Chart-Based Transfer Rule Application in Machine Translation

Adam Meyers
New York University
meyers@cs.nyu.edu

Michiko Kosaka
Monmouth University
kosaka@monmouth.edu

Ralph Grishman
New York University
grishman@cs.nyu.edu

Abstract

Transfer-based Machine Translation systems require a procedure for choosing the set of transfer rules for generating a target language translation from a given source language sentence. In an MT system with many competing transfer rules, choosing the best set of transfer rules for translation may involve the evaluation of an explosive number of competing sets. We propose a solution to this problem based on current best-first chart parsing algorithms.

1 Introduction

Transfer-based Machine Translation systems require a procedure for choosing the set of transfer rules for generating a target language translation from a given source language sentence. This procedure is trivial for a system if, given a context, one transfer rule can be selected unambiguously. Otherwise, choosing the best set of transfer rules may involve the evaluation of numerous competing sets. In fact, the number of possible transfer rule combinations increases exponentially with the length of the source language sentence. This situation mirrors the problem of choosing productions in a nondeterministic parser. In this paper, we describe a system for choosing transfer rules, based on statistical chart parsing (Bobrow, 1990; Chitrao and Grishman, 1990; Caraballo and Charniak, 1997; Charniak et al., 1998).

In our Machine Translation system, transfer rules are generated automatically from parsed parallel text along the lines of (Matsumoto et al., 1993; Meyers et al., 1996; Meyers et al., 1998b). Our system tends to acquire a large number of transfer rules, due mainly to alternative ways of translating the same sequences of words, non-literal translations in parallel text and parsing errors. It is therefore crucial that our system choose the best set of rules efficiently. While the technique discussed here obviously applies to similar such systems, it could also apply to hand-coded systems in which each word or group of words is related to more than one transfer rule. For example, both Multra (Hein, 1996) and the Eurotra system described in (Way et al., 1997) require components for deciding which combination of transfer rules to use. The proposed technique may be used with systems like these provided that all transfer rules are assigned initial scores rating their appropriateness for translation. These appropriateness ratings could be dependent or independent of context.[1]

2 Previous Work

The MT literature describes several techniques for deriving the appropriate translation. Statistical systems that do not incorporate linguistic analysis (Brown et al., 1993) typically choose the most likely translation based on a statistical model, i.e.., translation probability determines the translation. (Hein, 1996) reports a set of (hand-coded) feature structure based preference rules to choose among alternatives in Multra. There is some discussion about adding some transfer rules automatically acquired from corpora to Multra.[2] Assuming that they overgenerate rules (as we did), a system like the one we propose should be beneficial. In (Way et al., 1997), many different criteria are used to choose transfer rules to execute including: preferences for specific rules over general ones, and complex rule notation that insures that few rules can apply to the same set of words.

The Pangloss Mark III system (Nirenburg

[1] This translation procedure would probably complement, not replace existing procedures in these systems.

[2] `http://stp.ling.uu.se/~corpora/plug/reports/ansk_last/` is a report on this project for Multra.

and Frederking, 1995) uses a chart-walk algorithm to combine the results of three MT engines: an example-based engine, a knowledge-based engine, and a lexical-transfer engine. Each engine contributes its best edges and the chart-walk algorithm uses dynamic programming to find the combination of edges with the best overall score that covers the input string. Scores of edges are normalized so that the scores from the different engines are comparable and weighted to favor engines which tend to produce better results. Pangloss's algorithm combines whole MT systems. In contrast, our algorithm combines output of individual transfer rules within a single MT system. Also, we use a best-first search that incorporates a probabilistic-based figure of merit, whereas Pangloss uses an empirically based weighting scheme and what appears to be a top-down search.

Best-first probabilistic chart parsers (Bobrow, 1990; Chitrao and Grishman, 1990; Caraballo and Charniak, 1997; Charniak et al., 1998) strive to find the best parse, without exhaustively trying all possible productions. A probabilistic figure of merit (Caraballo and Charniak, 1997; Charniak et al., 1998) is devised for ranking edges. The highest ranking edges are pursued first and the parser halts after it produces a complete parse. We propose an algorithm for choosing and applying transfer rules based on probability. Each final translation is derived from a specific set of transfer rules. If the procedure immediately selected these transfer rules and applied them in the correct order, we would arrive at the final translation while creating the minimum number of edges. Our procedure uses about 4 times this minimum number of edges. With respect to chart parsing, (Charniak et al., 1998) report that their parser can achieve good results while producing about three times the minimum number of edges required to produce the final parse.

3 Test Data

We conducted two experiments. For experiment1, we parsed a sentence-aligned pair of Spanish and English corpora, each containing 1155 sentences of Microsoft Excel Help Text. These pairs of parsed sentences were divided into distinct training and test sets, ninety percent for training and ten percent for test. The training

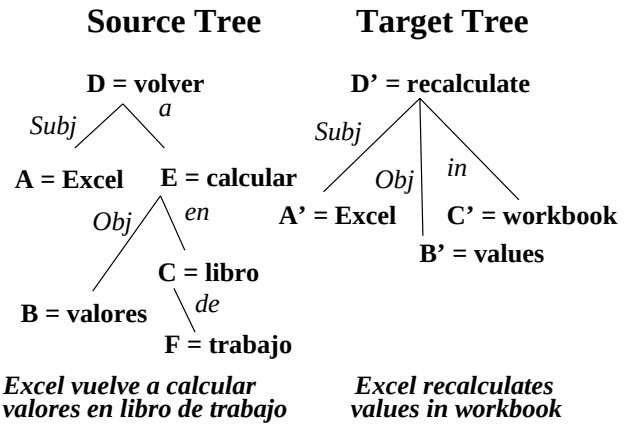

Figure 1: Spanish and English Regularized Parse Trees

set was used to acquire transfer rules (Meyers et al., 1998b) which were then used to translate the sentences in the test set. This paper focuses on our technique for applying these transfer rules in order to translate the test sentences.

The test and training sets in experiment1 were rotated, assigning a different tenth of the sentences to the test set in each rotation. In this way we tested the program on the entire corpus. Only one test set (one tenth of the corpus) was used for tuning the system during development. Transfer rules, 1109 on average, were acquired from each training set and used for translation of the corresponding test set. For Experiment 2, we parsed 2617 pairs of aligned sentences and used the same rotation procedure for dividing test and training corpora. The Experiment 2 corpus included the experiment1 corpus. An average of 2191 transfer rules were acquired from a given set of Experiment 2 training sentences.

Experiment1 is orchestrated in a careful manner that may not be practical for extremely large corpora, and Experiment 2 shows how the program performs if we scale up and eliminate some of the fine-tuning. Apart from corpus size, there are two main difference between the two experiments: (1) the experiment1 corpus was aligned completely by hand, whereas the Experiment 2 corpus was aligned automatically using the system described in (Meyers et al., 1998a); and (2) the parsers were tuned to the experiment1 sentences, but not the Experiment 2 sentences (that did not overlap with experiment1).

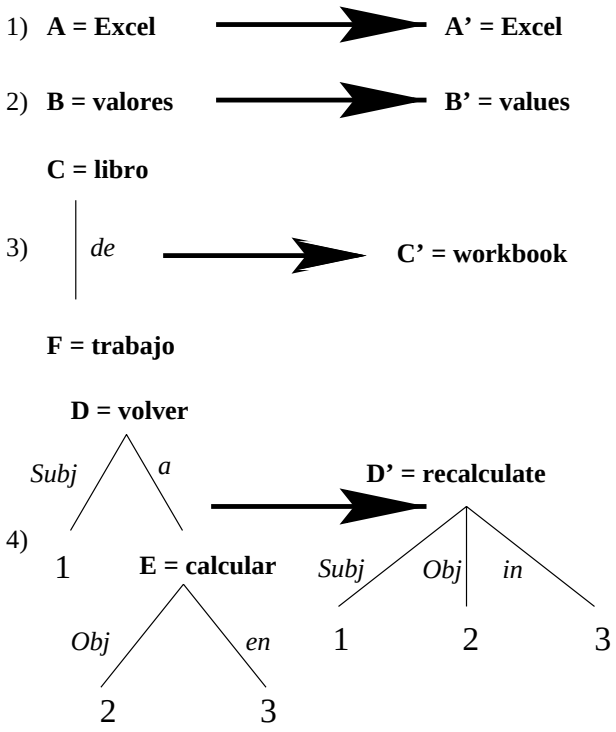

Figure 2: A Set of Transfer Rules

4 Parses and Transfer Rules

Figure 1 is a pair of "regularized" parses for a corresponding pair of Spanish and English sentences from Microsoft Excel help text. These are F-structure-like dependency analyses of sentences that represent predicate argument structure. This representation serves to neutralize some differences between related sentence types, e.g., the regularized parse of related active and passive sentences are identical, except for the feature value pair {Mood, Passive}. Nodes (values) are labeled with head words and arcs (features) are labeled with grammatical functions (subject, object), prepositions (*in*) and subordinate conjunctions (*before*).[3]. For demonstration purposes, the source tree in Figure 1 is the input to our translation system and the target tree is the output.

The transfer rules in Figure 2 can be used to convert the input tree into the output tree. These transfer rules are pairs of corresponding rooted substructures, where a substructure (Matsumoto et al., 1993) is a connected set of arcs and nodes. A rule consists of either a pair of "open" substructures (rule 4) or a pair of "closed" substructures (rules 1, 2 and 3). Closed substructures consist of single nodes (A,A',B,B',C') or subtrees (the left hand side of rule 3). Open substructures contain one or more open arcs, arcs without heads (both substructures in rule 4).

[3]Morphological features and their values (Gram-Number: plural) are also represented as arcs and nodes.

5 Simplified Translation with Tree-based Transfer Rules

The rules in Figure 2 could combine by filling in the open arcs in rule 4 with the roots of the substructures in rules 1, 2 and 3. The result would be a closed edge which maps the left tree in Figure 1 into the right tree. Just as edges of a chart parser are based on the context free rules used by the chart parser, edges of our translation system are based on these transfer rules. Initial edges are identical to transfer rules. Other edges result from combining one closed edge with one open edge. Figure 3 lists the sequence of edges which would result from combining the initial edges based on Rules 1–4 to replicate the trees in Figure 1. The translation proceeds by incrementally matching the left hand sides of Rules 1–4 with the input tree (and insuring that the tree is completely covered by these rules). The right-hand sides of these compatible rules are also combined to produce the translation. This is an idealized view of our system in which each node in the input tree matches the left-hand side of exactly one transfer rule: there is no ambiguity and no combinatorial explosion. The reality is that more than one transfer rules may be activated for each node, as suggested in Figure 4.[4] If each of the six nodes of the source tree corresponded to five transfer rules, there are $5^6 = 15625$ possible combinations of rules to consider. To produce the output in Figure 3, a minimum of seven edges would be required: four initial edges derived from the original transfer rules plus three additional edges representing the combination of edges (steps 2, 3 and 4 in Figure 3). The speed of our system is measured by the number of actual edges divided by this minimum.

[4]The third example listed would actually involve two transfer rules, one translating "volver" to "repeat" and the second translating "calcular" to "calculate".

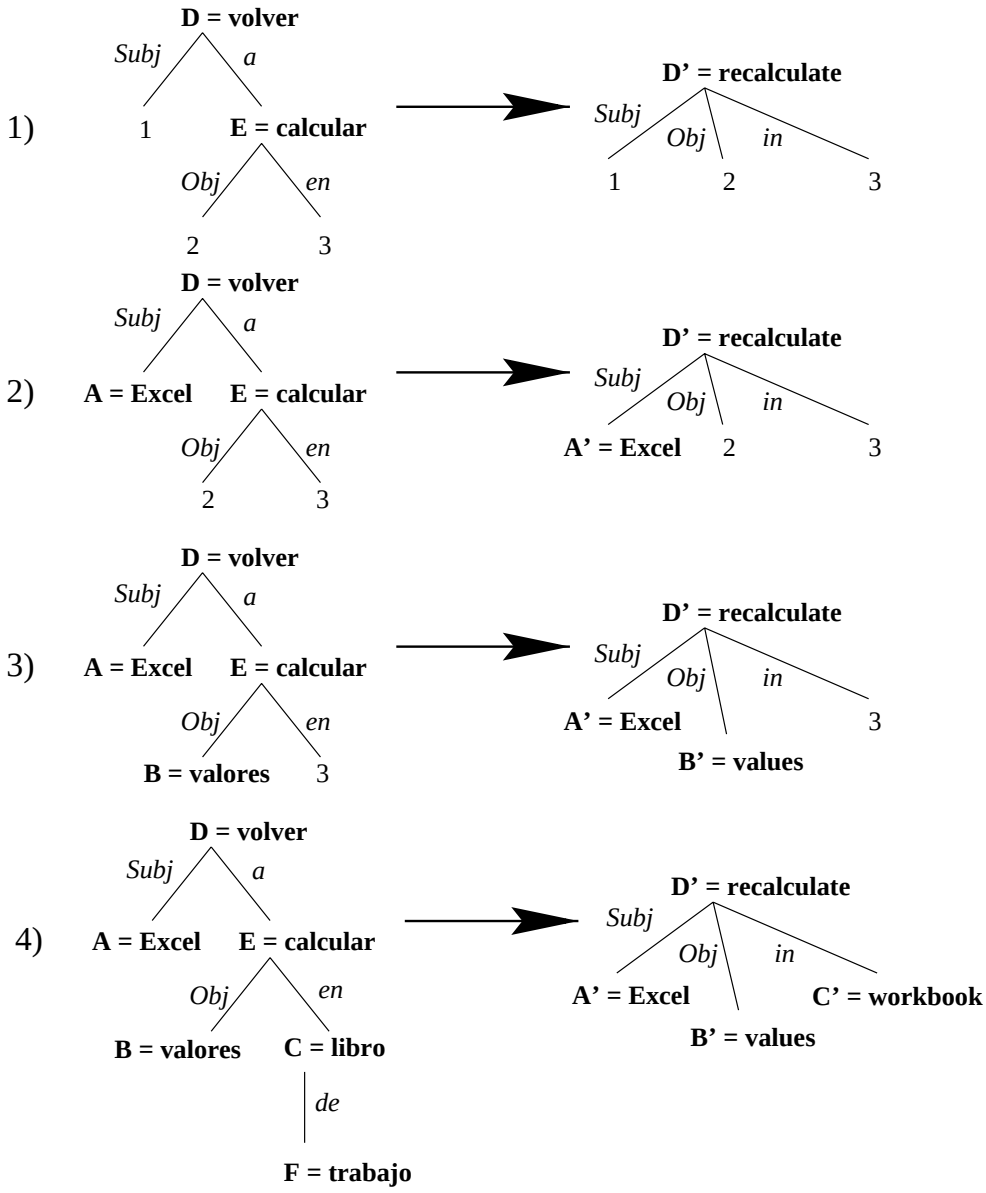

Figure 3: An Idealized Translation Procedure

6 Best First Translation Procedure

The following is an outline of our best first search procedure for finding a single translation:

1. For each node N, find T_N, the set of compatible transfer rules
2. Create initial edges for all T_N
3. Repeat until a "finished" edge is found or an edge limit is reached:
   (a) Find the highest scoring edge E
   (b) If complete, combine E with compatible incomplete edges
   (c) If incomplete, combine E with compatible complete edges
   (d) Incomplete edge + complete edge = new edge

The procedure creates one initial edge for each matching transfer rule in the database[5] and puts these edges in a

[5] The left-hand side of a matching transfer rule is compatible with a substructure in the input source tree.

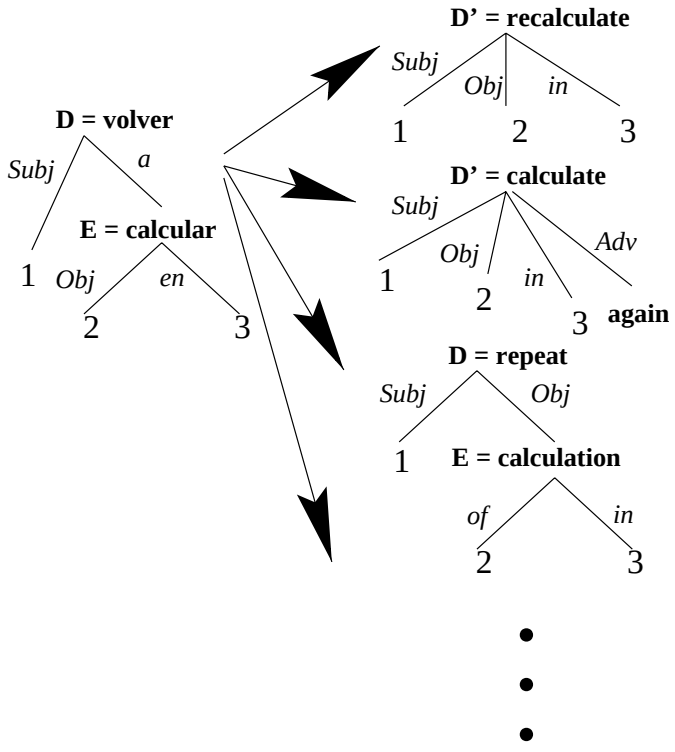

Figure 4: Multiple Transfer Rules for Each Substructure

queue prioritized by score. The procedure iteratively combines the best scoring edge with some other compatible edge to produce a new edge and inserts the new edge in the queue. The score for each new edge is a function of the scores of the edges used to produce it. The process continues until either an edge limit is reached (the system looks like it will take too long to terminate) or a complete edge is produced whose left-hand side is the input tree: we call this edge a "finished edge".

We use the following technique for calculating the score for initial edges.[6] The score for each initial edge E rooted at N, based on rule R is calculated as follows:

1. $SCORE1(S) = log_2(\frac{Freq(R)}{Freq(All\ Rules\ at\ N)})$

   Where the frequency (Freq) of a rule is the number of times it matched an example in the training corpus, during rule acquisition. The denominator is the combined frequencies of all rules that match N.

[6] This is somewhat dependent on the way these transfer rules are derived. Other systems would probably have to use some other scoring system.

Experiment 1: 1155 sentences		
	Norm	No Norm
Total Translations	1153	1127
Over Edge Limit	2	28
Actual Edges	93,719	579,278
Minimum Edges	22,125	20,125
Edge Ratio	3.3	14.8
Accuracy	70.9	70.9
Experiment 2: 2617 sentences		
	Norm	No Norm
Total Translations	2610	2544
Over Edge Limit	7	73
Actual Edges	262,172	1,398,796
Minimum Edges	48,570	42,770
Edge Ratio	4.0	15.5
Accuracy	62.6	61.5

Figure 5: Results

2. $Score(S) = Score1(S) - Norm\ Factor$

   Where the Norm (normalization) factor is equal to the highest SCORE1 for any rule matching N.

Since the log_2 of probabilities are necessarily negative, this has the effect of setting the E of each of the most probable initial edges to zero. The scores for non-initial edges are calculated by adding up the scores of the initial edges of which they are composed.[7]

Without any normalization ($Score(S) = SCORE1(S)$), small trees are favored over large trees. This slows down the process of finding the final result. The normalization we use insures that the most probable set of transfer rules are considered early on.

7 Results

Figure 5 gives our results for both experiments 1 and 2, both with normalization (Norm) and without (No Norm). "Total Translations" refer to the number of sentences which were translated successfully by the system and "Over Edge Limit" refers to the number of sentences which caused the system to exceed the edge limit, i.e., once the system produces over 10,000 edges, translation failure is assumed. The system cur-

[7] Scoring for special cases is not included in this paper. These cases include rules for conjunctions and rules for words that do not match any transfer rules in a given context (we currently leave the word untranslated.)

rently will only fail to produce some translation for any input if the edge limit is exceeded. "Actual Edges" refers to the total number of edges used for attempting to translate every sentence in the corpus. "Minimum Edges" refer to the total minimum number of edges required for successful translations. The "Edge Ratio" is a ratio between: (1) "Total Edges" less the number of edges used in failed translations; and (2) The "Minimum Edges". This ratio, in combination with, the number of "Over Edge Limit" measures the efficiency of a given system. "Accuracy" is an assessment of translation quality which we will discuss in the next section.

Normalization caused significant speed-up for both experiments. If you compare the total number of edges used with and without normalization, speed-up is a factor of 6.2 for Experiment 1 and 5.3 for Experiment 2. If you compare actual edge ratios, speed-up is a factor of 4.5 for Experiment 1 and 3.9 for Experiment 2. In addition, the number of failed parses went down by a factor of 10 for both experiments. As should be expected, accuracy was virtually the same with and without normalization, although normalization did cause a slight improvement. Normalization should produce the essentially the same result in less time.

These results suggest that we can probably count on a speed-up of at least 4 and a significant decline in failed parses by using normalization. The differences in performance on the two corpora are most likely due to the degree of hand-tuning for Experiment 1.

7.1 Our Accuracy Measure

"Accuracy" in Figure 5 is the average of the following score for each translated sentence:

$$\frac{|T_{NYU} \bigcap T_{MS}|}{1/2 \times (|T_{NYU}| + |T_{MS}|)}$$

T_{NYU} is the set of words in NYU's translation and T_{MS} is the set of words in the original Microsoft translation. If T_{NYU} = "A B C D E" and T_{MS} = "A B C F", then the intersection set "A B C" is length 3 (the numerator) and the average length of T_{NYU} and T_{MS} is 4 1/2 (the denominator). The accuracy score equals $3 \div 4\ 1/2 = 2/3$. This is a Dice coefficient comparison of our translation with the original. It is an inexpensive method of measuring the performance of a new version of our system. Improvements in the average accuracy score for our sample set of sentences usually reflect an improvement in overall translation quality. While it is significant that the accuracy scores in Figure 5 did not go down when we normalized the scores, the slight improvement in accuracy should not be given much weight. Our accuracy score is flawed in that it cannot account for the following facts: (1) good paraphrases are perfectly acceptable; (2) some differences in word selection are more significant than others; and (3) errors in syntax are not directly accounted for.

NYU's system translates the Spanish sentence "1. Seleccion la celda en la que desea introducir una referencia" as "1. select the cell that you want to enter a reference in". Microsoft translates this sentence as "1. Select the cell in which you want to enter the reference". Our system gives NYU's translation an accuracy score of .75 due to the degree of overlap with Microsoft's translation. A human reviewer would probably rate NYU's translation as completely acceptable. In contrast, NYU's system produced the following unacceptable translation which also received a score of .75: the Spanish sentence "Elija la función que desea pegar en la fórmula en el cuadro de diálogo Asistente para funciones" is translated as " "Choose the function that wants to paste Function Wizard in the formula in the dialog box", in contrast with Microsoft's translation "Choose the function you want to paste into the formula from the Function Wizard dialog box". In fact, some good translations will get worse scores than some bad ones, e.g., an acceptable one word translation can even get a score of 0, e.g.,"SUPR" was translated as "DEL" by Microsoft and as "Delete" by NYU. Nevertheless, by averaging this accuracy score over many examples, it has proved a valuable measure for comparing different versions of a particular system: better systems get better results. Similarly, after tweaking the system, a better translation of a particular sentence will usually yield a better score.

8 Future Work

Future work should address two limitations of our current system: (1) Bad parses yield bad transfer rules; and (2) sparse data limits the size of our transfer rule database and our options for

applying transfer rules selectively. To attack the "bad parse" problem, we are considering using our MT system with less-detailed parsers, since these parsers typically produce less error-prone output. We will have to conduct experiments to determine the minimum level of detail that is needed.[8]

Previous to the work reported in this paper, we ran our MT system on bilingual corpora in which the sentences were aligned manually. The cost of manual alignment limited the size of the corpora we could use. A lot of our recent MT research has been focused on solving this sparse data problem through our development of a sentence alignment program (Meyers et al., 1998a). We now have 300,000 automatically aligned sentences in the Microsoft help text domain for future experiments. In addition to providing us with many more transfer rules, this should allow us to collect transfer rule co-occurrence information which we can then use to apply transfer rules more effectively, perhaps improving translation quality. In a preliminary experiment along these lines using the Experiment 1 corpus, co-occurrence information had no noticeable effect. However, we are hopeful that future experiments with 300,000 aligned sentences (300 times as much data) will be more successful.

References

Robert J. Bobrow. 1990. Statistical agenda parsing. In *DARPA Speech and Language Workshop*, pages 222–224.

Peter Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19:263–312.

Sharon A. Caraballo and Eugene Charniak. 1997. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics*, 24:275–298.

Eugene Charniak, Sharon Goldwater, and Mark Johnson. 1998. Edge-Based Best-First Chart Parsing. In *Proceedings of the Sixth Annual Workshop for Very Large Corpora*, Montreal.

Mahesh V. Chitrao and Ralph Grishman. 1990. Statistical parsing of messages. In *DARPA Speech and Language Workshop*, pages 263–266.

Anna Sågvall Hein. 1996. Preference Mechanisms of the Multra Machine Translation System. In Barbara H. Partee and Petr Sgall, editors, *Discourse and Meaning: Papers in Honor of Eva Hajiĉová*. John Benjamins Publishing Company, Amsterdam.

Y. Matsumoto, H. Ishimoto, T. Utsuro, and M. Nagao. 1993. Structural Matching of Parallel Texts. In *31st Annual Meeting of the Association for Computational Linguistics: "Proceedings of the Conference"*.

Adam Meyers, Roman Yangarber, and Ralph Grishman. 1996. Alignment of Shared Forests for Bilingual Corpora. In *Proceedings of Coling 1996: The 16th International Conference on Computational Linguistics*, pages 460–465.

Adam Meyers, Michiko Kosaka, and Ralph Grishman. 1998a. A Multilingual Procedure for Dictionary-Based Sentence Alignment. In *Proceedings of AMTA'98: Machine Translation and the Information Soup*, pages 187–198.

Adam Meyers, Roman Yangarber, Ralph Grishman, Catherine Macleod, and Antonio Moreno-Sandoval. 1998b. Deriving Transfer Rules from Dominance-Preserving Alignments. In *Proceedings of Coling-ACL98: The 17th International Conference on Computational Linguistics and the 36th Meeting of the Association for Computational Linguistics.*

Sergei Nirenburg and Robert E. Frederking. 1995. The Pangloss Mark III Machine Translation System: Multi-Engine System Architecture. Technical report, NMSU CRL, USC ISI, and CMU CMT.

Andrew Way, Ian Crookston, and Jane Shelton. 1997. A Typology of Translation Problems for Eurotra Translation Machines. *Machine Translation*, 12:323–374.

[8] One could set up a continuum from detailed parsers like Proteus down to shallow verb-group/noun-group recognizers, with the Penn treebank based parsers lying somewhere in the middle. As one travels down the continuum to the lower detail parsers, the error rate naturally decreases.