks. *Decision Sciences.* Sept/Oct 1992 23, 5: 1199-1219.

Silverman, Barry G. (1992b) Survey of Expert Critiquing Systems: Practical and Theoretical Frontiers. *Communications of the ACM.* April 1992 35, 4: 106.

Simmons, R. and Davis, R. (1993) The Roles of Knowledge and Representation in Problem Solving In David, J.M., Krivine, J.-P. and Simmons, R., (eds) *Second Generation Expert Systems***,** Springer, Berlin, 273-298.

Slater, P. (1977) *Dimensions of Intrapersonal Space: Volume 2* John Wiley.

Slowinski, R., Slowinski, K. and Stefanowski, J. (1988) Rough Set Approach to Analysis of Data from Peritoneal Lavage in Acute Pancreatitis *Medical Information,* 13:143-159.

Slowinski, Roman (1995) Rough Set Approach to Decision Analysis *AI Expert* March 1995.

Smith, D.C. (1977) *Pygmalian: A Computer Program to Model and Stimulate Creative Thought* Basel, Stuttgart.

Smith, D. C., Irby, C., Kimball, R., Verplank, W. and Harslem, E. (1982) The Star User Interface *Byte,* April 1982, 7(4):242-282.

Soloway, E, Bachant, J. and Jensen, K. (1987) Assessing the Maintainability of XCON-in-RIME: Coping with Problems of a very Large Rule Base *Proceedings of the Sixth International Conference on Artificial Intelligence*, Seattle, WA: Morgan Kaufman, Vol 2:824-829.

Sowa, J.F. (1984) *Conceptual Structures: Information Processing in Mind and Machine* Addison Wesley: Reading, Massachusetts.

Sowa, J.F. (1991) Toward the Expressive Power of Natural Language In John Sowa (ed) *Principles of Semantic Networks: Explorations in the Representation of Knowledge* Morgan Kaufmann Publishers Inc. California, 157-190.

Spangenberg, N and Wolff, K.E. (1988) Conceptual Grid Evaluation In H.H. Bock (ed) *Classification and Related Methods of Data Analysis* Elsevier Science Publishers B.V. North Holland.

Stachowitz, R. and Combs, J. (1987) Validation of Knowledge-Based Systems *In Proceedings 20th Annual Hawaii Internation Conference on System Sciences***,** Volume 1:686-695.

Stark, M. (1993) Impacts of Object-Oriented Technologies: Seven Years of Software Engineering 1993 *Systems Software* (23):163-169.

Steels, L. (1993) The Componential Framework and Its Role in Reusability In David, J.M., Krivine, J.-P. and Simmons, R. (eds), *Second Generation Expert Systems*, Springer, Berlin, 273-298.

Stelzner, M and Williams, M.D., (1988) The Evolution of Interface Requirements for Expert Systems in Hendler, J. A., *Expert Systems: The User Interface* Ablex Publishing Corp.

Stroulia, E. and Goel, A, Reflective, Self-Adaptive Problem Solvers (1994) In Steels, L. Schreiber, G. and Van de Velde, W. (eds) *A Future for Knowledge Acquisition Proceedings of the 8th European Knowledge Acquisition Workshop, EKAW'94,* Lecture Notes in Artificial Intelligence 867, Springer Verlag, Berlin, 394-413.

Strauss, A. (1978) *Negotiation: Varieties, Contexts, Processes and Social Order* Jossey-Bass Publishers, San Francisco, CA.

Stumme, G. (1997) Concept Exploration - A Tool for Creating and Exploring Conceptual Hierarchies In Dickson Lukose, Harry Delugach, Marry Keeler, Leroy Searle, and John F. Sowa, (Eds) (1997), *Conceptual Structures: Fulfilling Peirce's Dream, Proceedings of the Fifth International Conference on Conceptual Structures (ICCS'97),* August 3 - 8, University of Washington, Seattle, USA, Lecture Notes in Artificial Intelligence, Number 1257, Springer-Verlag, Berlin, 318-331.

Suchman, L. (1987) *Plans and Situated Actions* Cambridge UK: Cambridge University Press.

Suwa, M., Scott, A. and Shortliffe, E. (1982) An Approach to Verifying Completeness and Consistency in a Rule-Based Expert System *AI Magazine* 3(4):16-21.

Swartout, W. and Moore, J. (1993) Explanation in second generation ES In David, J.M., Krivine, J.-P. and Simmons, R., (eds), *Second Generation Expert Systems*, Springer, Berlin, 543-585.

Telles, M. (1990) Updating an Older Interface *In Conference Proceeding of CHI'90, Human Factors in Computing Systems* ACM, New York (April 1990), 183-191.

Teng, James T.C., Mirani, Rajesh and Sinha, Atish (1988) A Unified Architecture for Intelligent DSS. In *Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences.* The Computer Society of the IEEE, Washington.

Tesler, L. (1981) The Smalltalk Environment *Byte* August 1981, 90-147.

Thomas, K. (1976) Conflict and Conflict Management In Duneette (ed) *Handbook of Industrial and Organisational Psychology* Rand McNally College Publishing Co.

Thong (1997) http://www.cse.unsw.edu.au/~s2176432/index.html.

Torsten_Heycke (1994) URL:http://www-camis.stanford.edu/projects/htw.html.

Tracz, Will (1988) Software Reuse Maxims *ACM SIGSOFT Software Engineering Notes* 4(13):17-21.

Tsotsos, John K. (1982) Knowledge of the Visual Process*:* Content, Form and Use *IEEE 1982* 654-669.

Tu, S.W., Eriksson, H., Gennari, J.H., Shahar, Y. and Musen, M.A. (1994) Ontology-Based Configuration of Problem-Solving Mehtods and Generation of Knowledge-Acquisition Tools: Application of PROTEGE-II to Protocol-Based Decision Support *Knowledge Systems Laboratory*, Medical Computer Science, KSL-94-22, December 1994.

Tu, S. W. (1997) URL:http://camis.stanford.edu/projects/eon/index.html.

Tuhrim, S., J.A. Reggia and M. Floor, (1988) Expert System Development: Letting the Domain Specialist Directly Author Knowledge Bases, in Hendler, J. A., *Expert Systems: The User Interface* Ablex Publishing Corp.

Turban, Efraim (1993) *Decision Support and Expert Systems: Management Support Systems.* 4th edition, MacMillan Publishing Company, New York.

Turban, Efraim & Watkins, Paul R. (1986) Integrating Expert Systems and Decision Support Systems, Sprague, Ralph H., & Watson, Hugh J. (eds), *Decision Support Systems: Putting Theory into Practice.* Englewood Cliffs: Prentice-Hall Inc, 138-152.

Tyler, S. (1978) *The Said and the Unsaid: Mind, Meaning, and Culture* Academic Press, New York.

Van de Velde, W. (1993) Issues in Knowledge Level Modelling In David, J.M., Krivine, J.-P. and Simmons, R., editors *Second Generation Expert Systems*, Springer Verlag, Berlin, 211-231.

Van Heijst, G. Schreiber, A.T and Wielinga, B.J. (1997) Using Explicit Ontologies in KBS Development, *International Journal of Human-Computer Studies* 45:184-292.

Velasquez, Juan D. (1997) Modeling Emotions and Other Motivations in Synthetic Agents In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 10-15.

Vera, A.H and Simon, H.A (1993) Situated Action: A Symbolic Interpretation *Cognitive Science* 17:7-48.

Vere, S. A. (1980) Multilevel Counterfactuals for Generalizations of Relational Concepts and Productions *Artificial Intelligence* 14:139-164.

Vescovi, M., Iwasaki, Y., Fikes, R. and Chandrasekaran, B. (1994) CFRL: A Language for Specifying the Causal Functionality of Engineered Devices *Proceedings of the Eleventh National Conference on Artificial Intelligence* Washington, D.C. July 1994, 626-633.

Vogt, Frank and Wille, Rudolf (1995) TOSCANA - A Graphical Tool for Analyzing and Exploring Data *Knowledge Organisation* 22(2):78-81.

Wang, J.C., Boland, M., Graco, W. and He, H. (1996) Use of Ripple-Down Rules for Classifying Medical General Practitioner Practice Profiles Repetition In Compton, P., Mizoguchi, R., Motoda, H. and Menzies, T. (eds) *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia, 333-345.

Waterson, Andrew and Preece, Alun (1997) Knowledge Reuse and Knowledge Validation In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation of Knowledge-Based Systems Workshop,* July 28, 1997, Providence, Rhode Island, Technical Report WS-97-01, AAAI Press, Menlo Park, California, 33-39.

Webb, G. and Wells, J. (1995) Recent Progress in Machine-Expert Collaboration for Knowledge Acquisition In X. Yao (ed), *Proceedings of Eighth Australian Joint Conference on Artificial Intelligence AI'95***,** 13-17 November 1995, Canberra, World Scientific, Singapore, 91-298.

Webster, M (1995) The Relationship Between Models and Heuristics *BE Computer Engineering Honours Thesis,* University of New South Wales.

Weilinga, R., Van de Velde, W., Schreiber, G. and Akkermans, H., (1993) Towards a Unification of Knowledge Modelling Approaches In David, J.M., Krivine, J.-P. and Simmons, R., (eds) *Second Generation Expert Systems,* Springer, Berlin, 299-335.

Weintraub, M, (1991) An Explanation-Based Approach to Assigning Credit, *Phd Dissertation*, The Ohio State University.

Wenger, Etienne (1987) *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge* Mogan Kaufman, California.

White, B.Y. and Frederikson, J.R. (1989) Characterizing Knowledge Depth in Intelligent Safety Systems In Horn, Werner, *Causal AI Models* 1990 Hemisphere Publishing Corp., New York.

Wick, M.R. and Thompson, W.B. (1992) Reconstructive Expert System Explanation *Artificial Intelligence* 54(1-2), March 1992.

Wille, R. (1982) Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts In I. Rival (ed), *Ordered Sets* Reidel, Dordrecht, Boston, 445-470,.

Wille, R. (1989a) Lattices in Data Analysis: How to Draw them with a Computer In I. Rival (ed), *Algorithms and Order,* Kluwer, Dordrecht, Boston, 33-58.

Wille, R. (1989b) Knowledge Acquisition by Methods of Formal Concept Analysis In E. Diday (ed), *Data Analysis, Learning Symbolic and Numeric Knowledge* Nova Science Pub., New York, 365-380.

Wille, R. (1992) Concept Lattices and Conceptual Knowledge Systems *Computers Math. Applic. (23)* 6-9: 493-515.

*Wille, R. (1996) Conceptual Structures of Multicontexts In P.Eklund, G. Ellis and G. Mann (eds) *Conceptual Structures: Knowledge Representation as Interlingua) Proceedings of the 4th International Conference on Conceptual Structures*, LNAI 987, Springer. Verlag, **23-39.**

Winograd, T. And Flores, F. (1986)*Understanding Computers and Cognition: A New Foundation for Design* Ablex Norwood, NJ.

Wolf, Randy P. and Delaguch, Harry, S. (1997) PCCG: An Operational Tracked Grid for Creating Conceptual Graphs In Dickson Lukose, Harry Delugach, Marry Keeler, Leroy Searle, and John F. Sowa, (Eds) (1997), *Conceptual Structures: Fulfilling Peirce's Dream, Proceedings of the Fifth International Conference on Conceptual Structures (ICCS'97),* August 3 - 8, University of Washington, Seattle, USA, Lecture Notes in Artificial Intelligence, Number 1257, Springer Verlag, Berlin, 617-620.

Woods, D.D. (1987) Commentary: Cognitive Engineering in Complex and Dynamic Worlds *International Journal of Man-Machine Studies* 27:571-585.

Woods, W.A. (1991) Understanding Subsumption and Taxonomy: A Framework for Progress In John Sowa (ed) *Principles of Semantic Networks: Explorations in the Representation of Knowledge* Morgan Kaufmann Publishers, Inc. California, 45-94.

Woods, W. A. and Schmolze, J.G. (1991) The KL-ONE *Computers and Mathematics with Applications, - Special Issue on Semantic Networks in Artificial Intelligence*.

Woodward J.B and Shaw, M.L.G., (1993) Cognition Support Tools for Knowledge Acquisition in *Proceedings of the 8th Knowledge Acquisition for Knowledge-Based Systems Workshop*. Banff, Canada: 13.1-13.20.

Young, Lawrence F. (1989) *Decision Support and Idea Processing Systems.* Iowa: Win C. Brown Publishers.

Zachary, W.W. (1988) Decision Support Systems: Designing to Extend the Cognitive Limits In Helender, M. (ed) (1988) *Handbook of Human-Computer Interaction*, Elsevier Science Publishers B.V. North-Holland, 997-1030.

Zacklad, M. and Fontaine, D. (1993) Systematic Building of Conceptual Classification Systems with C-KAT in *Knowledge Acquisition for Knowledge-Based Systems,7th European Workshop, EKAW '93*, Toulouse and Calylus, France, September 6-10, 1993, 79-102.

Zalta, E.N. (1988) *Intensional Logic and the Metaphysics of Intentionality*, Cambridge, Massachusetts, MIT Press.

Zdrahal, Z and Motta, E. (1995) An In-Depth Analysis of Propose and Revise Problem Solving Methods In *9th Knowledge Acquisition for Knowledge Based Systems Workshop* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, 38.1-38.20.

Zeitz, C.M. (1997) Some Concrete Advantages of Abstraction: How Experts' Representations Facilitate Reasoning In Feltovich, Paul J., Ford, Kenneth, M. and Hoffman, Robert R. (eds) *Expertise in Context: Human and Machine* AAAI Press/MIT Press, Cambridge, Massachusetts, 43-65.

Zeng, Dajun and Sycara, Katia (1997) Benefits of Learning in Negotiation In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Application of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 36-41.