Automated Assessment in the Internet Classroom

Dave Barker-Plummer
CSLI, Stanford University
Stanford, California, 94305, USA

Robert Dale
Centre for Language Technology, Macquarie University
Sydney, NSW 2109, Australia

Richard Cox
Department of Informatics, University of Sussex
Falmer, E. Sussex, BN1 9QJ, UK

John Etchemendy
CSLI and Philosophy, Stanford University
Stanford, California, 94305, USA

Abstract

A likely feature of the Internet classroom is automated assessment of student exercises. In this paper, we describe the design and implementation of the Grade Grinder, an Internet-based assessment service developed for use with logic-teaching courseware. We discuss the utility of this platform both as a pedagogical resource, and as a provider of data for research on student problem-solving. We end by making some observations in regard to the scope for extending this approach beyond the domain of logic-teaching to other domains.

Introduction

The Grade Grinder is an innovative Internet-based assessment service for logic education. Access to the Grade Grinder is available to users of the *Language, Proof and Logic* courseware package for teaching formal logic to undergraduates (Barwise et al. 1999). These packages each consist of a textbook, some desktop applications used to complete exercises, and access to the Grade Grinder. Students receive reports on their submitted work by email, usually within minutes. The Grade Grinder has been in continuous service for more than eight years, and has been used by more than 38,000 students at more than a hundred institutions in at least a dozen countries.

For students, the Grade Grinder serves as a teaching assistant that is available twenty-four hours a day seven days a week. The feedback provided by the Grade Grinder allows students to work to correct their errors on their own and thereby facilitates self-directed learning. Once a student has perfected her work, or her deadline arrives, the Grade Grinder can be instructed to copy the grade report to her instructor. Thus, the Grade Grinder provides formative assessment that is individualised, on-demand, immediate, and motivational (Ridgway, McCusker, and Pead 2004).

For instructors, the Grade Grinder facilitates the focus of resources on those students who are unable to work to the correct result on their own, and who are likely to have deeper conceptual problems, rather than minor technical ones. The instructor may use the results of student submissions for summative assessment (e.g., to assign grades), or to focus instruction on topics that are causing difficulty.

For researchers, the Grade Grinder has accumulated a very large corpus of work produced by students learning logic. This database has high ecological validity and can be exploited in order to gain insights into the cognitive processes in play during formal reasoning (such as those associated with natural-to-formal language translation and interpretation), to extend research into individual differences in reasoning, to improve logic instruction, and of course to improve the performance and design of the courseware, including Grade Grinder itself.

In this paper we describe the design and implementation of the Grade Grinder service. We discuss possibilities for generalising the Grade Grinder for assessment in other subject areas and outline our recent research data mining the corpus of submissions (over 1.8 million of them) to discover patterns in student performance using the courseware.

Design and Implementation

In designing the Grade Grinder, we have taken the view that the software should be able to assess complex exercises typically set in a logic courses, rather than requiring us to tailor the exercises to those which are simple to grade automatically. For example, there are no "multiple choice" exercises in either of the courseware packages. In fact, the Grade Grinder allows the creation of exercises which might otherwise be too complicated for instructors to set. Some of the exercises have infinitely many correct answers, and even skilled logicians might have difficulty determining whether or not a student's answer is correct. By implementing complex algorithms within the Grade Grinder (for proving logical equivalence of answers) and providing automated assessment, we have expanded, rather than limited, the range of exercises that can realistically be assigned to students.

The Grade Grinder can assess a wide range of tasks specific to the subject domain of logic. These include the construction of proofs, the construction of truth tables, the trans-

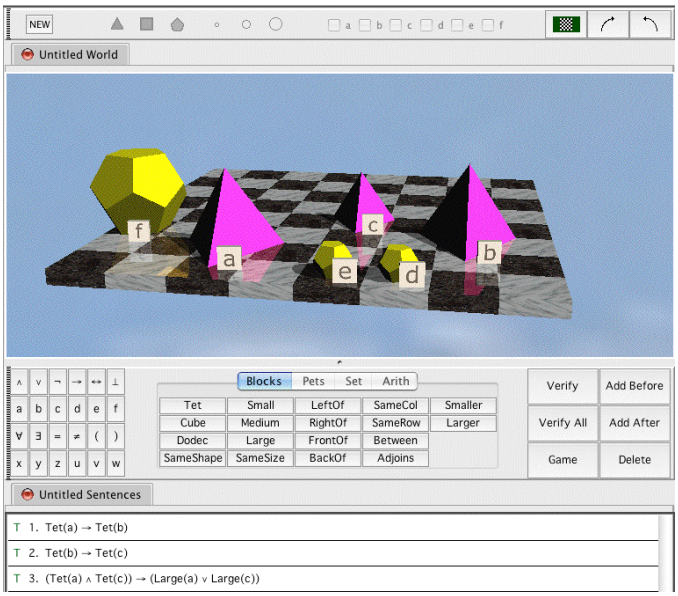

Figure 1: Tarski's World

lation of sentences from natural language into formal logic, the construction of "worlds" that make a collection of sentences true, and so forth. Some exercises require the simultaneous solution of more than one of these tasks, and others require the completion of *some* task but the student is not initially told which (for example, a student may be asked to either complete a proof of, or build a counterexample to, some claim.)

As a concrete example, the student might be asked to translate a natural language sentence such as *every dodecahedron is in front of a cube*, into first-order logic. Using our Tarski's World application (Figure 1), they can enter their translation and then build and examine a variety of configurations of blocks, tracking the truth of their formal sentence in the different configurations.

To accommodate the flexibility required to set these exercises, the Grade Grinder is designed in a modular fashion. Internally, there are a number of *Grader Units*, each of which is responsible for assessing the completion of a particular kind of task. A database of *Grading Instructions* records which Grader Units are required to assess which exercises. The domain knowledge lies in the Grader Units, while knowledge of the exercises written in the textbooks lies in the database. This architecture makes it straightforward to modify the Grade Grinder to assess work in other domains (since new Grader Units could be implemented and included in the running system), and to create new exercises using the same Grader Units.

Instructor Features

Several features are included in the Grade Grinder aimed specifically at instructors. When an exercise is submitted to an instructor the grade report email is copied to them. The report and supporting files are also collected on our web server. The instructor can log on to the web server and access the complete set of reports for her students, segmented by time period if desired. The instructor may view each of the submitted files within the web page, or download a copy to their local machine if required. Even without the assessment portion of the service, the Grade Grinder performs the useful function of collecting and organising student submissions freeing the instructor from collecting disks, managing drop boxes, losing student work, and opening individual files by hand.

One concern often expressed by instructors about student use of computers in their work is the ease with which work can be shared by students. Our courseware packages address this issue by including pseudo-unique keys in the files created by our desktop applications. These keys do not depend on the content of the file (two identical solutions derived independently are almost guaranteed to have different keys) but are based on the times at which the file has been modified. When the file is submitted, the Grade Grinder checks the key in the file against all others it has ever received, and includes a report of "collisions" in the grade report. If the keys are identical, then two (or more) students have submitted the very same file. If two files share a common prefix, then a common source file was shared and then independently modified to create the two files.

Data Mining Research

In (Barwise and Etchemendy 1998) the authors describe how developing educational software informs their understanding of the subject matter and their future research program. We are actively pursuing further new research avenues made possible by the development of the Grade Grinder.

In (Barker-Plummer et al. 2008; Cox et al. 2008) we present research results based on data mining of the student submissions to the Grade Grinder. A principle aim of this work is to characterise and taxonomise student errors and develop cognitive (information processing) accounts of their origin. Another aim is to improve the Grade Grinder's capabilities with respect to error diagnosis and feedback; this might be achieved via the addition of learner modelling. The Grade Grinder's feedback might be enriched through the use of automatic natural language paraphrasing so that the system is able to select and formulate an appropriate natural language expression that makes clear the difference between a student's incorrect answer and the correct one.

In (Barker-Plummer et al. 2008), we present empirical evidence for specific characteristics of natural language problem statements that frequently lead to students making mistakes. We developed a rich taxonomy of the types of errors that students make, and implemented tools for automatically classifying student errors into these categories. In that paper, we focus on three specific phenomena that were prevalent in our data: Students were found (a) to have particular difficulties with distinguishing the conditional from the biconditional, (b) to be sensitive to word-order effects during translation, and (c) to be sensitive to factors associated with the naming of constants. This information is of significant value in directing subsequent revisions to the exercises and in determining the form and content of new exercises.

In (Cox et al. 2008), we report some initial results that demonstrate that when we look at how students construct diagrammatic representations of information expressed in natural language sentences, this reveals misunderstandings that would not otherwise be apparent. In particular, constructing a diagram requires the student to provide an instantiated representation of the meaning of a natural language sentence, testing their understanding in a way that translation into first-order logic does not.

Learner Modeling and Visible Learning

Our current activities involve developing metadata coding methods for marking-up exercises and students' solutions. Dimensions we are attempting to represent in the scheme include: task difficulty and cognitive load (assessed in terms of the number of blocks world entities involved in an exercise, the types of predicates and connectives used, and the linguistic complexity of the natural language sentences to be interpreted); graphical agency (does the exercise require the student to check solutions against a prefabricated blocks world, or is she asked to construct her own world from scratch?); and the degree of between-modality translation (natural language to first-order logic and/or to diagram; natural language to diagram and/or to first-order logic, and so on). Our intent is that these codings, together with the student error analysis findings, will provide a basis for the hierarchical decomposition of the *Language, Proof and Logic* curriculum, in terms of conceptual knowledge pre-requisites and co-requisites. This could be machine-represented, perhaps as a Bayesian Belief Network (BBN) e.g. (Zapata-Rivera and Greer 2004; Grawemeyer and Cox 2005); the BBN would, in turn, provide a model against which an individual's learning profile could be 'traced' e.g. (Koedinger and Anderson 1999).

A model-tracing approach would support improved error diagnosis and feedback by the Grade Grinder and may also permit advice-giving regarding prospective pitfalls. So, for example, we would be in a position to automatically generate advice like the following:

> 78% of students who, like you, make errors associated with confounding the biconditional with the conditional in the chapter 7 exercises, later tend to have difficulties in Chapter 8 on the topic of .... It is suggested that you pay particular attention to ....

The Grade Grinder can identify sequences of attempts by a single student to solve an exercise or sequence of exercises. Hence a further extension of the Grade Grinder would allow students to inspect the system's model of their learning history in order to encourage metacognitive processes such as reflection and self-explanation. We may also be able to provide students with a facility by which they can compare their own error profiles to the aggregated profiles of their peers. This approach would combine approaches from research on inspectable (individual) student models (Bull, Pain, and Brna 1995; Zapata-Rivera and Greer 2004) with modelling the performance of groups of students (Jameson, Baldes, and Kleinbauer 2003; Suebnukarn and Haddawy 2004).

Deployment Experience

The Grade Grinder has been in continuous operation for nine years, and has been used by more than 38,000 students at over a hundred institutions in at least a dozen countries.

In order to protect against network service outages, the Grade Grinder was designed from day one to employ redundant servers. There are currently two, located at Stanford University and the University of Chicago. Data concerning the submissions received are consolidated to a central server which is used for delivery of web-based services.

The adoption of email for the delivery of grade reports has the advantage of simplicity, but email protocols do not provide guaranteed timely delivery, or timely notification of delivery failure. In addition email delivery has increasingly provided some challenges as Internet service providers strive to meet their users' demands for spam blacklists. Additionally, hyperactive spam filters can also result in apparent non-delivery of grade reports.

Our decision to implement desktop applications for the students to complete their work (as opposed to a web-based environment, for example) has significant consequences. We support our applications on the Macintosh, Windows and Linux platforms. We adopted the Java programming language for implementation, which is advertised as a "write-once, run-anywhere" technology. Even so we have had to deal with issues concerning the user experience on all platforms, continual upgrading of the underlying operating systems, and the underlying Java virtual machines. Significant resources are required to implement and maintain desktop applications to run on multiple platforms and locales.

Generalisation to Other Domains

The Grade Grinder was designed to provide assessment to introductory undergraduate logic students. A large fraction of the implementation effort has been focussed on the general infrastructure needed to provide this service, and a relatively small portion of the Grade Grinder code base is concerned with the specific subject matter. We believe that the Grade Grinder architecture supports easy extension to provide assessment services for other education domains. The main problems in developing such an extension are the specification of the exercises to be answered by the students, and the specification of the form of answers to these exercises.

In the current situation, the first of these problems is solved by configuring the Grade Grinder to assess those exercises that appear in our own textbooks. We have taken steps to permit the addition of instructor-designed exercises to the Grade Grinder database, and expect to field a version of the system that permits this fairly soon.

The second of these problems is solved by requiring that the students use our desktop software to solve the exercises, and to submit the resulting saved files. The Grade Grinder can be extended to open files in any well-specified file-format, but the structure of the information within the files must be understandable as a solution to some problem.

In the simplest example, the Grade Grinder might be used to assess essays written by students. In this case the submitted files could be constructed in plain or rich text formats,

for example. The challenge would be the provision of algorithms to assess the quality of the solution; this is already a focus of activity elsewhere, for example see (Kakkonen et al. 2005). At the other extreme, the Grade Grinder might be extended to open files constructed by applications like the Geometer's Sketchpad or Mathematica. Assessment might then be simplified by the highly structured form of the input, although many of the capabilities of the desktop applications may need to be duplicated in the Grade Grinder.

The International Dimension

The standard process for internationalization involves localizing the application at runtime to the locale of the desktop on which the application is being run. The difficulty with internationalizing server software is that the locale of the submitting student must be used to localize their grade report.

Another internationalization issue applies specifically to the domain of logic education, since the understanding of how to translate natural language sentences into formal language is dependent on the natural language that serves as the source language. *Language, Proof and Logic* has itself been translated into both German and Japanese. In the case of the Japanese translation, the exercise sentences appear in both English and (parenthetically) Japanese. On the other hand, the German translator opted to translate the exercises entire. In neither case has the desktop software or the Grade Grinder been localized, in particular the target language of first-order logic uses the atom Small(a) to indicate that a is the name of a small block, even if the exercise contains the corresponding German term "klein". The translators for the proposed Czech translation of *Language, Proof and Logic* believe that localizing the software for their users is critical to the success of their translation, and they plan additionally to localize the software. This will have the effect of changing the target logical language, and making the determination of whether an answer is correct dependent on the locale of the user.

Conclusion

To achieve the accessibility and utility required in the Internet classroom, it is almost certainly the case that automatic assessment mechanisms will be required. Although the exact nature of these will depend on the specific domain of assessment, the specific mechanisms will also need a more general infrastructure for managing the automated assessment process. In this paper we have described the Grade Grinder, a tried-and-tested solution to this problem that has been used for over eight years; we believe this same model can be extended straightforwardly to other educational domains.

We have also drawn attention here to the scope that such mechanisms provide for data gathering, whose analysis can feed back into both the development of exercises and other teaching materials, and into the ongoing improvement of the assessment tools themselves.

References

Barker-Plummer, D.; Cox, R.; Dale, R.; and Etchemendy, J. 2008. An empirical study of errors in translating natural language into logic. In *Proceedings of the 30th Annual Cognitive Science Society Conference*. To be held in Washington, DC, July.

Barwise, J., and Etchemendy, J. 1998. Computers, visualization and the nature of reasoning. In Bynum, T., and Moor, J. H., eds., *The Digital Phoenix: How Computers are Changing Philosophy*. Blackwell. 93–116.

Barwise, J.; Etchemendy, J.; Allwein, G.; Barker-Plummer, D.; and Liu, A. 1999. *Language, Proof and Logic*. CSLI Publications and University of Chicago Press.

Bull, S.; Pain, H.; and Brna, P. 1995. Mr collins: Student modelling in intelligent computer assisted language learning. *Instructional Science* 23:65–87.

Cox, R.; Dale, R.; Etchemendy, J.; and Barker-Plummer, D. 2008. Graphical revelations: Comparing students' translation errors in graphics and logic. In *Diagrams 2008, Fifth International Conference on the Theory and Application of Diagrams*. To be held in Herrsching, Germany, September 2008.

Grawemeyer, B., and Cox, R. 2005. Developing a bayesnet based student model for an external representation selection tutor. In Looi, C.-K.; McCalla, G.; Bredeweg, B.; and Breuker, J., eds., *Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology. Frontiers in Artificial Intelligence and Applications*, volume 125. IOS Press. 810–812.

Jameson, A.; Baldes, S.; and Kleinbauer, T. 2003. Generative models of group members as support for group collaboration. In Gaudioso, E., ed., *Workshop on User and Group Models for Web-based Adaptive Collaborative Environments, Proceedings of the Ninth International Conference on User Modeling*, 114.

Kakkonen, T.; Myller, N.; Timonen, J.; and Sutinen, E. 2005. Automatic essay grading with probabilistic latent semantic analysis. In *Proceedings of the Second Workshop on Building Educational Applications Using NLP*, 29–36. Ann Arbor, Michigan: Association for Computational Linguistics.

Koedinger, K., and Anderson, J. 1999. Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education* 8:30–43.

Ridgway, J.; McCusker, S.; and Pead, D. 2004. Literature review of e-assessment. *Bristol: Futurelab.*

Suebnukarn, S., and Haddawy, P. 2004. A collaborative intelligent tutoring system for medical problem-based learning. In Nunes, N., and Rich, C., eds., *Proceedings of the Ninth International Conference on Intelligent User Interface.*, 1421.

Zapata-Rivera, J.-D., and Greer, J. 2004. Interacting with inspectable bayesian student models. *International Journal of Artificial Intelligence in Education* 14:1–37.