

This Phrase-Based SMT System is Out of Order: Generalised Word Reordering in Machine Translation

Simon Zwarts

Centre for Language Technology
Macquarie University
Sydney, Australia
szwarts@ics.mq.edu.au

Mark Dras

Centre for Language Technology
Macquarie University
Sydney, Australia
madras@ics.mq.edu.au

Abstract

Many natural language processes have some degree of preprocessing of data: tokenisation, stemming and so on. In the domain of Statistical Machine Translation it has been shown that word reordering as a preprocessing step can help the translation process.

Recently, hand-written rules for reordering in German–English translation have shown good results, but this is clearly a labour-intensive and language pair-specific approach. Two possible sources of the observed improvement are that (1) the reordering explicitly matches the syntax of the source language more closely to that of the target language, or that (2) it fits the data better to the mechanisms of phrasal SMT; but it is not clear which. In this paper, we apply a general principle based on dependency distance minimisation to produce reorderings. Our language-independent approach achieves half of the improvement of a reimplementation of the hand-crafted approach, and suggests that reason (2) is a possible explanation for why that reordering approach works.

Help you I can, yes.
Jedi Master Yoda

1 Introduction

Preprocessing is an essential step in Natural Language applications. Reordering of words on a sentence level as a more extensive step for preprocessing has succeeded in improving results in Statistical Machine Translation (SMT). Here, both in the training and in the decoding phase, sentences in the source language are reordered before being processed.

This reordering can be done based on rules over word alignment learnt statistically; (Costa-Juss and Donollosa, 2006), for example, describe such a system. In this work an improvement in overall translation quality in a Spanish-English MT

system was achieved by using statistical word classes and a word-based distortion model to reorder words in the source language. Reordering here is purely a statistical process and no syntactical knowledge of the language is used.

Xia and McCord (2004) do use syntactical knowledge; they use pattern learning in their reordering system. In their work in the training phase they parse and align sentences and derive reordering patterns. From the English-French Canadian Hansard they extract 56,000 different transformations for translation. In the decoding phase they use these transformations on the source language. The main focus then is monotonic decoding (that is, decoding while roughly keeping the same order in the target language as in the source language — reordering done within phrases, for example, is an exception).

Syntactically motivated rules are also used in reordering models. In Collins et al. (2005) six hand-written rules for reordering source sentences are defined. These rules operate on the output of an automatic parser. The reordering rules however are language-pair (German-English) specific and hand-written.

We want to extend this idea of word reordering as preprocessing by investigating whether we can find a general underlying principle for reordering, to avoid either thousands of patterns, or arguably arbitrary hand-written rules to do this. To do this, we note that a common characteristic of the Collins et al. (2005) rules is that they reduce the distances of a certain class of long-distance

dependencies in German with respect to English. We note that minimising of dependency distances is a general principle appearing in a number of guises in psycholinguistics, for example the work of Hawkins (1990). In this paper we exploit this idea to develop one general syntactically motivated reordering rule subsuming those of Collins et al. (2005).

This approach also helps us to tease out the source of translation improvement: whether it is because the reordering matches more closely the syntax of the target language, or because fits the data better to the mechanisms of phrasal SMT.

The article is structured as follows: in section 2 we give some background on previous work of word reordering as preprocessing, on general word ordering in languages and on how we can combine those. In section 3 we describe the general rule for reordering and our algorithm. Section 4 describes our experimental setup and section 5 presents the results of our idea; this leads to discussion in section 6 and a conclusion in section 7.

2 Reordering Motivation

It has been shown that reordering as a preprocessing step can lead to improved results. In this section we look in a little more detail at an existing reordering algorithm. We then look at some general characteristics in word ordering in the field of psycholinguistics and propose an idea for using that information for word reordering.

2.1 Clause Restructuring

Collins et al. (2005) describe reordering based on a dependency parse of the source sentence. In their approach they have defined six hand-written rules for reordering German sentences. In brief, German sentences often have the tensed verb in second position; infinitives, participles and separable verb particles occur at the end of the sentence. These six reordering rules are applied sequentially to the German sentence, which is their source language. Three of their rules reorder verbs in the German language, and one rule reorders verb particles. The other two rules reorder the subject and put the German word used in negation in a more

English position. All their rules are designed to match English word ordering as much as possible. Their approach shows that adding knowledge about syntactic structure can significantly improve the performance of an existing state-of-the-art statistical machine translation system.

2.2 Word Order Tendencies in Languages

In the field of psycholinguistics Hawkins (1990) argues that the word order in languages is based on certain rules, imposed by the human parsing mechanism. Therefore languages tend to favour some word orderings over others. He uses this to explain, for example, the universal occurrence of postposed sentential direct objects in Verb-Object-Subject (VOS) languages.

In his work, he argues that one of the main reasons for having certain word orders is that we as humans try to minimise certain distances between words, so that the sentence is easier to process. In particular the distance between a head and its dependents is important. An example of this is the English Heavy Noun Phrase shift. Take the following two sentence variants:

1. I give <NP> back
2. I give back <NP>

Whether sentence 1 or 2 is favourable, or even acceptable, depends on the size (heaviness) of the NP. If the NP is *it* only 1 is acceptable. When the NP is medium-sized, like *the book*, both are fine, but the longer the NP gets the more favourable 2 gets, until native speakers will say 1 is not acceptable anymore. Hawkins explains this by using head-dependent distances. In this example *give* is the head in the sentence; if the NP is short, both the NP and *back* are closely positioned to the head. The longer the NP gets the further away *back* is pushed. The theory is that languages tend to minimise the distance, so if the NP gets too long, we prefer 2 over 1, because we want to have *back* close to its head *give*.

2.3 Reordering Based on Minimising Dependency Distances

Regarding the work of Collins et al., we suggest two possible sources for the improvement obtained.

Match target language word order Although most decoders are capable of generating words in a different order than the source language, usually only simple models are used for this reordering. In Pharaoh (Koehn, 2004), for example, every word reordering between languages is penalised and only the language model can encourage a different order. If we can match the word order of the target language to a certain degree, we might expect an increase in translation quality, because we already have more explicitly used information of what the new word ordering should be.

Fitting phrase length The achievement of Phrase-Based SMT (PSMT) (Koehn et al., 2003) was to combine different words into one phrase and treat them as one unit. Yet PSMT only manages to do this if the words all fit together in the same phrase-window. If in a language a pair of words having a dependency relation are further apart, PSMT fails to pick this up: for example, verb particles in German which are distant from the governing verb. If we can identify these long distance dependencies and move these words together into the span of one phrase, PSMT can actually pick up on this and treat it as one unit. This means also that sentences not reordered can have a better translation, because the phrases present in that sentence might have been seen (more) before.

The idea in this paper is to reorder based on a general principle of bringing dependencies closer to their heads. If this approach works, in not explicitly matching the word order of the target language, it suggests that fitting the phrase window is a contributor to the improvement shown by reordering. The approach also has the following attractive properties.

Generalisation To our knowledge previous reordering algorithms are not capable of reordering based on a general rule (unlike ‘arbitrary’ hand written language-pair specific rules (Collins et al., 2005) or thousands of different transformations (Xia and McCord, 2004)). If one is able to show that one general syntactically informed rule can lead to translation quality this is evidence in favour of the theory used explaining how languages themselves operate.

Explicitly using syntactic knowledge Although in the Machine Translation (MT) community it is still a controversial point, syntactical information of languages seems to be able to help in MT when applied correctly. Another example of this is the work of Quirk et al. (2005) where a dependency parser was used to learn certain translation phrases, in their work on “treelets”. When we can show that reordering based on an elegant rule using syntactical language information can enhance translation quality, it is another small piece of evidence supporting the idea that syntactical information is useful for MT.

Starting point in search space Finally, most (P)SMT approaches are based on a huge search space which cannot be fully investigated. Usually hill climbing techniques are used to handle this large search space. Since hill climbing does not guarantee reaching global minima (error) or maxima (probability scoring) but rather probably gets ‘stuck’ in a local optimum, it is important to find a good starting point. Picking different starting points in the search space, by preprocessing the source language, in a way that fits the phrasal MT, can have an impact on overall quality.¹

3 Minimal Dependency Reordering

Hawkins (1990) uses the distance in dependency relations to explain why certain word orderings are more favourable than others. If we want to make use of this information we need to define what these dependencies are and how we will reorder based on this information.

3.1 The Basic Algorithm

As in Collins et al. (2005), the reordering algorithm takes a dependency tree of the source sentence as input. For every node in this tree the linear distance, counted in tokens, between the node and its parent is stored. The distance for a node is defined as the closest distance to the head of that node or its children.

¹This idea was suggested by Daniel Marcu in his invited talk at ACL2006, *Argmax Search in Natural Language Processing*, where he argues the importance of selecting a favourable starting point for search problems like these.

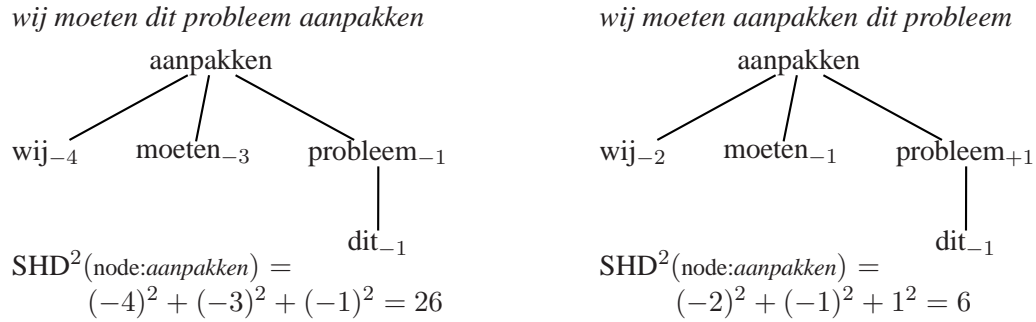


Figure 1: Reordering based on the sum of the head distances

To illustrate the algorithm of this section we present the following two examples:

1. Verb initial in VP:

normal order:

wij moeten dit probleem aanpakken
we must this problem tackle

reordered:

wij moeten aanpakken dit probleem
we must tackle this problem

reference:

we must tackle this particular problem

2. Verb Particle reordering:

normal order:

het parlement neemt de resolutie aan
the parliament takes the resolution over

reordered:

het parlement neemt aan de resolutie
the parliament takes over the resolution

reference:

parliament adopted the resolution

As an example of the calculation of distances, the left tree of Figure 1 is the dependency tree for the normal order for example 1; nodes are annotated with the distance from the word to its governor. Note that in example 1 *probleem* gets a value of 1, although the word itself is 2 away from its head; we are measuring the distance from this complete constituent and not this particular word.

Based on the distance of the different child nodes we want to define an optimal reordering and pick that one. This means we have to score all the different reorderings. We want a scoring measure to do this that ignores the sign of distances

and gives higher weight to longer dependency distances. Thus, similar to various statistical optimisation algorithms such as Least Mean Squares, we calculate the square of the Sum of the Head Distances (SHD²) for each node n , defined as:

$$\text{SHD}^2(n) = \sum_{c \in \text{children}(n)} \text{Distance}(c, n)^2$$

Every different ordering of children and head has a SHD² value; we are interested in minimising this value. We give an SHD² example in Figure 1.

We then reorder the children so that the SHD² score of a node is minimised. The righthand tree of Figure 1 gives an example. In example 1 we can see how the Dutch verb *aanpakken* is moved to the beginning of the verb phrase. In this example we match English word order, even though this is not an explicit goal of the metric. The second example does not match English word order as such, but in Dutch the verb *aannemen* was split into *aan* and *neemt* in the sentence. Our reordering places these two words together so that PSMT can pickup that this is actually one single unit. Note that the two examples also demonstrate two of Collins et al. (2005) hand-written rules. In fact, this principle subsumes all the examples given in that paper in the prototypical cases.

3.2 Selecting Minimal Reorderings

In implementing the algorithm, for each node with children we calculate the SHD² for all permutations (note that this is not computationally expensive as each node has only a small number of children). We select the collection of sibling orders with a minimal SHD². This is indeed a collection because different orderings can still have the same

SHD² value. In these cases we try to match the original sentence order much as possible.

There are many different way to calculate which permutation’s order is closer to the original. We first count all the constituents which are on the other side of the head compared to the original. For all permutations from the list with the same minimal SHD² we count these jumps and keep the collection with the minimal number of jumps. Then we count the breaks where the new order deviates from an ascending ordering, when the constituents are labelled with their original position. For every constituent ordering c this Break Count (BC) is defined as:

$$BC(c) = \sum_{n=2}^N \max(0, Pos_o(n-1) - Pos_o(n) + 1)$$

Here $Pos_o(x)$ is the original position of the x th constituent in the new order, and N is the length of the constituent c . As an example: if we have the five constituents 1, 2, 3, 4, 5 which in a new order y are ordered 2, 1, 3, 4, 5 we have one break from monotonicity, where from 2 we go to 1. We add the number of breaks to the size of the break. In this case, $BC(y) = 2$. The original constituent order always gets value 0. From the remaining collection of orderings we can now select that one with the lowest value. This always results in a unique ordering.

3.3 Source Language Parser

To derive the dependency trees we used the Alpino (Bouma et al., 2000) parser.² Because this grammar comes with dependency information we closely follow their definition of head-dependent relations, deviating from this in only one respect. The Alpino parser marks auxiliary verbs as being the head of a complete sentence, while we took the main verb as the head of sentence, transforming the parse trees accordingly (thus moving closer to semantic dependencies).

The Alpino parser does not always produce projective parses. Reading off parse trees of Alpino in some cases already changes the word order.

²We would like to thank van Noord from the University of Groningen for kindly providing us the parses made by Alpino for most of the Dutch sections for the relevant data.

3.4 Four Reordering Models

We investigate four models.

The first, the ‘Alpino model’, is to measure the impact of the parser used, as Alpino does some reordering that is intended to be linguistically helpful. We want to know what the result is of using a dependency parser which does not generate projective parses i.e. there are parses which do not result into the original sentence if we read off the tree for this parse. In this model we parse the sentences with our parser, and we simply read off the tree. If the tree is projective this results in the original tree. If this is not the case we keep for every node the original order of its children.

The second, the ‘full model’, chooses the reordering as described in sections 3.1 and 3.2. We hypothesised the algorithm may be too ‘enthusiastic’ in reordering. For example, when we encounter a ditransitive verb the algorithm usually would put either the direct or the indirect object in front of the subject. Longer constituents were moved to completely different positions in the sentence. This kind of reordering could be problematic for languages, like English, which heavily rely on sentence position to mark the different grammatical functions of the constituents.

We therefore defined the ‘limited model’, a restriction on the previous model where only single tokens can be reordered. When analysing previous syntactically motivated reordering (Collins et al., 2005) we realised that in most cases constituents consisting of one token only were repositioned in the sentence. Furthermore since sentence ordering is so important we decided only to reorder if ‘substantial’ parts were changed. To do this we introduced a threshold R and only accepted a new ordering if the new SHD² has a reduction of at least R in regard to the original sentence ordering. If it is not possible to reduce SHD² that far we would keep the original ordering. Varying R between 0 and 1, in this preliminary work we determined the value $R = 0.9$.

Finally we reimplemented the six rules in Collins et al. (2005) as closely as possible given our language pair and our parser, the Alpino parser. The ‘Collins Model’ will show us the im-

	$n = 1$	$n = 2$	$n = 3$	$n = 4$	BP ^a	BLEU
Baseline	0.519	0.269	0.155	0.0914	0.981	0.207
Alpino	0.515	0.264	0.149	0.085	0.972	0.198
Full	0.518	0.262	0.146	0.083	0.973	0.196
Limited	0.521	0.271	0.155	0.0901	0.964	0.203
Collins	0.521	0.276	0.161	0.0966	0.958	0.208

^abrevity penalty of BLEU

Table 1: Automatic Evaluation Metrics for the different Models

pact of our parser and our choice of language pair.

4 Experimental Setup

For our experiments we used the decoder Pharaoh (Koehn, 2004). For the phrase extraction we used our implementation of the algorithm which is described in the manual of Pharaoh. As a language model we used the SRILM (Stolcke, 2002) toolkit. We used a trigram model with interpolated Kneser-Ney discounting.

For a baseline we used the Pharaoh translation made with a normal GIZA++ (Och and Ney, 2003) training on unchanged text, and the same phrase extractor we used for our other four models.

As an automated scoring metric we used the BLEU (Papineni et al., 2002) and the F-Measure (Turian et al., 2003)³ method.

For our training data we used the Dutch and English portions of most of the Europarl Corpus (Koehn, 2003). Because one section of the Europarl corpus was not available in a parsed form, this was left out. After sentence aligning the Dutch and the English part we divided the corpus into a training and a testing part. From the original available Dutch parses we selected every 200th sentence for testing, until we had 1500 sentences. We have a little over half a million sentences in our training section.

³In this article, we used our own implementations of the BLEU and the F-Measure score available from <http://www.ics.mq.edu.au/~szwarts/Downloads.php>

5 Results

For the three models and the baseline, results are given in Table 1. The first interesting result is the impact of the parser used. In the Europarl corpus 29% of the sentences have a different word order when just reading off the Alpino parse compared to the original word order. It turns out that our results for the Alpino model do not improve on the baseline.

In the original Collins et al. work, the improvement over the baseline was from 0.252 to 0.268 (BLEU) which was statistically significant. Here, the starting point for the Collins reordering is the read-off from the Alpino tree; the appropriate baseline for measuring the improvement made by the Collins reordering is thus the Alpino model, and the Collins model improvement is (a comparable) 0.01 BLEU point.

The Full Reordering model in fact does worse than the Alpino model. However, in our Limited Reordering model, our scores show a limited improvement in both BLEU and F-Measure above the Alpino model score.

In this model only half of the sentences (49%) are reordered compared to the original source sentences. But as mentioned in section 2 not-reordered sentences can also be translated differently because we hope to have a better phrase table. When we compare sentence orders from this model against the sentence ordering from the direct read-off from the Alpino parser 46% of the sentences have a different order, so our method does much more than changing the 29% changed sentences of the Alpino read-off up to 49%.

In Table 2 we present some examples where we actually produce better translations than the baseline,

Limited Reordering success		
1	B	the democratisation process is web of launched
	L	the democratisation process has been under way
	R	the process of democratisation has only just begun
2	B	the cap should be revised for farmers to support
	L	the cap should be revised to support farmers
	R	the cap must be reorganised to encourage farmers
3	B	but unfortunately i have to my old stand
	L	but unfortunately i must stand by my old position
	R	but unfortunately i shall have to adhere to my old point of view
4	B	how easy it is
	L	as simple as that
	R	it is as simple as that
5	B	it is creatures with an sentient beings
	L	there are creatures with an awareness
	R	they are sentient creatures
Limited Reordering failure		
6	B	today we can you with satisfaction a compromise proposal put
	L	today we can you and i am pleased submitting a compromise proposal
	R	i am pleased to see that we have today arrived at a compromise motion
7	B	this is a common policy
	L	is this we need a common policy
	R	a common policy is required in this area

Table 2: Examples of translation, B: Baseline, L: Our Limited Reordering model, R: Human Reference

and below that some examples where the baseline beats our model on translation quality. Example 3 takes advantage of a moved verb; the original Dutch sentence here ends with a verb indicating that the situation is unchanged. Example 2 also takes advantage of a moved final verb. In example 4, the baseline gets confused by the verb-final behaviour.

6 Discussion

The Full Reordering model, without the limitation of moving only one token constituent and the R threshold, reorders most of the sentences: 90% of the Dutch sentences get reordered. As can be seen from Table 1 our scores drop even further than using only the Alpino model. Getting too close to the ideal of limiting dependency distance, we actually move large clauses around so much, for a language which depends on word order to mark grammatical function, that the sentences gets scrambled and lose too much information. Manually judging the sentence we can find examples where the sentence locally improved in quality, but overall most translations are worse than the baseline.

In addition, the phrase table for the Fully Reordered model is much smaller than the phrase table for the non-reordered model. At first we

thought this was due to the new model generalising better. For example we find the verb particle more often next to the governing verb than in other contexts. However a better explanation for this in light of the negative results for this model is based on the GIZA++ training. Eventually the phrases are derived from the output of a GIZA++ training which iteratively tries to build IBM model 4 (Brown et al., 1994) alignments on the sentence pairs. When the source sentences are extremely reordered (e.g. an object moved before the subject) the distortion model of model 4 makes it harder to link these words, so eventually we would extract fewer phrases.

Regarding the results of the Limited model compared to original Collins et al. results, we used the default settings for Pharaoh, while Collins et al. probably did not. This could explain the difference in baseline scores (0.207 vs 0.252) for languages with similar syntactic features.

Comparing the results of the Limited model to the reimplementations of the Collins rules in this work, we see that we have achieved half of the improvement without using any language-specific rules. That the approach works by bringing related words closer, in a way that can be taken advantage of by the phrase mechanisms of SMT without ex-

PLICITLY matching the syntax of the target language, suggests that this is a source of the improvement obtained by the reordering approach of Collins et al.

In future work we would like to implement the proposed reordering technique after correcting for the parser distortions, hopefully confirming the Collins results over the standard baseline for this language pair, and also confirming the relative improvement of the approach of this paper.

7 Conclusion

Previous work has demonstrated that reordering of the text in the source language can lead to an improvement in machine translation quality. Earlier methods either tried to learn appropriate rules for reordering, or have used hand-coded rules that take account of specific differences between language pairs. In this work, we have explored how a claimed universal property of language — that there is a tendency to minimise the distance between a head and its dependents — can be adapted to automatically reorder constituents in the source language. This leads to an improvement in translation quality when the source language, Dutch, is one where this tendency is less present than in the target language English. We demonstrate that, in the Dutch-English case, unrestricted application of the head-dependent distance minimisation strategy is not optimal, and that a restricted version of the strategy does best; we show that this can achieve half of the improvement of the handcrafted rules by using only one language-independent principle, and suggests that what is contributing to the improvement obtained in the reordering is the collapsing of elements into the phrasal window.

References

Gosse Bouma, Gertjan van Noord, and Robert Malouf. 2000. Alpino: Wide Coverage Computational Analysis of Dutch. In *Computational Linguistics in the Netherlands (CLIN)*.

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1994. The Mathematic

of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 531–540, Ann Arbor, Michigan, June.

Marta R. Costa-Juss and Jos A. R. Donollosa. 2006. Statistical Machine Reordering. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 70–76.

J. A. Hawkins. 1990. A parsing theory of word order universals. *Linguistic Inquiry*, 21:223–261.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, Morristown, NJ, USA. Association for Computational Linguistics.

Philipp Koehn. 2003. Europarl: A Multilingual Corpus for Evaluation of Machine Translation Philipp Koehn, Draft, Unpublished.

Philip Koehn. 2004. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models, Philipp Koehn, AMTA.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Jul.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings Intl. Conf. on Spoken Language Processing*, volume 2, pages 901–904.

Joseph P. Turian, Luke Shen, and I. Dan Melamed. 2003. Evaluation of machine translation and its evaluation. In *Proceedings of MT Summit IX*.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (Coling)*, pages 508–514.