EUROPEAN ASSOCIATION FOR MACHINE TRANSLATION

MT Summit VIII

Workshop on Example-Based Machine Translation

18-22 SEPTEMBER 2001
MT SUMMIT VIII
SPAIN
SANTIAGO DE COMPOSTELA

Proceedings of the Workshop

18 September 2001
Santiago de Compostela, Spain

Programme Committee:

- Sivaji Bandyopadhyay, India
- Ralf Brown, USA
- Michael Carl, Germany
- Ilyas Cicekli, Turkey
- Brona Collins, Belgium
- Oliver Streiter, Taiwan
- Stephan Vogel, Germany
- Andy Way, Ireland

Workshop Website:

http://www.compapp.dcu.ie/~away/EBMT.html

INTRODUCTION

This volume contains the papers for presentation at the Workshop on Example Based Machine Translation, which is part of the MT Summit VIII held on September 18, in Santiago de Compostela, Spain.

In recent years, corpora of multilingual translated texts have become widely available for a number of languages. Notwithstanding the seminal paper by Nagao (84), it is primarily since the early 90's that such bilingual texts have been exploited in the area of Machine Translation (MT).

The two main paradigmatic approaches which have been proposed are Statistics-based Machine Translation (SBMT) and Example-Based Machine Translation (EBMT). A related variant of EBMT that we ignore here, despite being widely used in the localisation area, is that of Translation Memories (TM).

While translation memory systems are used in restricted domains, SBMT systems require training on huge, good quality bilingual corpora. As a consequence TMs can hardly be applied as a general purpose solution to MT and SBMT as yet cannot produce complex translations to the desired quality, even if such translations are given to the system in the training phase. EBMT seeks to exploit and integrate a number of knowledge resources, such as linguistics and statistics, and symbolic and numerical techniques, for integration into one framework. In this way, rule-based morphological, syntactic and/or semantic information is combined with knowledge extracted from bilingual texts which is then re-used in the translation process.

However, it is unclear how one might combine the different knowledge resources and techniques in an optimal way. In EBMT, therefore, the question is asked: what can be learned from a bilingual corpus and what needs to be manually provided? Furthermore, we remain uncertain as to how far the EBMT methodology can be pushed with respect to translation quality and/or translation purpose. Finally, one wonders what the implications and consequences are for size and quality of the reference translations, (computational) complexity of the system, sizeability and transportability, if such an approach is taken. Given this background, the contributions of the workshop seek to shed some light on these open questions, among others.

Instead of inviting a speaker, the organizers have decided to feature two outstanding papers submitted to the workshop which are allocated a longer presentation and discussion time. We are glad that Harold Somers and Kevin McTait have accepted this invitation.

We are very grateful to the programme committee for the effort they put in reviewing the papers and their comments and suggestions for the workshop. Additionally we would like to thank Ute Hauck for converting file formats and compiling these proceedings into printable form.

Michael Carl and Andy Way

July 2001

Table of Contents

Transfer-Rule Induction for Example-Based Translation

Ralf D. Brown
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
ralf+@cs.cmu.edu

Abstract

Previous work has shown that grammars and similar structure can be induced from unlabeled text (both monolingually and bilingually), and that the performance of an example-based machine translation (EBMT) system can be substantially enhanced by using clustering techniques to determine equivalence classes of individual words which can be used interchangeably, thus converting translation examples into templates. This paper describes the combination of these two approaches to further increase the coverage (or conversely, decrease the required training text) of an EBMT system. Preliminary results show that a reduction in required training text by a factor of twelve is possible for translation from French into English.

1 Introduction

Lexicalist Example-Based Machine Translation (EBMT) systems such as those of Veale and Way (1997) and the author (Brown, 1999) have the advantage that they require little or no additional knowledge beyond the parallel text forming the example base, but the disadvantage that the example base must be quite large to provide good coverage of unrestricted texts. Since parallel texts of the required size are often difficult or (for less-used languages) even impossible to obtain, there have been several efforts to reduce the data needs by generalizing the training examples into templates and then perform template matching (see Figure 1).

Three approaches to generalization have been used: manually-generated equivalence classes, automatically-extracted equivalence classes, and transfer-rule induction. The EBMT systems mentioned above both convert translation examples into templates using manually-created information, such as from a machine-readable dictionary with part-of-speech information, to replace words with tokens indicating the class of word which may occur in a particular location. More recently, the author added automatically-generated equivalence classes using word-level clustering (Brown, 2000) and Cicekli and Güvenir (2001) have implemented transfer-rule induction from parallel text.

This paper reports the results of combining the latter two approaches, using transfer-rule induction followed by word-level clustering to find not only single words but also transfer rules which can be combined into equivalence classes.

2 Transfer-Rule Induction

To induce a set of grammar rules from a parallel corpus, we make the same assumption used by Cicekli and Güvenir (2000; 2001) and van Zaanen (2000): when two sentence pairs in the corpus have some part in common but differ in some other part, the similar and dissimilar parts each correspond to some coherent constituent. Note that such a "constituent" need not be a traditional constituent as used by linguists, such as a noun phrase or prepositional phrase; for our purposes, it suffices that the groupings which are found can be used interchangeably.

Initially, the system only searches for pairs of training instances where the source-language halves show the pattern

$$S_1 \ D \ S_2$$

where S_1 and S_2 are the same in both instances (at most one of these may be the empty string) and D differs between the two training instances, but may contain common subsequences. The algorithm is outlined in Figure 2 and described in detail below. Naturally, such a simple pattern will not capture all interesting phenomena; future work will address more complex patterns.

A recursive method is used to find sets of training instances with common word sequences at beginning or end ("initial string" and "final string" or "prefix" and "suffix"). After sorting the sentence pairs by their source-language sentences, one can simply perform a linear scan of the collection for runs of training instances with at least I words in common. Each run found determines a subcorpus on which we can search for runs with at least $I+1$ words in common. At each level in the recursion, sorting the training instances as though the order of their words were reversed allows the same

Training Input	205 delegates met in London. 200 delegates met in Paris.
String Match	delegates met in
Template Match	<number> delegates met in <city>.

Figure 1: String Match vs. Template Match

```
1. read the corpus into memory, creating a rough bitext mapping for each bilingual sentence pair
2. sort the corpus alphabetically by source-language sentence
3. for each $F$, find all sequences of sentence pairs which share the same first $F$ words in the source language
4. for each sequence, create a subcorpus and:
   (a) sort the subcorpus alphabetically by reversed source-language sentence
   (b) for each $L$, find all sequences of sentence pairs which share the same last $L$ words in the source language
   (c) for each sequence, create another subcorpus and:
       i. perform a pairwise comparison between sentence pairs, adding the differences to a new equivalence class and to the corpus. The bitext map is used to discard those differences which do not appear to match between source- and target-language sentences.
       ii. if sufficiently long, add the common initial/final strings to the corpus
5. apply the learned rewriting rules to the corpus, except to sentence pairs where doing so would generate a single token
6. repeat steps 2 through 5 until no more new equivalence classes are added or the number of iterations reaches a preset maximum
```

Figure 2: The Induction Process

type of scan to find runs of training instances with a common final string of specified length.

Consider the small set of example sentence pairs in Figure 3. These all share the common initial string "nous regardons" and final string "."; further, all but the first one share the initial string "nous regardons les", and instances 2-4 share the initial string "nous regardons les approvisionnements en". We will first process the smallest set (instances 2-4), then the intermediate set (instances 2-5), and finally the complete set (instances 1-5).

Thus, for each combination of initial-string length and final-string length, a set of training instances has been defined by the above scans, whose differences may be assigned to an equivalence class. These instances are then compared pair-wise to determine the differences between them. For each pair, the target-language halves are compared, also segmenting them into a common initial string, dissimilar central portion, and common final string. To ensure that the dissimilar center does in fact correspond to the difference on the source-language side, a bitext mapping is used. The bitext mapping is generated for each training instance from a bilingual dictionary, indicating which words in the target-language half potentially correspond with each source-language word. If, for either of the two training instances, the bitext map rules out any part of the target-language difference, neither instance is added to the current equivalence class (one or both of the instances may eventually have its center portion added when compared against other sentence pairs in the corpus).

Should a pair of instances pass the bitext-map test, the portions which differ between the two are added to the training corpus as new (but shorter) training instances, and are added to the equivalence class of all instances having the same initial and final strings. After the pair-wise comparison between each pair in the set of instances with that common prefix and suffix is complete, the initial and final strings themselves are also added to the corpus as two additional training instance provided that they are sufficiently long (currently, at least two words each).

Once the corpus has been completely processed, the result is a corpus augmented by various sentence fragments which are assumed to be constituents of some kind. We now apply the learned equivalence classes interpreted as a set of rewriting rules or a context free grammar, replacing each instance of a class member by the class name.

Input:

1. nous regardons la production de acier .
   we are watching steel production .
2. nous regardons les approvisionnements en énergie .
   we are watching energy supplies .
3. nous regardons les approvisionnements en engrais .
   we are watching fertilizer supplies .
4. nous regardons les approvisionnements en matériel .
   we are watching equipment supplies .
5. nous regardons les produits chimiques agricoles .
   we are watching agricultural chemicals .

Rewritten Corpus:

- nous regardons la production de acier .
  we are watching steel production .
- nous regardons les <cl_2> .
  we are watching <cl_2> .
- nous regardons
  we are watching

Induced Rules – **<CL_2>**:

- approvisionnements en <CL_0>
  <CL_0> supplies
- produits chimiques agricoles
  agricultural chemicals

Induced Rules – **<CL_0>**:

- engrais
  fertilizer
- matériel
  equipment
- énergie
  energy

Figure 3: Sample Induction

The replacements may occur anywhere in a sentence pair, including the portions which had been used as common prefix or suffix strings during the learning phase. An exception is made if the end result of applying the grammar to a training instance is a single class name; in this case, the training instance is left unchanged. The rationale for applying rewriting rules in this manner is to increase the similarity of the items in the corpus to permit more matches on the next iteration, e.g. two instances which previously had different initial segments may have the same initial segment after rewriting rules are applied, because the differing phrases are both members of the same equivalence class.

At this point, if any changes were made, the entire process repeats using the updated corpus, until no more equivalence classes can be created. To forestall an extremely lengthy execution time should a large number of iterations be required, the program can also terminate learning after a specified number of iterations.

After the induction completes, it can optionally be re-run in the reverse direction, comparing target-language sentences with each other. This feature can increase the yield of equivalence classes by 50% or more, since the target-language sentences will show different patterns of similarities with each other which were not captured during the first pass.

Figure 3 shows the result of this process. The pairwise comparison between instances 2 through 4 yields the single-word differences which have been added to equivalence class <CL_0>. The further comparisons between instances 2 through 5 yield the equivalences in <CL_2>; after applying the rewriting rules created by <CL_0>, three of its members collapse into a single rule containing an equivalence-class marker.

The output of the induction phase is a set of

1. find bilingual word pairs which uniquely correspond to each other according to a bitext mapping generated with a bilingual dictionary
2. accumulate counts for the words in the immediate vicinity of each occurrence of a word pair, as well as the frequency of the word pair itself
3. convert the word counts into weighted term vectors, and generate an initial cluster for each vector; associate the word pair's frequency with the term vector
4. repeatedly find the two most similar clusters whose similarity is above the threshold value for the lesser of their frequency values and merge them (adding the associated frequencies), until no pair of clusters is above threshold or only two clusters remain
5. if the number of clusters is still above 2500, relax the clustering by allowing those clusters with frequency values of 5 or less to merge regardless of the clustering threshold for their frequency (if necessary, repeat for frequencies of 10, 15, etc.)
6. output the clusters with more than one member, recovering the word pair with which each vector in a cluster is associated

Figure 4: Clustering Algorithm

parallel rewriting rules which form the transfer-rule grammar (see Figure 5 for a few examples from actual runs), and optionally an updated parallel corpus with the rewriting rules added and already applied. The updated corpus which is already present in the computer's memory can then be used as input to the word-clustering phase.

3 Word Clustering

To cluster words into equivalence classes, we used the approach of (Brown, 2000), outlined in Figure 4. The main feature of this approach is a transformation step which converts the word-clustering problem into a document-clustering problem.

The first step in word clustering is to determine *which* words should be clustered. Since it is also necessary to cluster bilingually, a dictionary is used to generate a rough bitext mapping between the source and target halves of each sentence pair in the training corpus. Whenever the bitext map indicates a unique correspondence between a word in the source-language sentence and some word in the target-language sentence, form a word pair from the source- and target-language words and treat it as an indivisible unit. These word pairs are what will be clustered.

For each occurrence of a word pair, add the source-language words immediately surrounding its occurrence (for these experiments, the three words preceding and the three words following) to a term vector which tallies all neighboring words across all occurrences of the word pair. This converts the problem into one of finding which term vectors cluster together, a standard document-clustering approach.

Next, the term vectors are clustered using bottom-up agglomerative clustering. Initially, one cluster is created for each vector; next, the two clusters with the highest similarity measure are merged, and the process is repeated until no more clusters have sufficiently high similarity with any other clusters. The similarity metric used is a term-weighted cosine similarity measure, e.g. the normalized inner product of the two vectors:

$$cos(\vec{u},\vec{v}) = \frac{\vec{u}\cdot\vec{v}}{||\vec{u}|| \times ||\vec{v}||}$$

The selected threshold for clustering proved to be excessively conservative for larger training sets, leaving a majority of word pairs in single-element clusters, so a back-off scheme was added to force the total number of final clusters closer to a target of 2500 clusters. If more than the target number of clusters remain when the first clustering pass terminates, clusters with frequency values (the sum of term frequencies for all word pairs included in the cluster) of five or less may merge regardless of the selected threshold. If necessary, further iterations will allow clusters with frequencies of up to 10, 15, 20, etc. to merge. This approach was selected to avoid the need for tuning the thresholds for each individual training set size, which would be impractical at best.

Once the vectors have been clustered, the clusters are output. For every cluster, the word pair to which each vector in the cluster corresponds is recovered and is written to the results file along with the name of the cluster. The results file becomes input to the EBMT system when it indexes the training corpus, allowing it to convert the parallel text into templates using the cluster names as class markers. Figure 6 shows a sample of the clusters produced from 107,000 words of parallel training text. The clustering process produced a total of 1178 clusters containing 3187 word pairs after discarding clusters containing only a single word pair (as such singletons are not useful).

```
<CL_18>:
plutôt ridicule
rather ridiculous

peut-être le condamner avec de fausses louanges , mais je ne y puis rien
If it is damning him with feint praise , I cannot help

<cl_847> monsieur l'Orateur
<cl_847> Mr . Speaker

bien pour Toronto
lucky in Toronto

...
```

```
<CL_119>:
question # 2 est oui
part two is yes

question # 2601 de la première session de la 29e législature ne tenait pas compte de les travaux de
construction entrepris en 1973 - 1974 à la résidence de le premier ministre à Ottawa
question No . 2,601 of the First Session of the twenty-ninth Parliament did not list construction
work carried out in the year 1973 - 74 at the Prime Minister's Ottawa residence
```

```
<CL_122>:
, confié à la garde de un service de sécurité , fut transporté par tout le pays à bord de un avion de
la Défense nationale et un exemplaire en a été remis à toutes les capitales provinciales au moment
même où je prenais la parole à la Chambre
was sent under security guard in national defence aircraft right across the country , and copies were
released in all provincial capitals simultaneously with my rising in the House

que <cl_644>
that the <cl_644>
```

Figure 5: Sample Replacement Rules

4 Combining Induction and Clustering

The induction process produces a large number of usually small equivalence classes, which limits the amount of generalization that can be produced. Hence, we would like to merge different classes of rewriting rules which are used in similar contexts, in the same manner as individual words are clustered.

A by-product of the induction algorithm is an updated corpus with all replacement rules already applied, leaving single-word markers in place of the phrases found by transfer-rule induction. Applying the clustering process to the modified corpus allows both words and replacement rules to cluster together. While replacement-rule nonterminals tend to cluster with other nonterminals, many clusters contain both words and nonterminals (see Figure 6).

5 Experimental Design

To gauge the effect of the different approaches, various combinations of method and training size were run and evaluated.

Four conditions were compared: simple string matching against the corpus (see Figure 1), matching templates formed using single-word clustering alone, templates formed using transfer-rule induction alone, and templates formed with the combination of clustering and induction. For French-English, the training data in each case consisted of a subset (up to 1.1 million words in 19730 sentence pairs) of the Hansard corpus made available by the Linguistic Data Consortium (Linguistic Data Consortium, 1997) and a bilingual dictionary formed by combining the ARTFL French-English dictionary (ARTFL Project, 1998) with a probabilistic dictionary extracted from the Hansard corpus. The test data consisted of 45,320 words of French text from a disjoint portion of the Hansard corpus.

The data for the Spanish-English experiments consisted of up to one million words of parallel text drawn primarily from the UN Multilingual Corpus(Graff and Finch, 1994) available from the Linguistic Data Consortium and a bilingual dictionary derived from the Collins Spanish-English

Cluster	French	English
507	NE	NOT
	NE	NO
524	SONT	ARE
	FURENT	WERE
568	CETTE	THIS
	CETTE	THAT
575	PROCHAIN	NEXT
	DERNIER	LAST
659	UNE	THE
	UNE	AN
	UNE	A
	UN	THE
	UN	A
	LE	THE
	LE	OF
	LE	IT
	LE	IN
	LE	A
	LA	THE
	LA	OF
	LA	IN
678	VALEUR	VALUE
	SÉCURITÉ	SECURITY
	PRODUCTION	PRODUCTION
	PLUS	MORE
	NÉCESSITÉ	NEED
	LIVRAISON	DELIVERY
	FA CON	WAY
	DATE	DATE
	CROISSANCE	GROWTH
	CONSTRUCTION	CONSTRUCT- ION
	COMMERCIAL- ISATION	MARKETING
1085	POINTS	POINTS
	LIVRES	POUNDS
	ANS	YEARS
1150	VALABLES	VALID
	TARD	LATER
1196	MILLIONS	MILLION
	MILLIARDS	BILLION
1776	ABSURDE	NONSENSE
	<CL_18>	<CL_18>
2158	PÊCHEURS	FISHERIES
	PÉNURIES	SHORTAGES
	OFFICIERS	OFFICERS
2609	<CL_54>	<CL_54>
	<CL_98>	<CL_98>
	<CL_375>	<CL_375>
	<CL_458>	<CL_458>
	<CL_462>	<CL_462>
	...6 more...	

Figure 6: Sample Clusters from 107,000 words

dictionary and statistically extracted from UN-corpus and other parallel text. The test data consisted of 9,059 words of Spanish test from a disjoint portion of the UN Multilingual Corpus.

The input to the actual EBMT system consisted of the bilingual dictionary (for word-level alignment) plus the original parallel text, and (as appropriate) the output of the clustering and/or induction phases. When the clustering algorithm was applied in isolation, the single-word rewriting rules found through clustering were supplied to EBMT. When the transfer-rule induction was used, the induced transfer rules were supplied to EBMT.

The transfer-rule induction was limited to twelve iterations in each direction. During development it was found that the process has largely converged after six iterations, even though total convergence may require fifteen or more iterations, with the last several iterations each adding only a few new equivalence classes with a handfull of members.

The performance measure used to determine the effectiveness of the various methods is the coverage of the test text, i.e. the percentage of the total words in the test input for which the EBMT system could generate at least one candidate translation. Although this metric does not measure quality, the design of the EBMT system generally enforces some minimum level of translation quality – the translation software will not output translations when the word-level alignment for the retrieved training example fails or is deemed too poor[1]. Recent manual judgements on a Mandarin Chinese-English version of the EBMT system (Zhang et al., 2001) have confirmed that increased coverage indeed correlates with improved translation quality.

Figure 7 shows some sample output, which will be discussed in more detail in Section 7; for the moment, it is important to note that the score shown is a penalty – 0.0 is considered a perfect alignment, while matches for which the penalty exceeds five times the number of words are not output at all.

6 Computational Complexity

Each iteration of the induction algorithm takes time $O(n^2)$, where n is the number of words of training text, since ultimately each sentence pair must be compared against every other sentence pair (subdividing the problem into shorter sequences does not increase the time complexity, and sorting is $O(n \ log \ n)$). The number of iterations required to run the induction to completion

[1]In fact, if all corpus matches yielded good alignments, coverage would be in excess of 90% instead of 80.14% for two million words of French-English training text using just string matching.

Il me semble qu'il conviendrait maintenant de reprendre le débat.
I think it would be proper at this time for the debate to continue.

String Match, 1.1 million words:

Match	Sc	Translation
il me semble qu'il	0	it seems to me that
il me semble que	0	It seems to me that
il me semble	0	It seems to me
il me	0	Let me
il me	0	me explore
me semble	0.3	seems to me
me semble	0	I think
semble que il	0	it seems this
semble que	0	seems that
que il	0	well as
maintenant de	2.5	now the
de reprendre	0	to resume
le débat .	1	in debate .
le débat .	0	the debate .
le débat	0	in debate
le débat	0	debate in
débat .	0	debate .

Induction+Clustering, 1.1 million words:

Match	Sc	Translation
il me semble qu'il	17	it seems to me that
semble qu'il	0	it seems this
qu'il	0	that he
il conviendrait	0	IT APPROPRIATE
maintenant	0	NOW
de reprendre	0	to resume
le débat .	0	THE DEBATE .
le débat .	1	in DEBATE .
débat .	0	DEBATE .

Figure 7: Sample Translation 1

is potentially $O(n)$, but appears in practice to be somewhat less than $O(log\ n)$; there is a three- to four-fold increase between a 50,000-word corpus and a twenty times larger million-word corpus. More experimentation with larger corpora (several to tens of millions of words) will be required to determine the actual value.

In practice, the first iteration takes the longest time, by a factor of two or more, and subsequent iterations complete more quickly. Per-iteration execution times generally continue to decrease until the fifth iteration, after which they tend to vary both up and down but stay relatively constant. Two factors are likely at work here: on each succeeding iteration, there are fewer and smaller runs of sentences, reducing the quadratic pair-wise comparison; and the individual training instances are shorter, either because they are fragments of an older instance or due to replacement of phrases by single tokens.

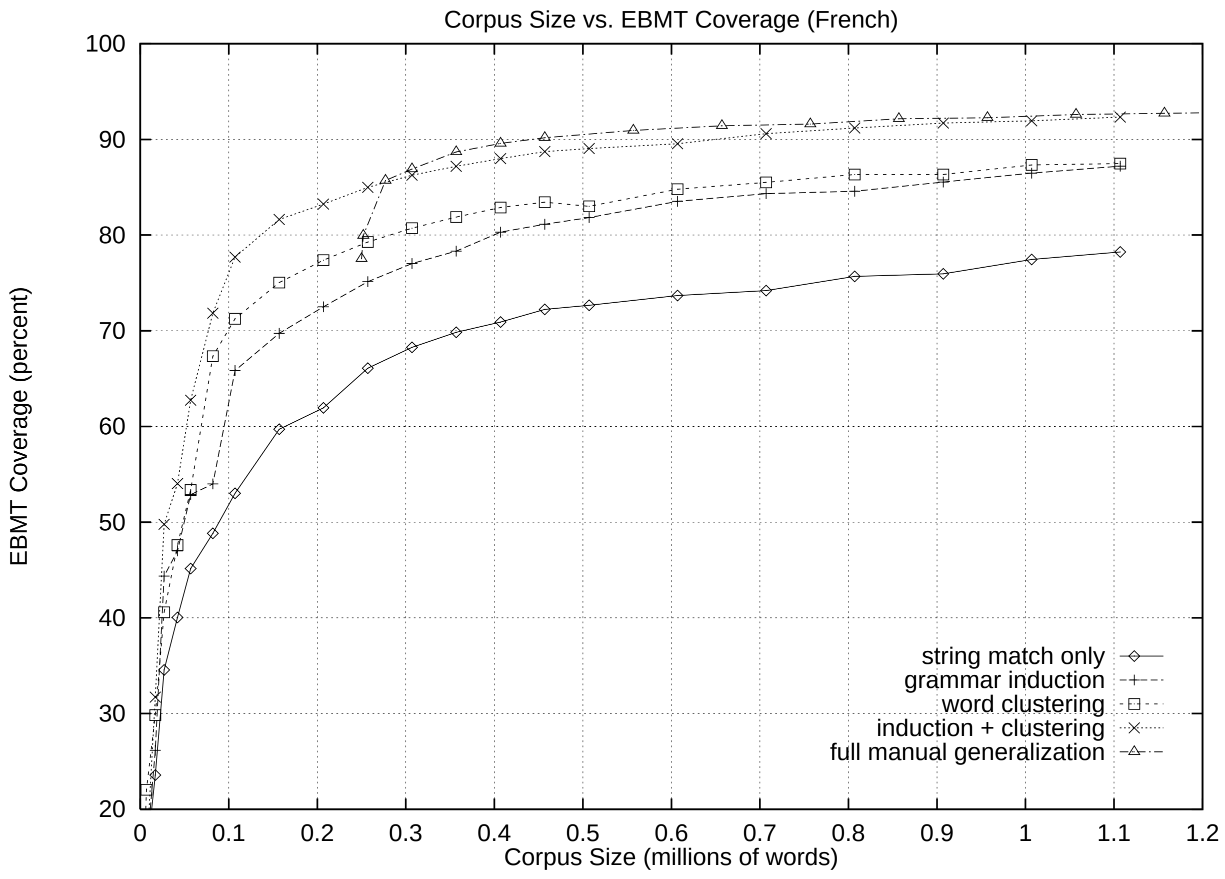
The clustering algorithm is also $O(n^2)$, but here n is the number of term vectors, i.e. the number of distinct bilingual word pairs, which grows more slowly than the number of words in the training text. Thus, the execution time for the complete process of induction plus clustering is slightly worse than quadratic in the size of the input.

7 Results

As shown in Figure 8, clustering and transfer-rule induction each outperformed simple string matching, and the combination substantially outperformed both. In fact, the combined algorithm exceeds the coverage of string matching trained on two million words of French-English parallel text with only 157,000 words of training data, more than a twelve-fold reduction in training data with no additional knowledge sources. For comparison, the best results (Brown, 1999) achieved using manually-created generalization information consisting of a large part-of-speech tagged bilingual dictionary and several hundred bilingual production rules based on those tags are shown in the graph as well. The automatic algorithms very nearly match the performance of the manual approach, without the quarter-million words of additional data in the dictionary and grammar rules used by manual generalization.

Similar results were obtained for Spanish-English (see Figure 9), where the combined algorithm had greater coverage with 104,000 words of training data than string matching on a one-million-word corpus.

Initial evaluation of the translation quality showed that most of the degradation in quality from grammar induction or the combination of grammar induction and clustering was due to misalignments, where the translation found by the system included extraneous words or omitted a portion of the true translation. The most egregious case was fairly easy to avert, simply by not applying rewriting rules to a new sentence pair if, after applying the rules, it has the form "<equivclass>" == "<equivclass> extra words" (or vice-versa). While the extra word(s) might be appropriate for the particular phrase, it is unlikely that they will be appropriate for all members of the equivalence class. This limitation reduced the coverage slightly, but substantially improved the quality of the EBMT system's output. A stricter consistency check than the current test of whether the bitext mapping positively rules out any part of the candidate translation would most likely further improve the translation quality, although the current system shows only minor degradation. Much of that degradation can be attributed to overgeneralization to cases where the



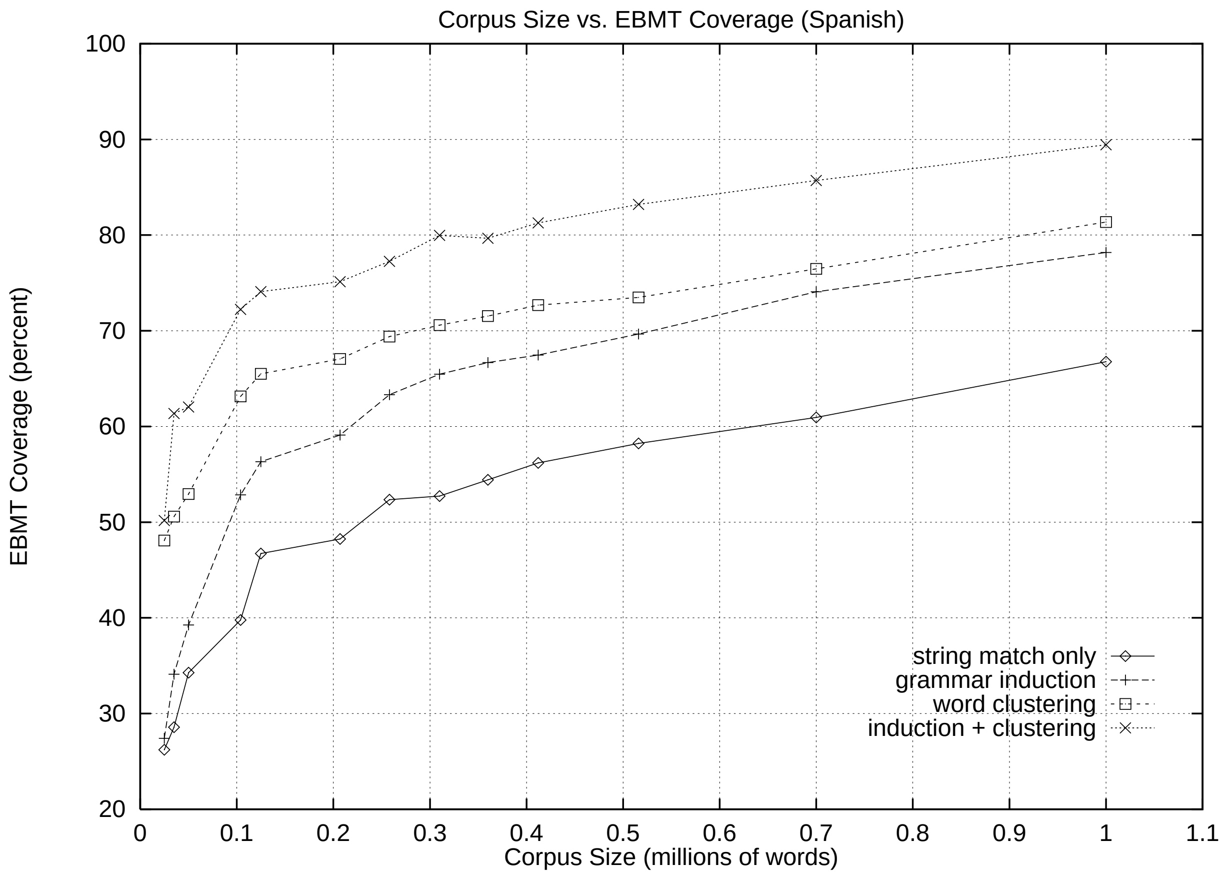
Corpus Size (words)	String Match Only	Induction	Clusters	Clust+Induc
107,000	53.01%	65.85%	71.25%	77.70%
157,000	59.72%	69.73%	75.05%	81.64%
207,000	61.94%	72.51%	77.38%	83.25%
307,000	68.28%	77.02%	80.73%	86.27%
1,107,000	75.26%	87.20%	87.33%	92.34%
2,007,000	80.14%	–	–	–

Figure 8: Performance Comparison (French)

usual default translation rule does not apply.

Some examples of the EBMT output are shown in Figures 7 and 11; the reference translation from the Hansard corpus is given for each. Figure 11 includes a very terse form of the output due to the limited space available; all matches which are contained within some longer match (from another example in the corpus) have been excluded, and only the best-scoring match from among those covering a particular phrase is shown. The actual output is several times longer, and includes some translations with better quality than the maximal matches actually shown. For each match, the alignment score (lower is better) and the translation are shown. Words in all-capital letters indicate where a single word matched an equivalence class generated by clustering, rather than the surface string. The sentence in Figure 7 was randomly selected from among the shorter sentences in the test set, while Figure 11 is the very last sentence in the test set.

Shortly before the final version of this paper was submitted, some changes were made to the EBMT system to improve its run-time efficiency. As part of those changes, some tweaks were made to the word-level alignment algorithm, including the generation of the correspondence table used for clustering as well as word-level alignment during translation. Those tweaks have resulted in somewhat paradoxical and as yet unexplained changes in the system's coverage (see Figure 10) – coverage for string matching and grammar induction dropped considerably, while clustering is greatly improved and the combination of induction and



Corpus Size (words)	String Match Only	Induction	Clusters	Clust+Induc
104,000	39.78%	52.86%	63.15%	72.23%
207,000	48.25%	59.11%	67.05%	75.14%
310,000	52.72%	67.47%	70.60%	79.98%
1,000,000	66.77%	78.19%	81.36%	89.44%

Figure 9: Performance Comparison (Spanish)

clustering remains almost unchanged[2]. Now that performance of clustering alone is so much closer to the performance of the combined algorithm, it becomes clear that the changes in the corpus produced by grammar induction interfere somewhat with clustering. With small training sets, the combined algorithm actually fares somewhat worse than clustering alone.

8 Conclusion

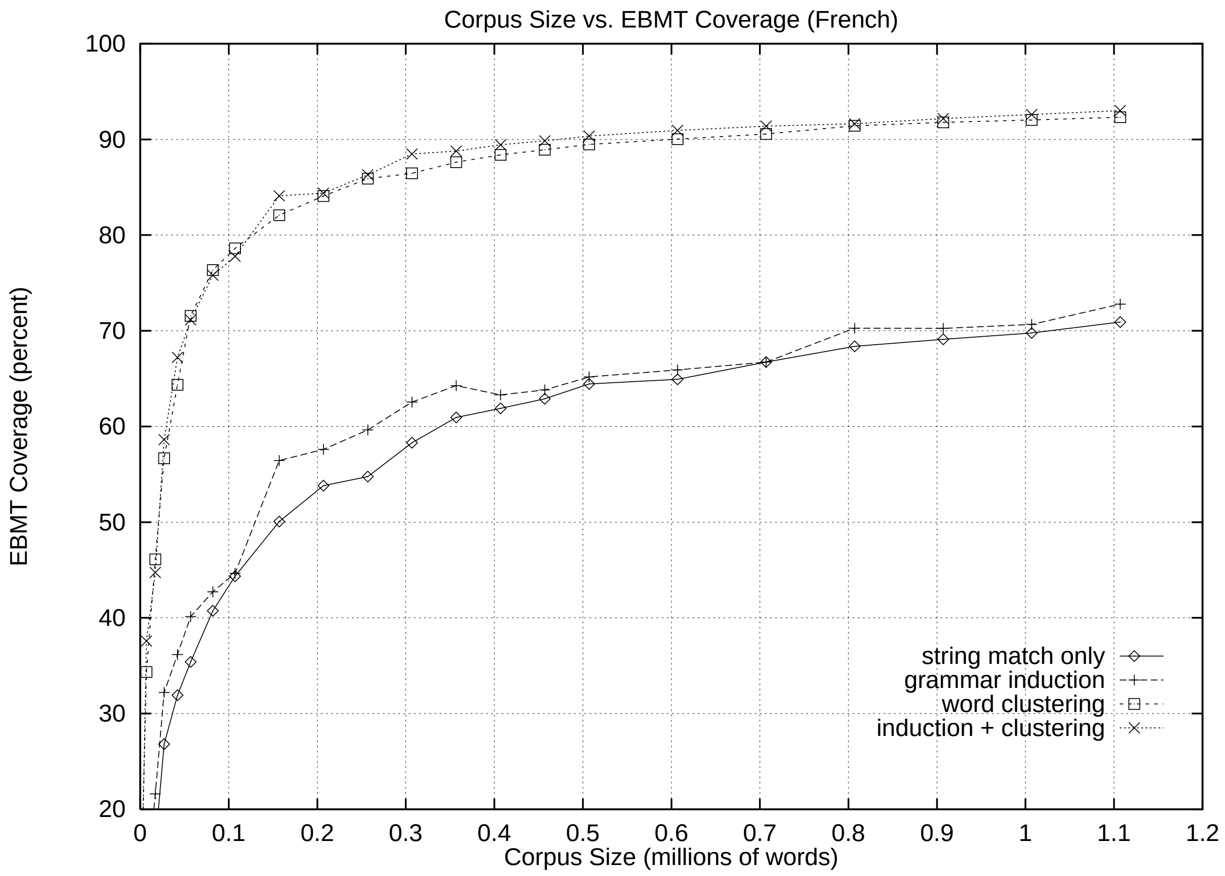
Experimental results indicate that combining transfer-rule induction in the style of (Cicekli, 2000) with the author's prior work on single-word clustering is beneficial, resulting in a system that outperforms either method used in isolation and dramatically reducing the amount of parallel training text required for a broad-coverage EBMT system. Coverage for a given amount of training text is increased with little or no impact on translation quality.

9 Ongoing and Future Work

The applicability of this approach has already been shown for two language pairs, but its effectiveness for very divergent language pairs remains to be demonstrated. Future experiments will include Mandarin-English as well as French-English and Spanish-English.

The transfer-rule induction can certainly be enhanced, for example by checking for common sequences within the dissimilar center portions, allowing them to be split even further. Currently, the learned equivalences tend to be fairly long; shorter phrases will be more general and more

[2]The restriction imposed on grammar induction to avoid bad translations slightly reduced the performance, and then the changes to the EBMT system somewhat improved coverage.



Corpus Size (words)	String Match Only	Clusters	Induction	Clust+Induc
107,000	44.35%	78.62%	44.66%	77.77%
207,000	53.81%	84.09%	57.61%	84.38%
307,000	58.31%	86.46%	62.56%	88.48%
1,107,000	70.92%	92.32%	72.79%	93.01%

Figure 10: Updated French Performance

likely to be matched in previously-unseen text during translation (improving coverage).

The word clustering parameters need to be tuned. Currently, the same parameters are used in conjunction with transfer-rule induction and in isolation. There is no *a priori* reason for the optimal settings in one case to be optimal for the other. In addition, the target of 2500 clusters was chosen arbitrarily and should be tuned for the best trade-off between quality and coverage.

The interference between transfer-rule induction and clustering noted during the most recent experimental runs should be isolated and, if possible, mitigated.

Finally, there is the possibility that adding a small amount of seed knowledge to the grammar induction process (similar to what has was previously done with single-word clustering) could substantially improve performance. Such seeding will require additional support in the software.

10 Acknowledgements

This research was supported primarily by the National Science Foundation's STIMULATE program (grant number 9618941). The author would like to thank the anonymous reviewers for their helpful comments, which have considerably improved the quality of this paper.

References

ARTFL Project. 1998. *ARTFL Project: French-English Dictionary.* Project for American and French Research on the Treasury of the French Language, University of Chicago. `http://-humanities.uchicago.edu/ARTFL.html`.

Ralf D. Brown. 1999. Adding Linguistic Knowledge to a Lexical Example-Based Translation

Comme il est 2 heures 03, la Chambre s'ajourne à 11 heures aujourd'hui, conformément au paragraphe 24(1) du Règlement.
It being 2.03 a.m., this House stands adjourned until later this day at ll a.m., pursuant to Standing Order 24(1).

String-Match Only, 1.1 million words:

Match	Score	Translation
comme il	1	He just a couple of
il est 2	1.275	it is 2
est 2 heures	1.275	is 2 o'clock
03 , la chambre	1	03 , the House
, la chambre se	1	, the House will
à 11 heures	0	at 11 a.m.
hui , conformément	1	today , pursuant
paragraphe 24 (1) de le règlement .	41.15	subsection 24 (1) of the rule .

Induction+Clustering, 1.1 million words:

Match	Score	Translation
comme	0	LIKE
il est 2 heures	1.825	IT IS 2 o'clock
03 , la chambre	0.75	03 , THE HOUSE
à 11 heures	1.11	to 11 a.m.
hui , conformément	1	today , pursuant
conformément au paragraphe 24 (1) de	2.65	ACCORDANCE TO SUBSECTION 24 (1) of
paragraphe 24 (1) de le règlement .	1.55	subsection 24 (1) OF STANDING .

Figure 11: Sample Translation 2

System. In *Proceedings of the Eighth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-99)*, pages 22–32, Chester, England, August. `http://www.cs.cmu.edu/~ralf/-papers.html`.

Ralf D. Brown. 2000. Automated Generalization of Translation Examples. In *Proceedings of the Eighteenth International Conference on Computational Linguistics (COLING-2000)*, pages 125–131.

Ilyas Cicekli and H. Altay Guvenir. 2001. Learning Translation Templates from Bilingual Translation Examples. *Applied Intelligence*. (to appear).

Ilyas Cicekli. 2000. Similarities and Differences. In *Proceedings of SCI2000*, pages 331–337, July. `http://www.cs.bilkent.edu.tr/-~ilyas/pubs.html`.

D. Graff and R. Finch. 1994. Multilingual Text Resources at the Linguistic Data Consortium. In *Proceedings of the 1994 ARPA Human Language Technology Workshop*. Morgan Kaufmann.

Linguistic Data Consortium. 1997. *Hansard Corpus of Parallel English and French*. Linguistic Data Consortium, December. `http://-www.ldc.upenn.edu/`.

Menno van Zaanen. 2000. ABL: Alignment-Based Learning. In *Proceedings of the Eighteenth International Conference on Computational Linguistics (COLING-2000)*, pages 961–967. `http://www.comp.leeds.ac.uk/menno/-docs/`.

Tony Veale and Andy Way. 1997. Gaijin: A Template-Driven Bootstrapping Approach to Example-Based Machine Translation. In *Proceedings of the NeMNLP'97, New Methods in Natural Language Processessing*, Sofia, Bulgaria, September. `http://-www.compapp.dcu.ie/~tonyv/papers/-gaijin.html`.

Ying Zhang, Ralf D. Brown, and Robert E. Frederking. 2001. Adapting an Example-Based Translation System to Chinese. In *Proceedings of the Human Language Technology Conference 2001*. (to appear) `http://www.hlt2001.org/`.

Inducing Translation Grammars from Bracketed Alignments

Michael Carl
Institut für Angewandte Informationsforschung,
Martin-Luther-Straße 14
66111 Saarbrücken, Germany,
carl@iai.uni-sb.de

Abstract

This paper presents an algorithm for generating and filtering an invertible and structural analogous translation grammar from bilingual aligned and linguistically bracketed texts. The algorithm is discussed in general terms and applied to German-English alignments. It is shown that the induction of structural analogous translation grammars can lead to disambiguation of meaning and correction of bracketing errors.

1 Introduction

Recent developments of Example-Based Machine Translation (EBMT) show a trend of inducing translation grammars from aligned texts. The aligned text, which serves as a basis for the induction of translation grammars, consists of a set of alignments, i.e. a set of source language/target language expressions[1] which are translations of each other. A translation grammar, in turn, consist of lexical transfer rules and generalizations.

The advantage of translation grammars is 1) that they can be induced off-line thus reducing computation time in the working-phase of the system and 2) that consistency of the alignments can be checked and handled more easily.

According to (Somers, 1999), EBMT systems differ in the number and the quality of resources used and in the way this knowledge is represented, stored and used for translation. However, EBMT systems differ also in the way generalizations are computed. In the framework of EBMT, a number of methods have been proposed for inducing translation grammars. In so-called "pure" EBMT systems, the only available knowledge resource is the aligned text itself (cf. (Block, 2000; Brown, 2001)), while in richer systems linguistic knowledge resources are used to a varying degree (cf. (Güvenir and Cicekli, 1998) and this present approach). In almost all cases where generalizations are induced from aligned texts, a set of two or more alignments are compared and suitable sub-sequences are replaced by variables.

Grammar induction from sets of monolingual examples have been studied since the 1960s. As early as 1967, Gold showed that even regular grammars cannot be exactly identified from positive examples alone. This insight is based on the fact that there will always be at least two possible candidate hypotheses, i) the language contains all possible sentences over a set of symbols $\sum^*$ and ii) the language corresponds to the set of provided examples alone. While i) is most compact and general, ii) is the most complex and least general hypothesis. A similar situation can be constructed for the induction of translation grammars from alignments: i) each symbol in the target language is a possible translation of every symbol in the source language and ii) the translation grammar corresponds to the set of alignments alone. Neither of these hypotheses seem attractive and so it seems desirable to find a compromise between the size of the hypothesis description and their generality. The compromise for the induction of translation grammars has been to look for collocations in the left-hand side (LHS) and

[1]These expressions are typically sentences but can be single words, clauses or paragraphs.

Generalization of Differences in Alignments (Güvenir and Cicekli, 1998)

Alignments:

1.	I took a	ticket	from Mary	↔	Mary'den bir	bilet	aldim
2.	I took a	pen	from Mary	↔	Mary'den bir	kalem	aldim

Generalization of Differences:

3.	I took a	$\mathcal{X}_1$	from Mary	↔	Mary'den bir	$\mathcal{Y}_1$	aldim

---

Generalization of Differences and Generalization of Similarities in Alignments (McTait, 2001)

Alignments:

4.	The commission	gave	the plan	up	↔	La commission	abandonna	le plan
5.	Our government	gave	all laws	up	↔	Notre gouvernement	abandonna	toutes les lois

Generalization of Differences:

6.	(...)	gave	(...)	up	↔	(...)	abandonna	(...)

Generalization of Similarities:

7.	The commission	(...)	the plan	(...)	↔	La commission	(...)	le plan
8.	Our government	(...)	all laws	(...)	↔	Notre gouvernement	(...)	toutes les lois

---

Generalization of Chunk Pairs in Alignments (Block, 2000):

Alignment:

9.	das	ist	was	Sie	wollen am	Mittwoch	morgen	zurückzukommen
	↕	↕	↕	↕		↕	↕	
	which	is	what	you	were wanting to come back	Wednesday	morning	

Chunk Pairs			Pattern Pairs						
das	↔	which							
ist	↔	is							
das ist	↔	which is	V ist	↔	V is	das V	↔	which V	
ist was	↔	is what	V was	↔	V what	ist V	↔	is V	
das ist was	↔	which is what	V ist was	↔	V is what	das V was	↔	which V what	...
ist was Sie	↔	is what you	V was Sie	↔	V what you	ist V Sie	↔	is V you	...
⋮		⋮	⋮		⋮	⋮		⋮	

Figure 1: Methods for Inducing Generalizations from Alignments

right-hand side (RHS) of the alignments and to assume these collocations to be translated compositionally.

In this paper, I first examine previous work on how generalizations and translation grammars are computed from alignments. I then present an original approach which assumes sub-sequences in the LHS and RHS of the alignment to be bracketed. These brackets are potential candidates for compositional translation. From the bracketed sequences, the algorithm extracts hypotheses of transfer rules, generates generalizations and filters a translation grammar.

The algorithm is applied to German-English alignments. It is shown how the meaning of ambiguous sentences can be disambiguated by assuming structural analogy of both language sides and how bracketing errors can be corrected.

2 Previous Work

A translation grammar consists of lexical transfer rules and generalizations. Generalizations are generated while inducing a translation grammar and differ from lexical transfer rules as they contain variables.

With respect to the way generalizations are computed one can distinguish between i) methods which generalize chunk translations in alignments (e.g. (Block, 2000; Boström, 1999) and this present approach), ii) methods which generalize differences in sets of alignments (Güvenir and Cicekli, 1998) and iii) methods which generalize similarities in sets of alignments (McTait, 2001).

In (Güvenir and Cicekli, 1998) the system searches for pairs of alignments where the LHS and the RHS show similar sequences. The differences are then substituted by variables while the identical subsequences remain as non-reduced tokens in the generalizations. An example of this is given in figure 1, top. The identical parts in alignments 1 and 2 remain unreduced in generalization 3, while the different parts (*ticket* $\leftrightarrow$ *bilet* and *pen* $\leftrightarrow$ *kalem*) are reduced to a variable. (McTait, 2001) generalizes alignments in a similar way as shown in generalization 6 in figure 1, middle. In addition to this, (McTait, 2001) also generalizes similarities in alignments which yields generalization 5 and 6 as a complement to generalization 4. In contrast to the approach proposed in this paper, (McTait, 2001) allows $m : n$ mappings as, for instance, in generalization 6.

A different approach is taken by (Block, 2000). In his approach, word translations in alignments are assigned probabilities based on a statistical word alignment tool. Statistical word alignments are indicated by vertical flashes ($\updownarrow$) in the alignment 9, figure 1, bottom. Based on this statistical word translation information, so-called chunk pairs are extracted. Subsequently, generalizations (i.e. pattern pairs) are generated for each chunk pair. This is achieved by replacing a chunk pair in another chunk pair by a variable if the former is a substrings in the latter chunk pair. An example is given in figure 1, bottom. The chunk pairs *das ist* $\leftrightarrow$ *which is* and *das* $\leftrightarrow$ *which* have been extracted from the alignment 7. The pattern pair *V ist* $\leftrightarrow$ *V is* was obtained by replacing the shorter chunk pair by the variable V. As a last step, trivial pattern pairs are filtered from the set of pattern pairs.

3 Overview

The algorithm I present in this paper is similar to the approach in (Block, 2000). First, chunk pairs (i.e. lexical transfer rules) are extracted from each alignment. Then generalizations are computed from the extracted transfer rules and from the alignments. Last an invertible translation grammar is filtered. In contrast to (Block, 2000) the present algorithm does not use a statistical word alignment tool. Instead, constituents in both language sides of the alignments are bracketed. From these brackets, hypotheses of lexical transfer rules are extracted by combining each LHS-bracket with each RHS-bracket. We currently use an extended version of the shallow parser KURD (Carl and Schmidt-Wigger, 1998) for bracketing, although knowledge-poor methods could similarly be used as, for instance, described in (Zaanen, 2000).

Generalization works similarly to the model described in (Block, 2000), but more than one pattern pair can be replaced in one generalization. Also, probabilities and weights are assigned to each of the rules. From the set of extracted and generated rules, a compositional, invertible and structural analogous translation grammar is filtered. A translation grammar is structural analogous, if it generates structural analogous derivations for LHS and RHS. Struc-

c_1:(a)	$\leftrightarrow$	(e')	c_5:(e)	$\leftrightarrow$	(e')	c_9:	(de)	$\leftrightarrow$ (e')	c_{13}:(cde)	$\leftrightarrow$	(e')
c_2:(a)	$\leftrightarrow$	(a')	c_6:(e)	$\leftrightarrow$	(a')	c_{10}:	(de)	$\leftrightarrow$ (a')	c_{14}:(cde)	$\leftrightarrow$	(a')
c_3:(a)	$\leftrightarrow$	$(a'b')$	c_7:(e)	$\leftrightarrow$	$(a'b')$	c_{11}:	(de)	$\leftrightarrow$ $(a'b')$	c_{15}:(cde)	$\leftrightarrow$	$(a'b')$
c_4:(a)	$\leftrightarrow$	$(a'b'c')$	c_8:(e)	$\leftrightarrow$	$(a'b'c')$	c_{12}:	(de)	$\leftrightarrow$ $(a'b'c')$	c_{16}:(cde)	$\leftrightarrow$	$(a'b'c')$

Figure 2: Set of lexical transfer rules C_1 extracted from a_1

tural analogy is achieved if each generalization has the same number of variables in its LHS and RHS. This implies that any LHS derivation tree has the same depth and branching as the RHS derivation tree, while the order of the branches can be permuted in both language sides.

In addition, the filtered translation grammar does not contain ambiguous transfer rules, i.e. each LHS and RHS in the filtered translation grammar occurs exactly once. This invertibility condition is achieved by encoding a minimal context which makes it unique in the filtered translation grammar.

In the remainder of this paper, I first describe how transfer rules are extracted from bracketed alignments and how probabilities and weights are assigned. Then I describe how a set of invertible transfer rules is filtered. Last I give examples for inducing German-English translation grammars and discuss potentials of the algorithm.

4 Complexity of Inducing Translation Templates

The program assumes a bilingual text P which consists of n alignments $a_1 \ldots a_n$. Each alignment a_i consists of a left-hand side (LHS) and a right-hand side (RHS). By means of a (shallow) parser, both language sides are independently bracketed (parsed) which results in a representation similar to the following[2]:

$$a_1 : (a)\ b\ (c(d(e))) \leftrightarrow (((a')b')c')\ d'\ (e')$$

Without further knowledge, one cannot know which of the LHS-brackets translate into its corresponding bracket in the RHS or whether a LHS-bracket has a RHS-translation at all. But it is assumed that if a LHS-bracket has a translation in the RHS of the same alignment then it would translate into exactly one bracket. As we have no knowledge which brackets to align, we assume that each LHS-bracket translates with the same probability into any RHS-bracket. For an alignment a_i we can therefore extract $p \times q$ lexical transfer rules $C_i : \{c_1 \ldots c_{p*q}\}$, where p is the number of LHS-brackets and q is the number of RHS-brackets. The set C_1 extracted from a_1 is shown in figure 2.

In a second step, translation templates (or generalizations) are induced from the alignments and from the extracted lexical transfer rules $c_1 \ldots c_{p*q}$. While a lexical transfer rule consists only of terminal symbols, a generalization contains variables (so-called reductions) in places where a shorter transfer rule matches a sub-sequence in the LHS and in the RHS. A generalization has thus at least one reduction in each language side and it has an equal number of reductions in the LHS and the RHS. As we assume a 1-to-1 mapping of the LHS and RHS-brackets each reduction in LHS is linked to exactly one reduction in RHS and vice versa.

From the transfer rule c_{16}, for instance, can be generated 4 different generalizations while from transfer rule c_{11} only one generalization can be generated. These generalizations are shown in figure 3. From the alignment a_1 can be generated 25 generalizations by substituting one or more transfer rules $c_1 \ldots c_{16}$.

More generally, from a transfer rule c_j which has p brackets in its LHS and q brackets in its RHS, an exponential number of generalizations $\#G_{c_j}$ can be generated:

$$\#G_{c_j} = \sum_{i=0}^{p} \binom{p}{i} * \binom{q}{i} = \binom{p+q}{p}$$

For instance, if we assume both, p and q to be 10 and none of the 10 brackets in either language side are included in another bracket (i.e. all brackets are top-level brackets), more than 180,000 different generalizations can be generated. This is a number far too big to be computed, as 10 brackets (i.e. constituents in the parsed sentence) is not many. Therefore,

[2]The letters "*abcde*" represent lemmas in LHS; the letters "$a'b'c'd'e'$" lemmas in the RHS. The brackets are also annotated with a phrasal tag and morpho-syntactic information. For the sake of simplicity, I will consider this information only in the last two sections.

G_{c_i}	Induced Generalization	$p(g_k)$	$w(g_k)$
$G_{c_{11}}:$	$\{\ g_1: \ (dA) \leftrightarrow (A'b')$	0.2	0.4
$G_{c_{16}}:$	$g_2: \ (cdA) \leftrightarrow (A'b'c')$	0.2	0.4
	$g_3: \ (cdA) \leftrightarrow (A'c')$	0.2	0.4
	$g_4: \ (cA) \leftrightarrow (A'b'c')$	0.2	0.4
	$g_5: \ (cA) \leftrightarrow (A'c')$	0.2	0.6

Figure 3: Set of generalizations $G_{c_{11}}$ and $G_{c_{16}}$ induced from transfer rules c_{11} and c_{16}

a number of heuristics is proposed for generating from the set of possible generalizations only those achieving the highest weight.

5 Probabilistic Translation Grammars

Before introducing heuristics, I would like to describe how probabilities and weights are assigned to lexical transfer rules and generalizations[3]. While inducing the translation grammar, each alignment a_i, each lexical transfer rule c_j and each generalization g_k, is linked to two sets.

$$\begin{aligned} a_i &\Rightarrow \{G_{a_i}, C_{a_i}\} \\ g_k &\Rightarrow \{R_{g_k}, C_{g_k}\} \\ c_j &\Rightarrow \{G_{c_j}, A_{c_j}\} \end{aligned}$$

Alignments a_i are associated with a set of lexical transfer rules C_{a_i} and a set of generalizations G_{a_i}. The lexical transfer rules in C_{a_i} have been extracted from alignment a_i while G_{a_i} contains generalizations of a_i. Each lexical transfer rule c_j is associated with a set A_{c_j} of alignments from which c_j has been extracted and a set of generalizations G_{c_j} which have been generated from c_j. Finally, each generalization g_k is associated with a set of lexical transfer rules C_{g_k} which have been replaced in the generalization (i.e. the daughters of the generalization g_k) and a set of references R_{g_k}. Since generalizations are generated from alignments and from the extracted lexical transfer rules, the set of references R_{g_k} may consist of alignments a_i and/or transfer rules c_j.

The probability of an alignment a_i is its frequency in the aligned text P divided by the number n of alignments in P.

[3]The notation used here is slightly different from a previous presentation in (Carl, 2001).

$$p(a_i) = \frac{f(a_i)}{n} \qquad (1)$$

The probability of a lexical transfer rule c_j is a function of the number of times c_j has been extracted from alignments a_i, $i = 1 \ldots n$, the cardinality $\#C_{a_i}$ of the set C_{a_i} and the size of P:

$$p(c_j) = \sum_{a_i \in A_{c_j}} \frac{1}{\sqrt{\#C_{a_i}} + n} \qquad (2)$$

The probability of a generalization g_k equals the maximum probability of the reference $r \in R_{g_k}$ from which g_k has been generated.

$$p(g_k) = max\{p(r \in R_{g_k})\} \qquad (3)$$

Based on these probabilities, a weight is computed for each a_i, c_j, g_k.

The weight $w(r_j)$ of a lexical transfer rule or an alignment equals the maximum weight of the generalization that has been generated from it.

$$w(r_j) = max\{w(g \in G_{r_j})\} \qquad (4)$$

In case no generalization can be generated from r_j: $w(r_j) = p(r_j)$. The weight of a generalization g_k equals the maximum of the probability of its reference $r_i \in R_{g_k}$ plus the sum of the weights of the lexical transfer rules which have been replaced in r_i to generate g_k. (i.e. $p(r_i)$ plus the sum of the daughters of g_k for reference r_i).

$$w(g_k) = max\{p(r_i \in R_{g_k}) + \sum_{c_i \in C_{g_k}} w(c_i)\} \qquad (5)$$

From this it follows that generalizations have higher weights if they contain i) more reductions or ii) if the reductions are to the highest extend compositional. As an example for achieving higher weights for more compositional generalizations, consider the following example.

As there are 16 lexical transfer rules extracted from alignment a_1 and assuming that $n = 1$, each of the rules in figure 2 has probability 0.2. A sub-sequence in c_{11} : $(de) \leftrightarrow (a'b')$ can be substituted by rule c_6 : $(e) \leftrightarrow (a')$. This substitution yields generalization g_1, as shown in figure 3. This generalization[4] is assigned the weight 0.4 (cf. equation 5). The same weight is assigned to the transfer rule c_{11} (cf. equation 4). Subsequently, when generalizing the longer rule c_{16} : $(cde) \leftrightarrow (a'b'c')$, a set of four generalizations G_{16} is generated from which g_5 is assigned the highest weight due to the compositional nature of the replaced daughter c_{11} : $(de) \leftrightarrow (a'b')$.

6 Generating Transfer Rules

Aligned texts $P : \{a_1 \dots a_n\}$ consist of $n \geq 1$ alignments. Each alignments $a_i,\ i = 1 \dots n$ is generalized in a sequential manner. For each a_i, first the $p \times q$ lexical transfer rules are extracted and sorted by the length of the shorter string LHS or RHS. Transfer rules $c_j \in C_{a_i}$ are then generalized starting with the shortest rule. The crucial points in the procedure **GenerateGrammar**() are lines 4 and 9. Extracting the set of lexical transfer requires a quadratic effort in the number of brackets, while generating G_{c_j} from c_j is exponential. To tackle this latter complexity, a version of the A* algorithm generates only a limited number of the highest weighted generalization:

```
1  GenerateGrammar($P$)
2  begin
3    for all $a_i \in P$:
4      extract lexical transfer rules $C_{a_i}$ from $a_i$;
5      for all $c_j \in C_{a_i}$ : $p(c_j) += 1/(\sqrt{\#C_{a_i}} + n)$;
6      add $a_i$ as $c_0$ to $C_{a_i}$;
7      sort $C_{a_i}$ by length of shorter LHS or RHS;
8      for each $c_j \in C_{a_i}$ starting with shortest $c$
9        generate $G_{c_j}$ from $c_j$
10       $w(c_j) = max\{w(g \in G_{c_j})\}$
11     end
12   end
13 end;
```

[4]The capital letters A, A', B and B' represent reductions of the replaced substrings where A maps into A' and B maps into B'.

In this way generalization of lexical transfer rules is reduced to $O(k * d)$ where k is the number of generalizations to be generated from c_j and d is the number of daughters in a generalization. In addition to this, a couple of parameters can be set to reduce the number of extracted transfer rules and generalizations:

- Transfer rules are only extracted where the difference in the number of words in LHS and RHS does not exceed a certain threshold. This constraint reflects the fact that generally the same number of (content) words appear on both sides of a translation.
- By the same token, transfer rules and generalizations are weighted by the difference of number of words in LHS and RHS.
- By means of a bilingual lexicon, transfer rules and generalizations may be assigned (high) priori weights.
- Only a limited number of highest weighted translation rules and generalizations is generated.
- A generalization can have a maximum number of reductions.
- Transfer rules are only asserted if their weights are above a threshold of an already existing, ambiguous transfer rule in the database.

Note that these parameters are suitable to reduce the number of generated generalizations and extracted lexical transfer rules.

7 Filtering Invertible Translation Grammars

From the set of alignments $P : \{a_1, \dots, a_n\}$ and their associated sets of generalizations $\{G_{a_1}, \dots, G_{a_n}\}$ an invertible translation grammar is filtered in a top down fashion. The aim here is to find a compositional set of transfer rules capable of reproducing the aligned text P in a most complete manner. To achieve this goal, translation ambiguities in the resulting grammar are avoided by including the smallest possible context which disambiguates each transfer rule. A transfer rule $\text{LHS}_2 \leftrightarrow \text{RHS}_2$

Default Translation Grammar

			$p(\cdot)$	$w(\cdot)$
e	$\leftrightarrow$	a'	0.2	0.2
a	$\leftrightarrow$	e'	0.2	0.2
dA	$\leftrightarrow$	$A'b'$	0.2	0.4
cA	$\leftrightarrow$	$A'c'$	0.2	0.6
AbB	$\leftrightarrow$	$A'd'B'$	1.0	1.8

Translation Grammar with Prior Lexical Knowledge

prior transfer rule $p(a \leftrightarrow a') = 0.5$

			$p(\cdot)$	$w(\cdot)$
a	$\leftrightarrow$	a'	0.50	0.50
e	$\leftrightarrow$	e'	0.24	0.24
$AbcdB$	$\leftrightarrow$	$A'b'c'd'B'$	1.00	1.74

Figure 4: Translation Grammar induced from alignment $(a)\ b\ (c(d(e))) \leftrightarrow (((a')b')c')\ d'\ (e')$

is ambiguous iff the translation grammar contains a different transfer rule $\text{LHS}_1 \leftrightarrow \text{RHS}_1$ where either LHS_1 equals LHS_2 or RHS_1 equals RHS_2. A translation grammar is invertible iff it contains no ambiguous transfer rules. In an invertible translation grammar, therefore, each LHS-string and each RHS-string occurs exactly once. Note that an invertible translation grammar is not deterministic. More than one translation can be generated from a source language string by differently bracketing the source string. However, each bracket has exactly one translation.

The alignments $a_i \in P$ are sorted by their weights and for each alignment starting with the highest weighted one, the function **FilterGrammar**(a_i) is called. The procedure **FilterGrammar()** prints the highest weighted generalizations of a_i and recursively prints their daughters. Less frequent rules, i.e. lower weighted rules, are likely to come along with more context. Only one generalization is printed for each alignment.

```
1  FilterGrammar($r_i$)
2  begin
3    if $G_{r_i}$ not empty
4      print $g_k$ : $max\{w(g_k \in G_{r_i})\}$
5      for all generalizations $g_m$ : delete $g_m$ \
         if((LHS$_m$ = LHS$_k$) or (RHS$_m$ = RHS$_k$)).
6      for all $c_j \in C_{g_k}$: FilterGrammar($c_j$)
7    else
8      print $r_i$
9      for all rules $c_m$: delete $c_m$ \
         if((LHS$_m$ = LHS$_i$ or (RHS$_m$ = RHS$_i$)).
10   end
11 end;
```

The procedure is called recursively in line 6 to print the highest weighted daughters c_j of generalization g_k.

Without prior knowledge of translation equivalences, the algorithm filters the most compositional, structural analogous translation grammar. A maximal compositional translation grammar generated and filtered from alignment a_1 is shown in figure 4, left. However, if the program is fed with prior knowledge of translation equivalences which do not - or only partially - support the structural equivalence of the bracketed alignments, a different translation grammar might be generated. Thus, the translation grammar in figure 4, right, is generated when providing the transfer rule $a \leftrightarrow a'$ with a prior probability of $p(a \leftrightarrow a') = 0.5$. Instead of the 16 lexical transfer rules in figure 2, only 10 transfer rules are extracted from alignment a_1 as $\{c_1, c_3, c_4, c_6, c_{10}, c_{14}\}$ are not consistent with the prior knowledge. The program tries to take this knowledge into account by filtering the most compositional translation grammar which is consistent with the data.

8 Ambiguous Bracketing

It is not always possible to obtain unambiguous brackets for every alignment. Ambiguous brackets might be due to (mal-) performance of the bracketing tool or to the implicit ambiguity of the sentence.

In the worst case, every subsequence in an alignment can be bracketed. There are maximally $\sum_{i=1}^{n} i$ different brackets in a sentence of length n. For an alignment with n tokens in LHS and m tokens in RHS, the number of extracted transfer rules amounts, thus, to:

$$\#C \;=\; \frac{(n^2+n)*(m^2+m)}{4}$$

While there are many different possible translation grammars that could be generated and filtered from this set, the most compositional grammar is very simple.

An example is given below. Assume all proper subsequences in the LHS and RHS of alignments a_2 and a_3 in figure 5 are bracketed. As both sides of the alignments have

$$\begin{array}{lccc} a_2 & abcd & \leftrightarrow & a'b'c'd' \\ a_3 & acbd & \leftrightarrow & c'a'b'd' \end{array}$$

Transfer Rules C_{a_2}

$$\left.\begin{array}{c} (a) \\ (b) \\ (c) \\ (d) \\ (ab) \\ (bc) \\ (cd) \\ (abc) \\ (bcd) \end{array}\right\} \times \left\{\begin{array}{l} (a') \\ (b') \\ (c') \\ (d') \\ (a'b') \\ (b'c') \\ (c'd') \\ (a'b'c') \\ (b'c'd') \end{array}\right.$$

Translation Grammar a_2

$$\begin{array}{ccc} (a) & \leftrightarrow & (a') \\ (b) & \leftrightarrow & (b') \\ (c) & \leftrightarrow & (c') \\ (d) & \leftrightarrow & (d') \\ (AB) & \leftrightarrow & (A'B') \end{array}$$

Transfer Rules C_{a_2}

$$\left.\begin{array}{c} (a) \\ (c) \\ (b) \\ (d) \\ (ac) \\ (cb) \\ (bd) \\ (acb) \\ (cbd) \end{array}\right\} \times \left\{\begin{array}{l} (c') \\ (a') \\ (b') \\ (d') \\ (c'a') \\ (a'b') \\ (b'd') \\ (c'a'b') \\ (a'b'd') \end{array}\right.$$

Possible transfer Rules for Translation Grammar a_3

$$\begin{array}{lccc} 1. & aA & \leftrightarrow & A'a' \\ 2. & Ac & \leftrightarrow & c'A' \\ 3. & ac & \leftrightarrow & c'a' \end{array}$$

Figure 5: Maximal Ambiguous Bracketing and Maximal Compositional Translation Grammar

four token and assuming that the entire alignments are not bracketed, this leads to $\#C_{a_2} = 9 \times 9 = 81$ transfer rules for alignment a_2 and $\#C_{a_3} = 9 \times 9 = 81$ transfer rules for alignment a_3. 16 of the transfer rules in C_{a_2} and C_{a_3} are identical such that in total 146 different transfer rules are generated[5]. In Figure 5 the LHS and RHS-brackets are plotted from which the 146 translation rules are generated. These rules and the alignments are then generalized and a translation grammar is filtered as described above. The most compositional translation grammar filtered for alignment a_2 consists of 5 translation rules as shown in figure 5, top right. For alignment a_3 only one additional translation rule is filtered. There are many possibilities to adding one transfer rule to translation grammar a_2 such that the grammar is still invertible and alignment a_3 can be generated. Three possible transfer rules are shown in figure 5 bottom, right.

From an invertible translation grammar, different translations may be obtained by applying a different subset and/or changing the order of transfer rules. Adding rule 2 to translation grammar a_2 increases the number of possible derivations which can be generated from $a'b'c'd'$ and, as a consequence of this, increases the number of possible translations:

$$\begin{array}{lcl} (a')(b')(c')(d') & \rightarrow & abcd \\ (a')(b')(c'(d')) & \rightarrow & abdc \end{array}$$

While bracketing a sentence in a different way may result in different translations for that sentence, in an invertible translation grammar each bracket and thus each transfer rule has one unique translation. To achieve higher reliability of the induced grammar, the function **FilterGrammar()** is, therefore, tuned to filter transfer rule 3 which reduces this kind of non-determinism.

9 Disambiguating Meaning

In the remaining part of this paper I investigate a few implications of the algorithm applied to German-English alignments. In this section I show that the meaning of an alignment can be disambiguated by filtering a structural analogous translation grammar. In the next section I show that the algorithm can also correct wrong brackets in alignments.

Consider the German-English alignment in figure 6. While there is only one non-overlapping sequence of brackets in the German LHS, the English translation has two interpretations. From the English sentence one cannot know whether *the block* is in *the box* or whether *the box* is on *the table*. Accordingly, two ambiguous brackets are generated: *(the block in the box)* and *(the box on the table)* which express these two interpretations. In the German translation, the relations of the phrases (i.e. the attachment of the PPs) is clarified by their cases. The dative *in der Kiste* expresses the location of the *Klotz*, while the accusative of *auf den Tisch* expresses the direction where the *Klotz* is placed. Assuming the same meaning of the German and the English sentences, *the block* is, thus, in *the box* and *John puts the block on the table.*

As discussed above, the algorithm filters transfer rules which show best structural analogy of the English and German brackets. When generating the translation grammar, the meaning of the English sentence is disambiguated ac-

[5]Note, however, that the program indexes each LHS and RHS only once, while the connections between LHS and RHS are realized through pointer. This amounts to 14 indexes in each language side of $C_{a_2 \cup a_3}$ and 2 * 146 connecting pointer.

(Hans)$_{\texttt{noun}}$	*stellt*	*((den Klotz)*$_{\texttt{dp}}$	*in*	*(der Kiste)*$_{\texttt{dp}}$*)*$_{\texttt{dp}}$	*auf*	*(den Tisch)*$_{\texttt{dp}}$
		$\leftrightarrow$				
(John)$_{\texttt{noun}}$	*puts*	*((the block)*$_{\texttt{dp}}$	*in*	*(the box)*$_{\texttt{dp}}$*)*$_{\texttt{dp}}$	*on*	*(the table)*$_{\texttt{dp}}$
(John)$_{\texttt{noun}}$	*puts*	*(the block)*$_{\texttt{dp}}$	*in*	*((the box)*$_{\texttt{dp}}$	*on*	*(the table)*$_{\texttt{dp}}$*)*$_{\texttt{dp}}$

Filtered Translation Grammar			$p(\cdot)$	$w(\cdot)$
(Tisch)$_{\texttt{noun}}$	$\leftrightarrow$	*(table)*$_{\texttt{noun}}$	0.50	0.50
(Kiste)$_{\texttt{noun}}$	$\leftrightarrow$	*(box)*$_{\texttt{noun}}$	0.10	0.10
(Klotz)$_{\texttt{noun}}$	$\leftrightarrow$	*(block)*$_{\texttt{noun}}$	0.10	0.10
(Hans)$_{\texttt{noun}}$	$\leftrightarrow$	*(John)*$_{\texttt{noun}}$	0.10	0.10
(art {noun}^{1}*)*$_{\texttt{dp}}$	$\leftrightarrow$	*(the* {noun}^{1}*)*$_{\texttt{dp}}$	0.10	0.60
({dp}^{1} *in* {dp}^{2}*)*$_{\texttt{noun}}$	$\leftrightarrow$	*(*{dp}^{1} *in* {dp}^{2}*)*$_{\texttt{noun}}$	0.10	1.30
({noun}^{1} *stellt* {dp}^{2} *auf* {dp}^{3}*)*	$\leftrightarrow$	*(*{noun}^{1} *puts* {dp}^{2} *on* {dp}^{3}*)*	1.00	3.00

Figure 6: Disambiguation of PP attachment

cording to the brackets of the German translation. The filtered translation grammar is shown in figure 6. Here we reproduce an experience of (Melamed, 2001):
"... the translation of a text into another language can be viewed as a detailed annotation of what that text means".
In this example the translation leads to a disambiguation of the English PP-attachment.

Note that the translation $(Tisch)_{\texttt{noun}} \leftrightarrow (table)_{\texttt{noun}}$ was provided with a prior probability of 0.5. The curely brackets in the translation grammar represent reductions previously marked by the capital letters A and B. The features[6] {noun} and {dp} are phrasal tags of the transfer rules and brackets which also appear in reductions of generalizations. The superscriped indices 1, 2 and 3 in the LHS and RHS-reductions denote the linking of the reductions. Note also, that the algorithm works on sequences of lemmas, rather than on the surface forms of the words. The lemma *art* in the German side of the transfer rule represents the surface forms of the articles *der, die, das, dem, den, des.*

Disambiguation of meanings through structural analogy does not work in cases where both language sides are equally ambiguous as, for instance, in the famous example:

Hans sieht den Mann mit dem Fernglas $\leftrightarrow$
John sees the man with the binoculars

1. ({noun}^{1} *sieht* {dp}^{2}) $\leftrightarrow$
   ({noun}^{1} *sees* {dp}^{2})
2. ({noun}^{1} *sieht* {dp}^{2} {pp}^{3}) $\leftrightarrow$
   ({noun}^{1} *sees* {dp}^{2} {pp}^{3})

Given the alignment is appropriately bracketed, at least two top-level generalizations are generated.

In case there is no conflicting evidence in the aligned text, the program is likely to filter the more compositional generalization 1 which assumes an attachment of the PPs to the close-by object rather than an attachment to the subject as in generalization 2.

10 Correcting Bracketing Errors

Meaning of sentences and their corresponding brackets can be ambiguous as shown above. Bracketing can also be be misguiding or wrong. However, if most of the brackets in a set of alignments are correct, the algorithm is likely to extract a correct translation grammar. Wrong brackets in alignments are ignored when filtering a compositional and structural analogous translation grammar from the alignments.

In the alignments in figure 7, only nouns and adjective-noun combinations (noun) as well as simple and complex determiner phrases (dp) are bracketed. While most of the brackets are correct, some of them are linguistically not justified. These unjustified brackets are purposely added to see how far the grammar induction can tackle such cases.

In alignments 1 and 6 all brackets in the German and English side are correct. Alignment 1 has a bracketed noun and one simple determiner phrase in either language side, while in alignment 6, complex embedded phrases are

[6]Nouns and adjective-noun combinations are marked noun, simple and complex determiner phrases are marked dp.

1. (Peter)$_{\text{noun}}$ lässt (das (Haus)$_{\text{noun}}$)$_{\text{dp}}$ ungern unbewohnt. $\leftrightarrow$
   (Peter)$_{\text{noun}}$ doesn't like (the (house)$_{\text{noun}}$)$_{\text{dp}}$ to be left empty.
2. Man hat ((Peter)$_{\text{noun}}$ (das (Rad)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{noun}}$ wechseln sehen. $\leftrightarrow$
   (Peter)$_{\text{noun}}$ was seen changing (the (wheel)$_{\text{noun}}$)$_{\text{dp}}$.
3. ((Peter)$_{\text{noun}}$ (der (Wolf)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{noun}}$ hält (ein (Referendum)$_{\text{noun}}$)$_{\text{dp}}$ für überflüssig. $\leftrightarrow$
   ((Peter)$_{\text{noun}}$ (the (wolf)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{noun}}$ thinks (a (referendum)$_{\text{noun}}$)$_{\text{dp}}$ unnecessary.
4. Man hat ((Peter)$_{\text{noun}}$ in (das (Haus)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{noun}}$ gehen sehen. $\leftrightarrow$
   (Peter)$_{\text{noun}}$ was seen to enter (the (house)$_{\text{noun}}$)$_{\text{dp}}$.
5. Es ist bekannt, dass ((Peter)$_{\text{noun}}$ (ein (Lügner)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{noun}}$ ist. $\leftrightarrow$
   (Peter)$_{\text{noun}}$ is known to be (a (liar)$_{\text{noun}}$)$_{\text{dp}}$.
6. Von (Peter)$_{\text{noun}}$ wird erwartet, dass er mit ((den (Gesetzen)$_{\text{noun}}$)$_{\text{dp}}$ (seines (eigenen (Landes)$_{\text{noun}}$)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{dp}}$ vertraut ist. $\leftrightarrow$
   (Peter)$_{\text{noun}}$ is supposed to know ((the (laws)$_{\text{noun}}$)$_{\text{dp}}$ of (his (own (country)$_{\text{noun}}$)$_{\text{noun}}$)$_{\text{dp}}$)$_{\text{dp}}$.

$p(\cdot)$	$w(\cdot)$			
		Translation Grammar filtered from Alignment a_3		
0.14	1.48	({noun}^{1} hält {dp}^{2} für überflüssig.)	$\leftrightarrow$	({noun}^{1} thinks {dp}^{2} unnecessary.)
0.11	0.22	(ein {noun}^{1})$_{\text{dp}}$	$\leftrightarrow$	(a {noun}^{1})$_{\text{dp}}$
0.11	0.11	(Referendum)$_{\text{noun}}$	$\leftrightarrow$	(referendum)$_{\text{noun}}$
0.08	1.12	({noun}^{1} {dp}^{2})$_{\text{noun}}$	$\leftrightarrow$	({noun}^{1} {dp}^{2})$_{\text{noun}}$
0.22	0.44	(art {noun}^{1})$_{\text{dp}}$	$\leftrightarrow$	(the {noun}^{1})$_{\text{dp}}$
0.08	0.08	(Wolf)$_{\text{noun}}$	$\leftrightarrow$	(wolf)$_{\text{noun}}$
0.60	0.60	(Peter)$_{\text{noun}}$	$\leftrightarrow$	(Peter)$_{\text{noun}}$
		Translation Grammar filtered from Alignment a_6		
0.13	1.47	Von {noun}^{1} wird erwartet, dass er mit {dp}^{2} vertraut ist.)	$\leftrightarrow$	({noun}^{1} is supposed to know {dp}^{2}.)
0.07	0.75	({dp}^{1} {dp}^{2})$_{\text{dp}}$	$\leftrightarrow$	({dp}^{1} of {dp}^{2})$_{\text{dp}}$
0.08	0.08	(Gesetz)$_{\text{noun}}$	$\leftrightarrow$	(law)$_{\text{noun}}$
0.08	0.23	(sein {noun^1})$_{\text{dp}}$	$\leftrightarrow$	(his {noun^1})$_{\text{dp}}$
0.08	0.15	(eigen {noun}^{1})$_{\text{noun}}$	$\leftrightarrow$	(own {noun}^{1})$_{\text{noun}}$
0.08	0.08	(Land)$_{\text{noun}}$	$\leftrightarrow$	(country)$_{\text{noun}}$
		Translation Grammar filtered from Alignment a_4		
0.15	1.22	(Man hat {noun}^{1} in {dp}^{2} gehen sehen.)	$\leftrightarrow$	({noun}^{1} was seen to enter {dp}^{2}.)
0.22	0.22	(Haus)$_{\text{noun}}$	$\leftrightarrow$	(house)$_{\text{noun}}$
		Translation Grammar filtered from Alignment a_2		
0.15	1.21	(Man hat {noun}^{1} {dp}^{2} wechseln sehen.)	$\leftrightarrow$	({noun}^{1} was seen changing {dp}^{2}.)
0.11	0.11	(Rad)$_{\text{noun}}$	$\leftrightarrow$	(wheel)$_{\text{noun}}$
		Translation Grammar filtered from Alignment a_1		
0.13	1.20	({noun}^{1} lässt {dp}^{2} ungern unbewohnt.)	$\leftrightarrow$	({noun}^{1} doesn't like {dp}^{2} to be left empty.)
		Translation Grammar filtered from Alignment a_5		
0.14	0.99	(Es ist bekannt, dass {noun}^{1} {dp}^{2} ist.)	$\leftrightarrow$	({noun}^{1} is known to be {dp}^{2}.)
0.11	0.11	(Lügner)$_{\text{noun}}$	$\leftrightarrow$	(liar)$_{\text{noun}}$

Figure 7: Wrong Brackets in Alignments and Correct Translation Grammar

bracketed. Note that the structure of the brackets in the LHS and RHS of these alignments are identical.

In alignment 2, the bracket *(Peter das Rad)* is wrong as *Peter* is the object of the verb *sehen* and in the same time the subject of *wechseln* while *das Rad* is the accusative object of *wechseln*. The problem here is that *das Rad* is ambiguous wrt. nominative or accusative. Similar brackets in alignment 3, *(Peter der Wolf)* are correct as the nominative *der Wolf* modifies the subject *Peter*.

The distinction between these two cases is difficult to determine for a partial parser as it requires knowledge about valency of verbs and the type of the sentence (i.e. verb middle or verb final). However, if we look at the English brackets of these alignments, one sees that *Peter* and *the wheel* are discontinuous in alignment 2 while the bracket *(Peter the wolf)* in alignment 3 show the same structure as German *(Peter der Wolf)*. Since the algorithm tries to filter most compositional, structural analogous translation grammars, the wrong German bracket in alignment 2 is disregarded. As shown in the lower part in figure 7, two generalizations are filtered for these alignments such that *(Peter)* $\leftrightarrow$ *(Peter)* and *(das Rad)* $\leftrightarrow$ *(the wheel)* are reduced into two reductions and *(Peter der Wolf)* $\leftrightarrow$ *(Peter the wolf)* is reduced into one reduction.

Hence, while correct brackets in one language might sometimes be difficult to determine, the brackets of the translation can be helpful to find their mutual analogous interpretation.

Two similar cases are shown in alignments 4 and 5. Here, the brackets *(Peter in das Haus)* and *(Peter ein Lügner)* are wrong. The algorithm overrules the wrong German brackets since their English translations are discontinuous. Instead most compositional, structural analogous generalizations are filtered by taking into consideration the brackets of the set of alignments as a whole and their sequences of aligned lexemes.

11 Conclusion

In this paper, an algorithm for sub-sentential alignment of aligned texts is discussed. The algorithm assumes both language sides to be linguistically bracketed. From the brackets, hypotheses of lexical transfer rules are extracted and a set of generalizations is generated. Each of these transfer rules is assigned a probability and a weight. From these rules, the algorithm, filters an invertible, structural analogous translation grammar. Implications and potentials of the approach are examined on a general level and a few examples applied to German-English alignments are presented in more detail.

References

Hans Ulrich Block. 2000. Example-Based Incremental Synchronous Interpretation. In *(Wahlster (ed.), 2000)*, pages 411–417.

Henrik Boström. 1999. Induction of Recursive Transfer Rules. In *Learning Language in Logic (LLL) Workshop*, Bled, Slovenia.

Ralph Brown. 2001. Transfer-Rule Induction for Example-Based Translation. In *Machine Translation Workshop of MT-Summit VIII.*

Michael Carl and Antje Schmidt-Wigger. 1998. Shallow Postmorphological Processing with KURD. In *Proceedings of NeMLaP3/CoNLL98*, pages 257–265, Sydney.

Michael Carl. 2001. Inducing probabilistic invertible translation grammars from aligned texts. In *CoNLL 2001.*

Halil Altay Güvenir and Ilyas Cicekli. 1998. Learning Translation Templates from Examples. *Information Systems*, 23(6):353–363.

Kevin McTait. 2001. Linguistic Knowledge and Complexity in an EBMT System Based on Translation Examples. In *Machine Translation Workshop of MT-Summit VIII.*

Dan I. Melamed. 2001. *Empirical Methods for Exploiting Parallel Texts.* MIT Press, Cambridge, MA.

Harold Somers. 1999. Review Article: Example-based Machine Translation. *Machine Translation*, 14(2):113–157.

Wolfgang Wahlster (ed.). 2000. *Verbmobil: Foundations of Speech-to-Speech Translation.* Springer, Heidelberg.

Menno van Zaanen. 2000. ABL: Alignment-Based Learning. In *COLING-2000*, pages 961–967.

Linguistic Knowledge and Complexity in an EBMT System Based on Translation Patterns

Kevin McTait

Department of Language Engineering
UMIST
Manchester, PO BOX 88, M60 1QD, UK
k.mctait@stud.umist.ac.uk

Abstract

An approach to Example-Based Machine Translation is presented which operates by extracting translation patterns from a bilingual corpus aligned at the level of the sentence. This is carried out using a language-neutral recursive machine-learning algorithm based on the principle of similar distributions of strings. The translation patterns extracted represent generalisations of sentences that are translations of each other and, to some extent, resemble transfer rules but with fewer constraints. The strings and variables, of which translations patterns are composed, are aligned in order to provide a more refined bilingual knowledge source, necessary for the recombination phase. A non-structural approach based on surface forms is error prone and liable to produce translation patterns that are false translations. Such errors are highlighted and solutions are proposed by the addition of external linguistic resources, namely morphological analysis and part-of-speech tagging. The amount of linguistic resources added has consequences for computational complexity and portability.

Introduction

A number of example-based machine translation (EBMT) systems operate by extracting and recombining translation patterns or templates from bilingual texts (Kaji et al, 1992; Güvenir & Cicekli, 1998; Brown 1999; Carl 1999; McTait & Trujillo, 1999). Translation patterns represent generalisations of sentences that are translations of each other in that various sequences of one or more words are replaced by variables, possibly with the alignments between word sequences and/or variables made explicit.

In this approach, which builds upon and improves that of McTait & Trujillo (1999), translation patterns are extracted from a bilingual corpus aligned at the level of the sentence. They are extracted by means of a language-neutral recursive machine-learning algorithm based on the principle of similar distributions of strings: source language (SL) and target language (TL) strings that co-occur in the same 2 (or more) sentence pairs of a bilingual corpus are likely to be translations of each other. The SL and TL strings that make up the translation patterns are aligned so that they provide not only sentential patterns of translation, but also a more refined bilingual knowledge source representing word / phrasal translations, necessary for the recombination phase, where TL translations are produced. Since the variables also represent strings, they too are aligned. Figure 1 is an example of a simple translation pattern indicating how a sentence in English containing *give…up*, may be translated by a sentence in French containing *abandonner*.

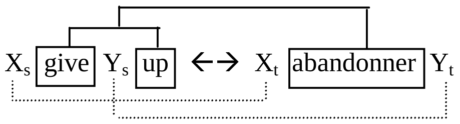


Figure 1: A Simple Translation Pattern

The translation pattern in figure 1 contains not only simple bijective or 1:1 alignments between text fragments or variables, but also non-bijective alignment types, such as the 2:1 alignment between *give…up* and *abandonner*. The ability to efficiently compute bijective and non-bijective alignments, as well as long distance dependencies, is conducive to more accurately describing translation phenomena.

Translation patterns are extracted, and the text fragments of which they are composed aligned, when the data is sparse, since strings only need to co-occur in a minimum of 2 sentence pairs. Furthermore, language-neutral techniques, such as cognates and bilingual lexical distribution, are used to align the strings or text fragments of which the patterns are composed.

The translation patterns extracted resemble, to some extent, transfer rules within a Rule-Based Machine Translation (RBMT) system, but with fewer constraints. A linear approach based on the distributions of surface forms within a corpus is liable to the extraction of translation patterns that are false translations. Solutions to this phenomenon are proposed by the addition of external linguistic knowledge sources such as morphological analysis and part-of-speech (POS) tagging. Figure 2 depicts the system architecture. The dotted lines indicate that the use of the knowledge sources is optional.

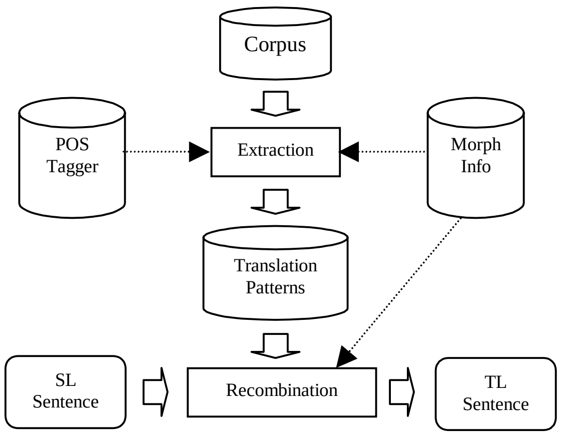


Figure 2: System Architecture

The addition of linguistic resources is intended to improve both accuracy and recall. However, their addition has consequences for portability and computational complexity. The more resources required, the less portable and the more complex the system.

This paper outlines the translation pattern extraction algorithm along with the corresponding recombination step where, given SL input sentences, TL translations are produced. The problems associated with a non-structural language-neutral approach are highlighted with solutions proposed involving the addition of external linguistic knowledge sources. Three variants of this approach are then ready for comparison, each with varying amounts of linguistic knowledge incorporated: i) the language-neutral approach based on surface forms, ii) the approach augmented to include morphological analysis and iii) the approach augmented to include both morphological analysis and POS tagging. Their performance is evaluated and compared, as is their complexity.

1 Existing Approaches

The concept of EBMT based on the extraction and recombination of translation patterns can be placed somewhere between 'traditional' linear EBMT - where the TL equivalents of overlapping partial exact matches of the SL input are computed dynamically and recombined (Nirenburg et al, 1993; Somers et al, 1994) - and systems that extract patterns that bear more resemblance to structural transfer rules (Kaji et al, 1992; Maruyama & Watanabe, 1992; Watanabe, 1995).

The extraction of translation patterns is typically reliant on the ability to generalise pairs of sentences in a corpus that are translations of each other. One method of classifying such systems is the method by which such generalisations are achieved and what constraints, if any, apply as a result. The broadest categorisation to make is those that use external linguistic resources and those that do not. However, distinctions may be made as to how and what external knowledge sources are used to generalise translation examples.

An approach that makes use of significant resources is Kaji et al. (1992). They use an English-Japanese bilingual dictionary and parsers to find correspondences at the phrase-structure level between two sentences that are translations of each other. These structures are then replaced by variables to produce translation patterns, similar to that in figure 1, except that the variables contain syntactic and possibly semantic constraints, due to the use of a thesaurus. The translation patterns described in Watanabe (1993) make use of a complex data structure involving a combination of lexical mappings and mappings between dependency structures, as is the case for the pattern-based context-free grammar rules found in Takeda (1996). Carl (1999) makes use of rich morphological analysis, enabling shallow parsing of the corpus to allow for the percolation of morpho-syntactic constraints in derivation trees. As Matsumoto & Kitamura (1995) show, it is also possible to generalise sentence pairs by replacing semantically similar words or dependency structures by means of a thesaurus.

Brown (1999) replaces certain strings denoting numbers, weekdays, country names etc. by an equivalence-class name, as well as including linguistic information such as number and gender. As Brown successfully shows, the level of abstraction or generalisation has consequences for coverage and accuracy.

Furuse & Iida (1992) describe three types of translation examples. The first type (2a) consists of literal examples, the second (2b) consists of a sentence pair with words replaced by variables and the third type (2c) are grammatical examples or context-sensitive rewrite rules, in effect, transfer rules of the kind found in traditional RBMT systems. The second type, despite its simplicity, most closely represents the translation patterns produced in this approach.

(2a) Sochira ni okeru ←→ We will send it to you
(2b) *X* o onegai shimasu ←→ may I speak to the *X′*
(2c) N_1 N_2 N_3 ←→ N_2 N_3 for N_1
N_1 = sanka / participation, N_2 = mōshikomi / application, N_3 = yōshi / form

Language-neutral techniques of extracting translation patterns are based on analogical reasoning (Güvenir & Cicekli, 1998; Malavazos & Piperidis, 2000) or inductive learning with genetic algorithms (Echizen-ya et al, 2000). The general principle applied is that given two sentence pairs in a corpus, the orthographically similar parts of the two SL sentences correspond to the orthographically similar parts of the two TL sentences. Similarly, the differing parts of the two SL sentences correspond to the differing parts of the TL sentences. The differences are replaced by variables to generalise the sentence pair. Highly inflective, or worse agglutinative, languages require an amount of linguistic pre-processing. In the case of Turkish, Güvenir & Cicekli (1998) use morphological analysis to alleviate orthographical differences.

Once generalisations of translation examples have been made, the SL and TL text fragments of which they are composed are generally aligned. In the case of the translation patterns of type (2b) in Furuse & Iida (1992) and also those of Carl (1999), there is no alignment problem to be solved since there are only *single* 1:1 or bijective mappings between strings or variables. In the case of Güvenir & Cicekli (1998), multiple 1:1 alignments in a translation template are solved by finding unambiguous or previously solved instances of the alignments in question from other translation templates. Few, if any, of the existing approaches cater for the fact that translation phenomena are not always bijective and that translation relations of a nature other than 1:1 exist i.e. the 2:1 relationship in figure 1.

The statistical models of Brown et al. (1993) cater for such relationships. However, they are computationally expensive, require large amounts of training data, rule out effective treatment of low-frequency words and are limited to unidirectional word-to-word translation models, thus ignoring the natural structuring of sentences into phrases. Later approaches (Dagan et al 1993; Wang & Waibel, 1998) address some, but not all, of these issues. The problem of aligning text fragments in translation examples is related to the bilingual vocabulary alignment problem. This includes words, terms and collocations (see Fung & McKeown (1997) and also Somers (1998) for an overview and bibliography). Generally, language-neutral or statistical vocabulary alignment techniques, based on distributions of word forms, are limited to computing 1:1 alignment patterns, again with large amounts of training data required.

2 Extraction & Recombination

2.1 Translation Patterns

A translation pattern can be defined formally as a 4-tuple *{S, T, A_f, A_v}*. *S* (*T*) represents a sequence of SL (TL) subsentential text fragments, separated by SL (TL) variables which represent subsentential text fragments (a subsentential text fragment is a series of one or more lexical items or tokens). In *S*, there can be any number *p* (*p>0*) of SL text fragments (F_p) with *p*, *p+1*, *p–1* SL variables (V_p). In *T*, there can also be any number *q* (*q>0*) of TL text fragments (F_q) with *q*, *q+1*, *q–1* TL variables (V_q). One possible configuration is depicted in figure 3.

$$F_1, V_1, F_2, V_2 ... F_p, V_p \leftrightarrow F_1, V_1, F_2, V_2 ... F_q, V_q$$

Figure 3: Possible Configuration of S and T

A_f represents the global alignment of text fragments between *S* and *T*, while A_v represents the global alignment of variables between *S* and *T*. The global alignment of the text fragments is represented as a set of local alignments *{<A,B>$_1$,<A,B>$_2$…<A,B>$_k$}*, where each local alignment is represented as a pair *<A,B>*. *A (B)* represents pointers to zero or more SL (TL) text fragments according to the local alignment patterns stipulated by the sequence comparison algorithm (section 2.2.3). The global alignment of the variables A_v is represented analogously.

2.2 Extracting Translation Patterns

The input to the translation-pattern extraction phase is a bilingual corpus aligned at the level of the sentence. The output is a set of translation patterns. The algorithm is language-neutral in nature and operates on the simple principles of string co-occurrence and frequency thresholds: possibly discontinuous pairs of SL and TL strings that co-occur in a minimum of 2 translation examples are likely to be translations of each other. Since strings are only required to co-occur a minimum of twice (frequency threshold), the algorithm is useful in instances of sparse data. However, the frequency threshold can be increased to improve the accuracy of the patterns (McTait & Trujillo, 1999). This section provides a highly simplified example, using the corpus in (3).

(3) The commission **gave** the plan **up** ←→
La commission **abandonna** le plan

Our government **gave** all laws **up** ←→
Notre gouvernement **abandonna** toutes les lois

2.2.1 Monolingual Phase

This stage is applied independently to the SL and TL sentences of the corpus. Lexical items (tokens) that occur in a minimum of 2 sentences are retrieved, together with a record of the sentences in which they were found: (4a) for the SL and (4b) for the TL.

(4a) *(gave)[1,2], (up)[1,2]*
(4b) *(abandonna) [1,2]*

The lexical items are allowed to combine to form longer word combinations (or *collocations*) constrained only by the sentences from which they were retrieved. The lexical items combine recursively to form a tree-like data structure of collocations. Each lexical item is tested to see if it can combine with the daughters of the root node and if so, recursively with each subsequent daughter, as long as there is an intersection of at least 2 sentence IDs (this enforces string co-occurrence in 2 or more sentences). The result is a tree of collocations of increasing length but decreasing frequency. The leaves become the most informative parts of the tree and are collected at the end of this phase. The longest provide more context and hence there is less chance of ambiguity.

As an example (figure 4), the SL lexical item *gave* is added to the root node (the integers denote the sentence IDs). The lexical item *up* is tested to see if it can combine with it since it is now a daughter of the root node. Since *gave* and *up* have an intersection of two sentence IDs, they are allowed to combine and form a new collocation node (*gave/up*). The TL lexical item is added to a separate tree and remains as in (4b) since there are no further TL lexical items with which it can combine.

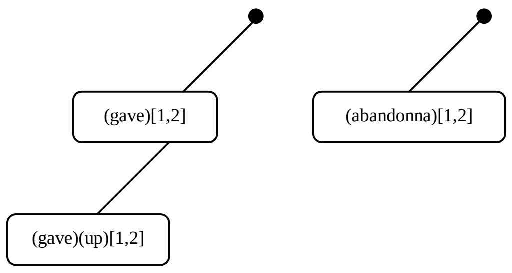


Figure 4: Collocation Trees

2.2.2 Bilingual Phase

SL and TL collocations are equated by simple co-occurrence criteria to form translation patterns: SL collocations that have exactly the same sentence IDs as TL collocations are considered to be translations of each other. This ensures that the patterns contain lexical items retrieved from the same sentences. The leaf-node collocations in figure 4 are equated to form the translation pattern (5). A translation pattern is formed from lexical items in 2 (or more) sentences, therefore its word order is determined from either of those sentences in the corpus. The discontinuities between the strings in (5) are represented as ellipses and denote variables.

(5) (…) gave (…) up ←→ (…) abandonna (…)

Translation patterns are not formed from inner leaves of the collocation trees since they would form patterns that are subsets of patterns from the leaf nodes. This would make them spurious. They also introduce ambiguity in that the effectiveness of EBMT lies in the retrieval of the longest possible matching segments. Furthermore, they would frequently be inaccurate. For example, if the node *gave[1,2]* were equated with *abandonna[1,2]* an incorrect pattern would be produced.

It is intuitive that if the text fragments - *F* in figure 3 - that make up the translation patterns are translations of each other, then the discontinuities or variables - *V* in figure 3 - that occur between them must also be translations. Since translation patterns are made up of lexical items from at least two translation examples, at least two *complement* translation patterns (6a) and (6b) are formed. They are created simply as the inverse of regular translation patterns and may contain lexical items that occur only once in the corpus.

The text fragments form the fundamental units of translation patterns such as (5) and (6a-b) and are not broken down further. The distinction into text fragments and variables, despite the formation of complement patterns, is a convenient representation for fragment alignment (2.2.3) and template matching (2.3).

(6a) The commission (…) the plan (…) ←→
La commission (…) le plan

(6b) Our government (…) all laws (…) ←→
Notre gouvernement (…) toutes les lois

2.2.3 Alignment of Text Fragments and Variables

Aligning the text fragments and variables of which translation patterns are composed produces translation patterns that are flexible enough for the recombination phase. In so doing, a more refined bilingual knowledge source (bilingual lexicon or "phrasicon") is produced. Aligning text fragments and variables is analogous to aligning words, phrases, terms or even sentences as in conventional bilingual alignment and involves similar problems.

Given that a translation pattern contains a SL and a TL sequence of one or more subsentential text fragments (separated by variables which also represent a separate series of subsentential text fragments), the problem may be viewed as bilingual sequence alignment or computing the optimal or most probable alignment between two sequences of text fragments in two languages. The two sequences of variables are aligned analogously by considering the text fragments that the variables represent, as defined by the corpus, as a separate series of text fragments. The solution to the alignment problem involves a sequence-comparison algorithm and a bilingual similarity (distance) metric.

The bilingual similarity metric is language-neutral in nature and is a combined score based on bilingual lexical distribution of the text fragments (BLD) and the number of cognates the text fragments share (7). The bilingual lexical distribution score (a real value between zero and 1) is computed by Dice's co-efficient (Dice, 1945) and cognates are determined by computing the Levenshtein Distance (Levenshtein, 1966) between SL and TL strings. The Levenshtein Distance is normalised over the maximum distance between the two strings, returning a similarity score or probability that the two strings are cognates. If the similarity score is above an experimentally determined threshold, the two strings are considered to be cognates. Bilingual lexical distribution suffers from well-known problems such as data sparseness, the identification of translingual collocates as opposed to true translations and the varying distributions of morphological variants of words. The inclusion of cognates into the distance metric addresses, to some extent, the problem of low frequency words and provides an alternative score.

$$(7)\qquad \frac{BLD + |Cognates|}{1 + |Cognates|}$$

The alignment of closed class words is particularly problematic. They are of such high frequency that they are unable to be aligned by lexical distribution. They are also not subject to cognate matching. Solutions to this problem include the non-alignment of text fragments composed uniquely of closed class words. An additional method is to 'fill in' the variable positions between text fragments where the variable is uniquely composed of closed class words. For instance *independent states (…) former Soviet Union* becomes *independent states of the former Soviet Union*. However, neither provides a complete solution.

The sequence-comparison algorithm needs to be able to compute alignments denoting translation relations of a more complex variety than simple 1:1 relationships, i.e 2:1, 1:0, etc. It must also compute alignments between adjacent and non-adjacent text fragments (long-distance dependencies) to allow for structural divergences between languages (figure 5). Finally, it needs to execute in a practicable asymptotic running time, therefore excluding an exhaustive search algorithm.

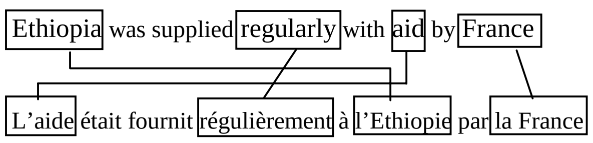


Figure 5. Pattern with Non-Adjacent Alignments[1]

One solution to the problem of sequence alignment is the Dynamic Programming (DP) framework (8). While the problem of sentence alignment is suited to the DP algorithm in that the order of sentences between languages is often similar (Gale & Church, 1993), subsentential fragment alignment must include a search for non-adjacent alignments or long-distance dependencies – a crucial task which DP is inherently unable to perform.

$$(8)\qquad D(i,j) = \min\begin{cases} D(i, j-1) + d(0, y_j) \\ D(i-1, j) + d(x_i, 0) \\ D(i-1, j-1) + d(x_i, y_j) \\ D(i-1, j-2) + d(x_i, \{y_{j-1}, y_j\}) \\ D(i-2, j-1) + d(\{x_{i-1}, x_i\}, y_j) \\ D(i-2, j-2) + d(x_i, y_{j-1}) + d(x_{i-1}, y_j) \\ D(i-1, j-3) + d(x_i, \{y_{j-2}, y_{j-1}, y_j\}) \\ D(i-3, j-1) + d(\{x_{i-2}, x_{i-1}, x_i\}, y_j) \end{cases}$$

An algorithm is proposed that performs the alignment in two passes: the first pass, for which DP is ideally suited, takes into account substitutions (1:1), insertions (0:1), deletions (1:0), compression (2:1, 3:1), expansion (1:2, 1:3) and *swaps* of adjacent fragments (Lowrance & Wagner, 1975), assuming all local alignments are adjacent. The second-pass algorithm computes the (possibly empty) set of non-adjacent alignments of a 1:1 nature. If any are found that improve the initial global alignment computed by DP, they are recorded and removed from the two sequences, while the remaining text fragments are re-fed into the DP algorithm. The final global alignment is then a concatenation of the non-adjacent alignments and the results of the second application of the DP algorithm. The same bilingual similarity metric is used for both passes.

The second-pass algorithm is summarised as follows: given two sequences *x* and *y* of lengths *m* and *n* respectively, each element x_i for $1 \le i \le m$ is compared

[1] The text between the boxed text would appear in the translation patterns as variables. It is included here to make sense of the example.

with each element y_j for $1 \leq j \leq n$. For each potential alignment $<x_i,y_j>$, the similarity score $Score(x_i,y_j)$ is computed. If $Score(x_i,y_j)$ satisfies two conditions, the text fragments x_i and y_j along with $Score(x_i,y_j)$ are recorded as a triple and added to the list of candidate non-adjacent alignments (figure 6). The first condition states that the alignment must be high-scoring (above a threshold) and the second states that the alignment must be non-adjacent i.e. the absolute difference between i and j must be greater than 1.

```
for $i = 1$ to $m$
    for $j = 1$ to $n$
        if $Score(x_i,y_j) > Threshold$ & $abs(i-j) > 1$
            add $\{x_i,y_j, Score(x_i,y_j)\}$ to list of candidate
            non-adjacent alignments
```

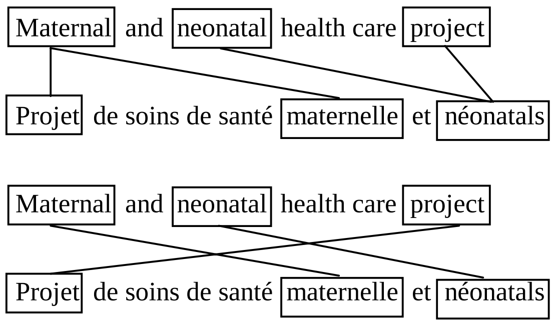
Figure 6. Finding Candidate Non-Adjacent Alignments

Once a list of triples representing candidate non-adjacent alignments has been collected $\{(\alpha,\beta, Score(\alpha,\beta)),...\}$, for each triple, three conditions are applied before the alignment is declared a valid non-adjacent alignment. First, if the SL (TL) text fragment α (β) of the proposed alignment $<\alpha,\beta>$ has been used elsewhere in a *non*-adjacent alignment, it cannot be used to form a subsequent one. Second, if the alignment $<\alpha,\beta>$ is a subset of an alignment $<A,B>$ that has previously been computed by DP (where A and B represent a series of SL and TL text fragments respectively and where $\alpha \in A$ and $\beta \in B$), they cannot be considered further. The reason for this is simply that the DP algorithm has found that an alignment other than a 1:1 type is more probable, thus the proposed alignment is not likely to improve the likelihood of the final global alignment.

$$(9)\quad \begin{array}{l} Score(\alpha,\beta) > Score(\langle A,B\rangle_p) \,\& \\ Score(\alpha,\beta) > Score(\langle A,B\rangle_q), \\ \alpha \in A_p,\ \beta \in B_q, 1 \leq p \leq k, 1 \leq q \leq k,\ p \neq q \end{array}$$

Finally, the score of the proposed non-adjacent alignment $<\alpha,\beta>$ is tested to see whether it is greater than both the two individual scores of the two alignments computed by DP of which α and β are a member. In more detail, each adjacent alignment computed by DP is represented as a pair $<A,B>$, where A and B represent a series of SL and TL fragments according to the alignment patterns stipulated by the DP equation. The initial global alignment computed solely by DP is a set containing one or more of these alignments, $\{<A,B>_1,\ldots,<A,B>_k\}$. The proposed non-adjacent alignment can also be represented by a pair $<\alpha,\beta>$ where α and β represent one SL and TL text fragment respectively. This condition is summarised in (9)[2], which stipulates that the score of the non-adjacent alignment must be better than both of the two adjacent alignments which subsume it, which means that the addition of the proposed non-adjacent alignment would *improve* the overall global alignment.

To illustrate the 2nd pass algorithm, the translation pattern in figure 7 is first aligned using the DP algorithm. The alignment of the text fragments (boxed text) is incorrect. After the application of the 2nd pass algorithm, a non-adjacent alignment (*project-projet*) is computed. Subsequent application of the DP algorithm produces the correspondences *maternal-maternelle*, and *neonatal-néonatals*.



Maternal and neonatal health care project
Projet de soins de santé maternelle et néonatals
Maternal and neonatal health care project
Projet de soins de santé maternelle et néonatals

Figure 7: Before and After 2nd Pass Algorithm

Only non-adjacent alignments of a 1:1 nature are considered to ensure that the alignment process executes in a practical asymptotic running time. The second-pass algorithm therefore runs with a worst-case complexity of $\mathcal{O}(mn)$, as does DP.

2.3 Recombination

In the case of EBMT systems where examples are stored as tree structures with the correspondences between the fragments explicitly labelled, the recombination problem is a matter of tree traversal (Sato, 1995; Watanabe, 1995). In this approach, the input to the recombination phase is an SL input sentence and a set of translation patterns with their constituent text fragments and variables aligned. The output is one or more TL translations which can be ranked best first according to confidence.

Given an SL input sentence, translation patterns that share lexical items with the SL input and partially (or fully) cover it are retrieved in a pattern-matching process. From these, the patterns whose SL side cover the SL input to the greatest extent (longest cover) are selected. They are termed *base patterns*, as they provide sentential context in the translation process. It is intuitive that the greater the extent of the cover of the base pattern, the more context and hence the less ambiguity and complexity in the translation process. If the SL side of the base pattern does not fully cover the SL input, any unmatched parts of the SL input are bound to the variables on the SL side of the base pattern. Translation patterns from the remaining set, containing SL text fragments that match the string value of the instantiated variables on the SL side of the base pattern, are retrieved. Since the text fragments and variables in translation patterns are aligned, their TL equivalents are easily retrieved and bound to the relevant TL variables in the TL side of the base pattern.

The following is a simple example: given the SL input in (10), suppose the longest covering base pattern is (11). To complete the match between (10) and the SL side of (11), a translation pattern containing the text fragment *Ethiopia* is required (12). The TL translation (13) is generated by recombining the text fragments: *Ethiopia* and *Etiopía* are aligned in (12) as are the variables in the base pattern (11). Since *Ethiopia* and *Etiopía* are aligned on a 1:1 basis and so are the variables in the base pattern (11), the TL text fragment *Etiopía* is bound to the variable on the TL side of (11) to produce (13). It is clearer to think of the algorithm proceeding in terms of matching

[2] The notation A_p is shorthand for "the A in $<A,B>_p$".

successive alignments of the variables in the base pattern, rather than successive variables on the SL side of the base pattern. This is because it is not only text fragments that are matched but also the alignment type. As pointed out in 2.2.3, alignments may be non-bijective.

(10) AIDS control programme for Ethiopia
(11) AIDS control programme for (…) ←→ programa contra el SIDA para (…)
(12) (…) Ethiopia ←→ (…) Etiopía
(13) programa contra el SIDA para Etiopía

2.3.1 Translation Confidence

On account of translation ambiguity, there may be more than one translation per SL input. The list of competing translations is ranked best-first according to a score of translation confidence (14). The score is based on three components. The first is a score based on a bigram model of the TL formed from the TL side of the corpus (*BG*). The model is used as a test of the "grammaticality" of the TL sentence. The second is a score that checks the extent of the overlap of the inserted TL text fragments (*OV*) in an attempt to reduce boundary friction. The third is a score reflecting the bilingual distance between translated text fragments (*BD*). Finally a penalty *p* is applied if any TL variables in the base pattern remain unfilled.

(14) $$(\omega_1(BG)+\omega_2(OV)+\omega_3(BD))(1-p)$$

The TL bigram model is computed from the TL side of the corpus. It is easily improved by adding more TL text. The strength of association between the words in the bigrams is computed by Dice's co-efficient. For each TL sentence produced of length *n*, *BG* in (14) is computed by taking the average score of each bigram within it, or each word pair $<w_i, w_{i+1}>$. This is summarised in (15): For each bigram in the TL sentence, its frequency in the TL corpus is divided by the sum of the frequencies of either word in the corpus and multiplied by 2. This score is added to a running total. The total is then divided by the number of bigrams in the sentence or *n-1*.

(15) $$\frac{\sum_{i=1}^{n-1} 2\left(\frac{|w_i, w_{i+1}|}{|w_i|+|w_{i+1}|}\right)}{n-1}$$

The overlap term (*OV*) in (14) makes use of the fact that the text fragments have been extracted from real texts and so there is information about the contexts in which the fragments are known to have occurred. The idea is borrowed from Somers et al. (1994) who use the image of hooks being attached before and after the text fragment, indicating the words (and POS tags) that can occur before and after it. A window of five words before and after the text fragment bound to each TL variable of the base pattern is considered. If the same sequence of 5 words or less preceding or subsequent to the text fragment is found in the TL side of the corpus, a score reflecting the extent of the overlap between the two sequences is returned. Where multiple match sequences are found, the maximum overlap value is returned. As an example, when the TL text fragment *Etiopía* is inserted into the TL base variable in (11), the word sequence [*programa, contra, el, SIDA, para*] is looked for in the corpus preceding *Etiopía*. Starting from *Etiopía*, the extent of the overlap is determined. If only *para* appears before *Etiopía* in the corpus, the overlap score returned would be 1/5 = 0.2. If there were a match sequence subsequent to *Etiopía*, the average of the previous and the subsequent overlap would be returned. The formula is given in (16): For each base pattern with *n (n>0)* TL variable positions, the overlap score for each variable V_i, is the average of the sum of the previous and subsequent overlaps. The final overlap score is the average overlap score for each V_i. If a V_i occurs at a sentence boundary, only the previous or subsequent overlap score is considered.

(16) $$\frac{\sum_{i=1}^{n}\left(\frac{PreviousOverlap(V_i)}{5}+\frac{SubsequentOverlap(V_i)}{5}\right)\Big/2}{n}$$

For each variable alignment, the bilingual distance score is calculated between the SL text fragments bound to the SL variables of the base pattern and their TL equivalents using equation (7) in the alignment process (section 2.2.3). The average score for each alignment is returned to become *BD* in equation (14).

Finally a penalty *p* is applied if there are any variables on the TL side of the base pattern that have not been instantiated on account of *partial* matches between the SL input and the SL side of the base pattern. Partial matches occur when not all variables on the SL side of the base pattern can be matched by SL text fragments in the remaining set of translation patterns. For example, if there were no translation pattern containing the text fragment *Ethiopia*, the TL variable would not be instantiated, resulting in the partial translation: *programa contra el SIDA para (…)*. The penalty *p* is simply the number of unfilled variables on the TL side of the base pattern over the total number of variables on the TL side of the base pattern. In the case where *p = 1* i.e. when all TL variables remain unfilled, *p* is scaled by 0.9 to avoid *(1-p)* becoming zero.

The three terms *BG*, *OV* and *BD* return a real value between 0 and 1.0. To ensure that the sum of the three terms remains between 0 and 1.0, $\omega_1 + \omega_2 + \omega_3 = 1.0$. In the experiments undertaken, $\omega_1 = \omega_2 = \omega_2 = 1/3$, thus providing equal weighting to all terms. The term containing the penalty *(1-p)*, returns a value between 0.1 and 1.0, ensuring that the entire score remains a real value no greater than 1.0.

3 Adding Linguistic Knowledge

An approach based on the distributions of surface forms is error prone and liable to the production of translation patterns that are false translations. Linguistic knowledge, in the form of morphological analysis and POS tagging, is added in an attempt to increase the accuracy of the translation patterns and coverage. This results in three variants of this approach to EBMT: The first variant, outlined in section 2, is based on surface forms and requires no significant external linguistic knowledge, apart from the sentence-aligned parallel corpus. However, in order to remove obvious errors a very small amount of (optional) minor linguistic information is added in the form of stop lists of closed-class words and information for tokenisation, which compromises, to a certain extent, language-neutrality and portability. We believe this information to be obtainable in a few person-hours. The second variant is augmented with morphological information in the form of a set of two-level morphological rules (Koskenniemi, 1983), a list of lemmas and a finite state transducer (PC-Kimmo). The corpus is lemmatised so that translation patterns composed of lemmas and translation patterns on the morphological level are extracted. Furthermore, clitics such as the French *des*, *du* etc are expanded in an effort to increase coverage and reduce boundary friction. TL surface forms are generated by reversing the direction of the transducer, combing TL lemmas with TL tags. Since no POS information nor word grammar is used, the surface forms are validated by means of a list of TL surface forms (a spelling dictionary in this case). The third variant is extended even further to include POS annotation, using TreeTagger (Schmid, 1994).

As more linguistic knowledge sources are added to a system, portability generally decreases. In this case, the linguistic knowledge added is deliberately kept to a minimum and to the sort of knowledge sources that are reasonably widely available, at least for European languages. Even if they are not, it is not too difficult nor time-consuming to produce a list of stems and a set of two-level spelling rules. POS taggers such as TreeTagger and Brill's tagger (Brill, 1992) are also easily trained for new languages if they do not already exist.

3.1 Morphological Analysis

The principle source of errors is the phenomenon where SL words that are orthographically identical (homographs, polysemes, orthographically identical variants of the same lemma) have different TL equivalents. This can be corrected, to a certain extent, by the addition of morphological analysis and POS tagging to distinguish them. Consider the example corpus in (17) where the morphological variants of *give* are orthographically identical, but their TL equivalents are not. This results in a translation pattern with an empty TL side and is consequently a false translation (18).

(17) He **gave the** plans **up** ←→
Il abandonna les projets
They **gave the** suggestion **up** ←→
Ils abandonnèrent la suggestion

(18) (…) gave the (…) up ←→

Lemmatising the corpus (19) produces a more accurate, if more general, translation pattern (20) composed of lemmas. Furthermore, abstracting all morphological variants to their root form increases the chance of string co-occurrence and collocation formation, resulting in a greater number of translation patterns with the potential to increase coverage.

(19) he **give the** plan **up** ←→
il **abandonner le** projet
they **give the** suggestion **up** ←→
ils **abandonner le** suggestion

(20) (…) give the (…) up ←→
(…) abandonner le (…)

3.2 Part-of-Speech Tagging

A further case is SL homographs that are of different parts-of-speech and are consequently different 'words' relating to different concepts. It is very likely that in such cases each individual homograph will have a different TL translation. Consider *wave* as a noun and a verb in the corpus (21) and the resulting erroneous translation pattern (22).

(21) She **waves** goodbye ←→
Elle agite la main et dit au revoir
The **waves** are enormous ←→
Les vagues sont énormes

(22) (…) waves (…) ←→

Lemmatisation is not a solution since both forms of *wave* abstract to the (orthographically) same root. The solution lies in POS tagging the corpus and modifying the string co-occurrence algorithm for extracting patterns to include the POS tag as a feature. In such a case, the two forms of *wave* would not 'co-occur' to form a collocation, since their respective POS tags would distinguish one form of *wave* from the other and consider them as different words.

3.2 Semantic Tagging

The solution to the false translation problem in this approach appears to lie in being able to distinguish different 'words' from each other, on criteria than just their orthography. Distinguishing different 'words' i.e. strings of characters that relate to different concepts is not entirely possible by morphological analysis and POS tagging alone. Consider the corpus in (23) where the two senses of the polysemous verb *mark* occur with identical orthographies and POS tag, but different translations, to produce (24). While the two meanings of *mark* may or not have been historically related, they are clearly separate and consequently have different translations. The solution appears to lie in effective semantic tagging, if it were feasible, and including that tag as part of a feature structure in the co-occurrence algorithm. Again this would enable the different semantic forms of a verb such as *mark* to be distinguished and prevented from forming collocations.

(23) She **marked the** papers ←→
Elle corrigea les examens
He **marked the** table ←→
Il tâcha la table

(24) (…) marked the (…) ←→

3.3 Other Sources of Error

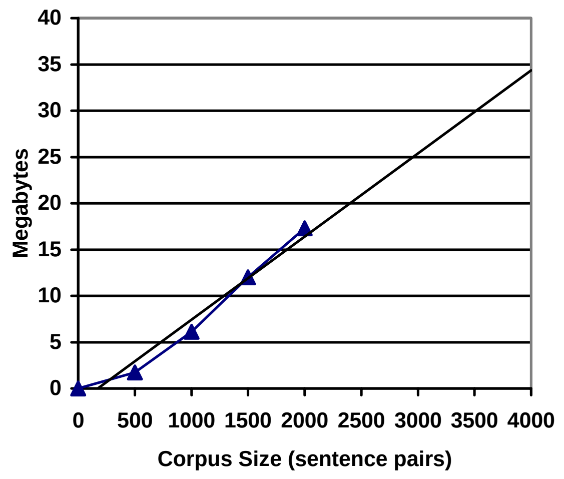
The fact that translation patterns are formed using an algorithm based on the distribution of strings in a corpus means that there are fewer constraints than in a traditional rule-based system or system that generalizes translation examples by morphological, syntactic or even semantic means. The patterns do not contain any restrictions on the morphology, number, gender or syntactic category of the text fragment that binds to the variables in the base pattern. Furthermore, adding external knowledge sources is no guarantee of increasing the accuracy (or coverage) of a system. Taggers, parsers and other analysers are prone to a certain degree of error themselves.

4 Complexity Issues

4.1 Translation Pattern Extraction

The translation pattern extraction algorithm, in particular the construction of the collocation trees (section 2.2.1) is a notable source of complexity. As each lexical item occurring in two or more sentences is added, the tree increases in size. Since each lexical item may be added to the daughter(s) of the root node and any subsequent node, the trees grow to become very much larger than the corpus itself. On account of their high frequency, closed class words are particularly problematic in that they dramatically increase the size of the trees.

In this instance, space becomes an issue. As the size of the corpus increases, the collocation trees become very large, in effect, placing a restriction on extensibility. Graph 1 shows how the size of the collocation trees for a corpus of French (measured in megabytes) increases in relation to the size of the corpus (measured in sentence pairs). The dark line indicates the extrapolation of the data. Although the relationship appears to be linear, the trees take a substantial portion of memory and the limit on corpus size is dictated by the memory of a particular machine (unless hard-drive space is efficiently utilised). The size of the collocation trees is the greatest obstacle to scaling up this approach. The solution lies in reducing the space required by the trees, in particular, reducing the number of levels. This may be achieved by adding the longest substrings that appear in 2 or more sentence pairs to the trees instead of individual word tokens.



Graph 1: Corpus Size vs. Size of Collocation Trees

The polynomial two-pass alignment algorithm outlined in 2.2.3 is also a notable source of complexity. In fact it is the part of the extraction algorithm that incurs the largest time processing overhead The well-known DP algorithm consists of two embedded loops and is clearly of a time and space complexity of $\mathcal{O}(mn)$, where *m* and *n* are the lengths of the two sequences to be aligned. DP is practical in terms of time and space, but not fast. In this application, the values of *m* and *n* are rarely large.

The grain-size of the local alignment types is also pertinent. Further text fragment alignment patterns such as 2:3, 3:2 and even 2:4 could easily be included as terms in the DP equation. More terms entails a larger processing overhead, but as grain-size increases, the problem of long distance dependencies becomes less of an issue as their likelihood diminishes. Moreover, the granularity of the alignment types involves the classic trade-off between accuracy and flexibility. While matches with longer fragments reduce ambiguity, flexibility suffers since there is a lower probability of a match. Matches with shorter segments involve a greater amount of translation ambiguity, passage and boundary friction (Nirenburg et al. 1993: 48). Fine as opposed to coarse-grained local alignment types have been favoured in this approach in order to maintain flexibility and recall.

4.2 Recombination

The core recombination algorithm is an area where the amount of entropy incurred by translation ambiguity and the lack of constraints in the translation patterns produce the most notable source of computational complexity. Since any one SL fragment can have more than one TL equivalent and there is no information present to favour one fragment over another, all possible TL fragments are considered. This results in (possibly) numerous translations per base pattern which are ranked by the translation confidence score (14).

This complexity is increased if recursive matches are incorporated. In the examples (10)-(13), the string values bound to the variables on the SL side of the base pattern are matched against *single* SL text fragments in the remaining translation patterns (direct matches). If no direct matches are found, a recursive matching algorithm is invoked. The algorithm matches a text fragment by attempting to recursively match successively shorter portions of it against the SL text fragments in the translation patterns. The TL equivalents are concatenated naively according to the order of the matches with the SL fragments. Increased complexity comes about since each SL fragment that forms a subpart of the entire match can have more than one TL equivalent, each of which has to be taken into account.

The addition of knowledge sources has consequence for the complexity of the recombination phase. The second variant of this approach, which is augmented to included morphological analysis includes the extraction of not only translation patterns composed of lemmas, but translation patterns composed of morphological tags or *morphological patterns*. They are analogous to regular translation patterns (see 2.1) except that they are composed of morphological tags instead of lexical items.

Morphological patterns are formed trivially by the replacement of lemmas in regular translation patterns extracted with the corresponding morphological tags computed during lemmatisation of the corpus. The tags represent (inflectional) morphological change from the root lexical form. The alignment between the sequences of fragments (composed of tags) and variables in a morphological pattern is carried out trivially by transposing the alignments from the regular translation pattern from which it was formed.

(25) The telephones worked ←→
Les téléphones fonctionnaient

The telephone failed ←→
Le téléphone échouait

(26) **The telephone**+*s* work+*ed* ←→
Le+*s* **téléphone**+*s* fonctionner+*aient*

The telephone fail+*ed* ←→
Le téléphone échouer+*ait*

The corpus (25) is lemmatised to form (26), from which translation pattern (27a) is extracted (among others). By replacing the lemmas with the corresponding morphological tags from (26), the morphological patterns (27b) and (27c) are formed. The empty parentheses denote cases of no morphological change from the root lexical form. The fragments and variables in (27b) and (27c) are aligned according to alignment of the fragments in (27a).

(27a) The telephone (…) ←→ Le téléphone (…)
(27b) [] +s (…) ←→ +s +s (…)
(27c) [] [] (…) ←→ [] [] (…)

The recombination algorithm is updated to operate with translation patterns composed of lemmas and morphological patterns. First, the SL input is lemmatised and recombination proceeds in exactly the same manner as described in 2.3, except that matching takes place on the lexical level and the result is a set of one or more sequences of TL lemmas. Morphological patterns are retrieved and recombined analogously to cover the sequence of SL morphological tags computed when the SL input is lemmatised. This results in a set of one or more sequences of TL morphological tags. TL sentences are produced by laying the sequences of TL lemmas and tags, generating sequences of surface forms. As an example, the sequence of lemmas (28a) is layered with the sequence of tags (28b) to produce (28c). It is clear that recombination that takes place on more than one level (here the lexical and morphological) increases the processing overhead.

(28a) <Programme, contre, le, maladie, diarrhéique>
(28b) <[], [], +s, +s, +s>
(28c) Programme contre les maladies diarrhéiques

One further complexity issue is ambiguity of analysis. When a surface form is lemmatised using just a list of lemmas and a set of two-level spelling rules, there may be ambiguity about the lemma to which it belongs and consequently ambiguity of the tag. The layering algorithm takes into account all alternatives thus adding a further parameter, increasing the complexity of the algorithm. The third variant of this approach takes this factor into account. The TreeTagger, which incorporates a lemmatiser, is used to analyse the corpus and resolve ambiguous analyses. This has the effect of removing this extra parameter and reducing the complexity of the layering algorithm.

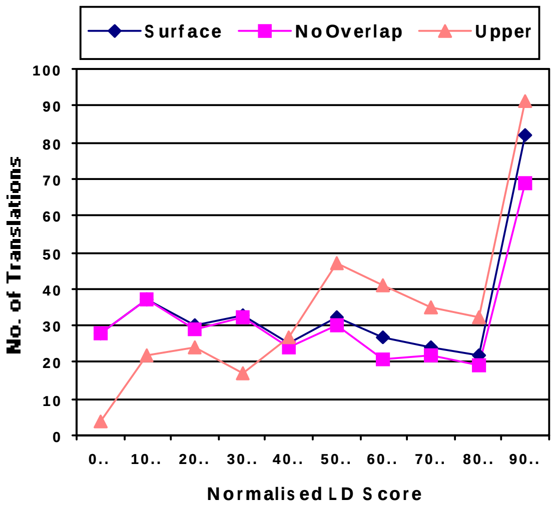
5 Experiments

Using the first of the three variants - the one based on surface forms - translation patterns were extracted from a 3,000-sentence-pair sample of the World Health Organisation AFI corpus of 4,858 French and English titles.[3] From the remaining set of 1,858 sentence pairs, 1,000 were randomly selected and used as a test set of unseen SL input. 36 SL sentences in the test set occurred in the training set. Furthermore, 728 SL sentences in the test set were entirely composed of lexical items found in the training set, leaving 272 that contained one or more unseen words. 10,767 patterns were extracted and the most frequent values of p and q (figure 3) were 2 or 3.

$$(29)\quad 1-\frac{LD(TR,RT)}{Longest(TR,RT)}$$

For each SL input, one ore more translation solutions were produced. The best translation among this set (*TR*) was selected according to the translation confidence score (14). The correctness of the translation was determined automatically by comparing it with the reference translation given in the corpus (*RT*). The Levenshtein Distance (*LD*) between the translation (*TR*) and the reference translation (*RT*) was normalised to account for the maximum distance between the two strings (maximum string length), returning a "percentage of similarity" score (29).

Graph 2 shows the distributions of the results. Only full translations were considered. The line depicted by 'Surface' indicates the results for the test set of 1,000 SL sentences. 340 SL sentences were successfully translated. Recall - the number of SL sentences translated divided by the number of sentences in the test set - stands at 340/1000 = 34%. Maximal recall is obtained by dividing 340 by 728 (47%) i.e. the number of SL sentences in the test set uniquely composed of lexical items found in the training set. 82 of the 340 translations (24%) were 90-100% accurate. Moreover, 73 translations (21.5%) were 100% correct. Accurate translations were largely produced by base patterns whose SL side substantially covered the SL input. Moreover, such patterns, of which (11) is typical, represent extremely frequent phenomena in the corpus.

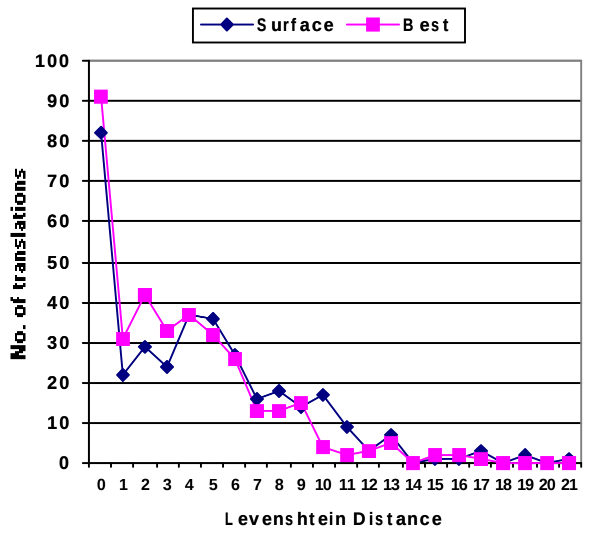


Graph 2: Distribution of Results

The line depicted by 'NoOverlap' shows the results for the test set with the 36 sentences which occur in the training set removed. Although no direct matches between the SL inputs and the SL sentences in the training set are sought (only the EBMT mechanism is evaluated), 10 of the 100% accurate translations are no longer present and there are less high-scoring translations generally. The line depicted by 'Upper' represents the theoretical upper bound of the system. For each SL input, the best possible translation solution – the one which returns the highest percentage of similarity score (29) among the set of all translation solutions – is selected, irrespective of its

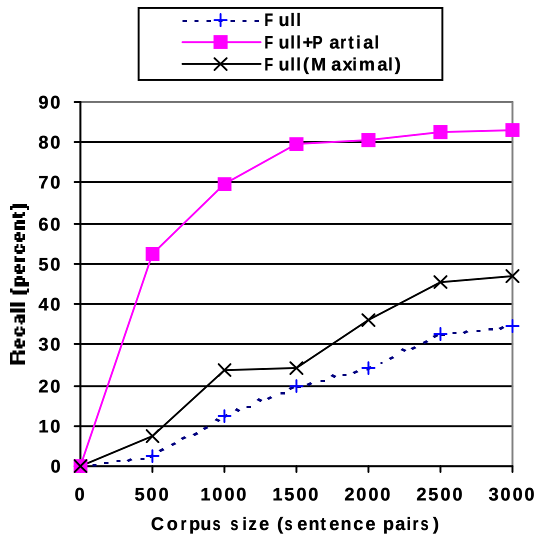
[3] http://www.who.int/pll/cat/cat_resources.html

translation confidence score (14). This line shows that less low-scoring translations and more high-scoring translations are produced. The upper bound is not reached due to the inefficacy of the translation confidence score i.e. it does not always assign the best translation with the highest score.



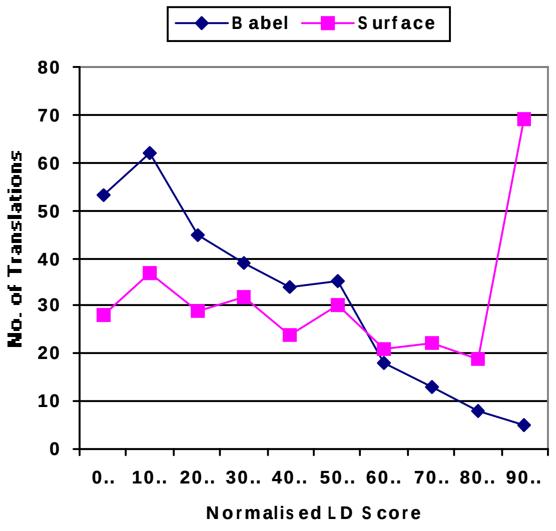
Graph 3: Word Error Rate

Graph 3 shows, for the same test set of 1,000 sentences, the distribution of word errors. For each translation solution, the error rate is calculated as the plain Levenshtein distance, as opposed to normalised Levenshtein Distance (29). In this case, the Levenshtein Distance represents the number of insertions, deletions or substitutions required to transform the translation string into the reference translation given in the corpus. The line 'Surface' represents the errors of the translation solutions selected according to (14) and the line 'Upper' represents the errors of the upper bound translations. In each case, the number of translations with zero errors is equal to the number of translations that were 100% accurate .



Graph 4: Recall vs. Corpus Size

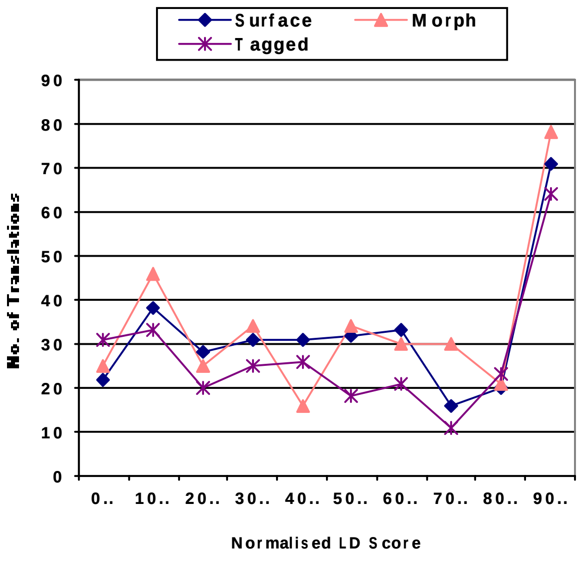
Graph 4 shows increasing rates of recall as corpus size, measured in sentence pairs, increases. A test set of 500 unseen SL input sentences was used in each case. The line 'Full' represents recall calculated by dividing the number of SL sentences fully translated by the number of SL sentences in the test set. The line 'Full+Partial' represents not only SL sentences that were fully translated, but includes those that were only partially matched against the translation patterns. The line 'Full(Maximal)' represents recall calculated by dividing the number of SL sentences fully translated by the number of SL sentences in the test set that are uniquely composed of lexical items found in the training set. The rates of recall rise fairly slowly if only full matches are considered, indicating the need for larger training sets. However, high rates of recall are rapidly achieved if incomplete or partial translations are considered. Including partial translations results in many more low-scoring than high-scoring translations.



Graph 5: Comparison with Babel Fish

In order to compare this approach with a commercial MT system, the same test set of 1,000 SL input sentences was translated using the *BabelFish/Systran* translation facility on `http://www.alta-vista.com`. *BabelFish* is robust enough to achieve 100% recall for the test set of SL input sentences, far outperforming this approach. However, in order to compare the rates of *precision*, graph 5 only depicts the distributions of results for the sentences that this approach was able to translate fully. To make the test fairer, the 36 SL sentences in the test set that occur in the training set were removed. The graph clearly shows that for that test set, this approach outperformed *BabelFish*. This is to be expected since this approach is tuned to the text type.

In order to test the hypothesis that adding linguistic knowledge improves accuracy and coverage (section 3), a comparison of the distributions of results of each of the three variants of this approach is shown in graph 6. For each variant, translation patterns were extracted from the same 2,500-sentence-pair sample in order to make a fair test. Complexity issues (as outlined in section 4) prevented the use of a larger training set. The variant based on surface forms extracted 9,327 patterns, the variant including morphological information extracted 9,610 while the variant augmented further with POS information extracted only 7,237 patterns. The same test set of 1,000 unseen input sentences from the previous experiments was used. Only full translations were considered. The results are summarised in table 1.



Graph 6: Comparison of Distributions of each Variant

The distributions of the three variants are largely similar. The salient differences between them are established in terms of recall and the number of high-scoring translations. The morphological variant translated more SL sentences than the variant based on surface forms. Furthermore, it produced a greater number of accurate (defined as 90-100% correct) or 100% correct translations. The variant including POS information performed less well than the variant based on surface forms. Fewer translation patterns were produced, resulting in a lower number of SL sentences translated and a lower number of high-scoring translations. The reasons for the poor performance of this variant include tagger errors and the fact that more constraints operate in the translation pattern extraction phase (section 3.2), resulting in fewer translation patterns.

	Surface	Morph	Tagged
Patterns	9,327	9,610	7,237
Translations	322	339	272
Recall	32.2%	33.9%	27.2%
Max recall	44.2%	46.5%	37.3%
Accurate	71	78	64
100% correct	61	68	53

Table 1: Comparison of the Three Variants

Since the evaluation method utilised in these experiments is automatic, it is possible that some of the lower scoring translations may be “correct” or useful by some other criteria i.e. acceptability to a post-editor as raw MT output. Its usefulness could be evaluated by the amount of time or effort required to revise it to the required standard, as in (Minnis, 1994; Whyman & Somers, 1999). Alternatively, one could use any of the various methodologies in the MT evaluation literature based on human observations and scales of correctness.

Discussion

The addition of morphological information to the approach based on surface forms increased recall, if only slightly, and returned a slightly higher number of translations that are 100% accurate or very nearly so. There are numerous possibilities as to why there is only a slight improvement: first, only a relatively small amount of linguistic information is added. Second, the nature of the corpus (titles) is such that there are few morphological variants per root form. There tends to be a higher density of root forms in titles than in ‘ordinary’ texts. Third, the recursive matching algorithm in recombination (section 4.2) adds an amount of noise to an already noisy channel. This has the effect of blurring the improvements made by adding morphological information. Finally, and more importantly, more training data is required.

The results of the variant where POS information is included are disappointing and do not improve upon the variant based on surface forms. Given the results as they stand, the variant augmented uniquely with morphological information performs only slightly better than the variant based on surface forms. Therefore, the effort or cost (financial and computational) of adding linguistic knowledge sources may not be justified, since they affect portability to new language pairs and domains.

The rates of recall are currently low. This can only be attributed to the fact that there is simply not enough data in the corpus of the size reported. The approach must be scaled up to include corpora of millions of words. However, this requirement is in contrast to the observations in graph 1 where the size of the collocation trees severely limits corpus size (unless hard-drive space is efficiently utilised). Therefore, scalability is problematic. On the other hand, if one considers the satisfactory rates of recall when partial matches are included (graph 4), one has to consider whether this methodology is of more use as a flexible translation memory system (Langé et al. 1997; Planas & Furuse, 1999; McTait et al. 1999) where information from more than one translation example can be used to create new TL suggestions.

References

Brill, E. (1992) A Simple Rule-based Part-of-Speech Tagger. *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, 152-155.

Brown, P.F, S.A. Della Pietra, V.J. Della Pietra & R.L. Mercer (1993) The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics* **19**: 263–311.

Brown, R.D. (1999) Adding Linguistic Knowledge to a Lexical Example-based Translation System. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, Chester, England, 22–32.

Carl, M. (1999) Inducing Translation Templates for Example-Based Machine Translation. *Machine Translation Summit VII*, Singapore, 617–624.

Dagan, I., K.W. Church & W.A. Gale (1993) Robust Bilingual Word Alignment for Machine Aided Translation. *Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives*, Columbus, Ohio, 1–8.

Dice, L.R. (1945) Measures of the Amount of Ecological Association Between Species. *Geology* **26**: 297-302.

Echizen-ya, H., K. Araki, Y. Momouchi & K. Tochinai (2000) Effectiveness of Layering Translation Rules based on Transition Networks in Machine Translation using Inductive Learning with Genetic Algorithms. *MT 2000, Machine Translation and Multilingual Applications in the New Millennium*, Exeter, England, 5-1-8.

Fung, P. & K. McKeown (1997) A Technical Word- and Term-Translation Aid Using Noisy Parallel Corpora Across Language Groups. *Machine Translation* **12**: 53–87.

Furuse, O. & H. Iida (1992) An Example-Based Method for Transfer-Driven Machine Translation. *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation. Empiricist v. Rationalist Methods in MT. TMI-92*, Montréal, Québec, 139-150.

Gale, W.A. & K.W. Church (1993) A Program for Aligning Sentences in Bilingual Corpora. *Computational Linguistics* **19**: 75-102.

Güvenir, H.A. & I. Cicekli (1998) Learning Translation Templates from Examples. *Information Systems* **23**: 353–363.

Kaji, H., Y. Kida & Y. Morimoto (1992) Learning Translation Templates from Bilingual Text. *Proceedings of the fifteenth* [sic] *International Conference on Computational Linguistics: COLING-92*, Nantes, France, 672–678.

Koskenniemi, K. (1983) Two-Level Model for Morphological Analysis. *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, 683-685.

Langé, J-M, E. Gaussier & B. Daille (1997) Bricks and Skeletons: Some Ideas for the Near Future of MAHT. *Machine Translation* **12**: 75-102.

Levenshtein, V.I. (1966) Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Cybernetics and Control Theory* **10**: 707-710.

Lowrance, R. & R.A. Wagner (1975) An Extension of the String-to-String Correction Problem. *Journal of the Association for Computing Machinery* **22**: 177-183.

McTait, K. & A. Trujillo (1999) A Language-Neutral Sparse-Data Algorithm for Extracting Translation Patterns. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-99)*, Chester, England, 98–108.

McTait, K, M. Olohan & A. Trujillo (1999) A Building Blocks Approach to Translation Memory. *Proceedings of the 21st Aslib International Conference on Translating and the Computer*, London, England.

Malavazos, C. & S. Piperidis (2000) Application of Analogical Modelling to Example Based Machine Translation. *The 18th International Conference on Computational Linguistics: COLING 2000 in Europe*, Saarbrücken, Germany, 516–522.

Maruyama, H. & H. Watanabe (1992) Tree Cover Search Algorithm for Example-Based Translation. *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-92)*, Montréal, Québec, 173–184.

Matsumoto, Y. & M. Kitamura (1995) Acquisition of Translation Rules from Parallel Corpora, in Mitkov & Nikolov (1997), pp. 405-416.

Minnis, S. (1994) A Simple and Practical Method for Evaluating Machine Translation Quality. *Machine Translation* **9**: 133-149.

Mitkov, R. & N. Nicolov (eds) (1997) *Recent Advances in Natural Language Processing: Selected Papers from RANLP '95*, Amsterdam: John Benjamins.

Planas, E. & O. Furuse (1999) Formalizing Translation Memories. *Machine Translation Summit VII*, Singapore, 331-339.

Nirenburg, S., C. Domashnev & D.J. Grannes (1993) Two Approaches to Matching in Example-Based Machine Translation. *Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium, 47–57.

Sato, S. (1995) MBT2: A Method for Combining Fragments of Examples in Example Based Machine Translation. *Artificial Intelligence* **75**: 31–49.

Schmid, H. (1994) Probabilistic Part-of-Speech Tagging Using Decision Trees. *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK, 44-49.

Somers, H.L., I. McLean & D. Jones (1994) Experiments in Multilingual Example-Based Generation. *CSNLP 1994: Third Conference on the Cognitive Science of Natural Language Processing*, Dublin, Ireland.

Somers, H. L. (1998) Further Experiments in Bilingual Text Alignment. *International Journal of Corpus Linguistics* **3**: 115-150.

Takeda, K. (1996) Pattern-Based Context-Free Grammars for Machine Translation. *Proceedings of the 34th Meeting of the Association for Computational Linguistics*, Santa Cruz, CA, 144-151.

Wang, Y-Y. & A. Waibel (1998) Modeling with Structures in Statistical Machine Translation. *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Quebec, 1357–1363.

Watanabe, H. (1992) A Similarity-Driven Transfer System. *Proceedings of the fifteenth* [sic] *International Conference on Computational Linguistics: COLING-92*, Nantes, France, 770-776.

Watanabe, H. (1993) A Method For Extracting Translation Patterns from Translation Examples. *Proceedings of the 5th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-93), MT in the Next Generation*, Kyoto, Japan, 292-301.

Watanabe, H. (1995) A Model of a Bi-Directional Transfer Mechanism Using Rule Combinations. *Machine Translation* **10**: 269–291.

Whyman, E.K. & H.L. Somers (1999) Evaluation Metrics for a Translation Memory System. *Software-Practice and Experience* **29**: 1265-1284.

A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora

Arul Menezes and Stephen D. Richardson

Microsoft Research
One Microsoft Way
Redmond, WA 98008, USA
arulm@microsoft.com
steveri@microsoft.com

Abstract

Translation systems that automatically extract transfer mappings (rules or examples) from bilingual corpora have been hampered by the difficulty of achieving accurate alignment and acquiring high quality mappings. We describe an algorithm that uses a best-first strategy and a small alignment grammar to significantly improve the quality of the mappings extracted. For each mapping, frequencies are computed and sufficient context is retained to distinguish competing mappings during translation. Variants of the algorithm are run against a corpus containing 200K sentence pairs and evaluated based on the quality of resulting translations.

1 Introduction

A machine translation system requires a substantial amount of translation knowledge typically embodied in bilingual dictionaries, transfer rules, example bases, or a statistical model. Over the last decade, research has focused on the automatic acquisition of this knowledge from bilingual corpora. Statistical systems build translation models from this data without linguistic analysis (Brown, 1993). Another class of systems, including our own, parses sentences in parallel sentence-aligned corpora to extract transfer rules or examples (Kaji, 1992; Meyers, 2000; Watanabe, 2000). These systems typically obtain a predicate-argument or dependency structure for source and target sentences, which are then aligned, and from the resulting alignment, lexical and structural translation correspondences are extracted, which are then represented as a set of rules or an example-base for translation.
However, before this method of knowledge acquisition can be fully automated, a number of issues remain to be addressed. The alignment and transfer-mapping acquisition procedure must acquire rules with very high precision and be robust against errors in parsing, sentence-level alignment and in the alignment procedure itself. The procedure must also produce transfer mappings that provide sufficient context to enable the translation system utilizing these mappings to choose the appropriate translation for a given context.

In this paper, we describe the alignment and transfer-acquisition algorithm used in the WindowsMT system, which attempts to address the issues raised above. This system acquires transfer mappings by aligning pairs of *logical forms (LFs)*, which are dependency structures similar to those described by Jensen (1993). These are obtained by parsing a sentence-aligned bilingual corpus. (The problem of aligning parallel corpora at the sentence level has been addressed by Meyers (1998b), Chen (1993) and others, and is beyond the scope of this paper).

2 Logical Form

A Logical Form (LF) is an unordered graph representing the relations among the most meaningful elements of a sentence. Nodes are identified by the lemma of a content word and directed, labeled arcs indicate the underlying semantic relations. It is intended to be as independent as possible of specific languages and their grammars. In particular, LFs from different languages use the same relation types and provide similar analyses for similar constructions. The logical form abstracts away from such language-particular aspects of a sentence as constituent order, inflectional morphology, and certain function words.

Figure 1a depicts the LFs for the following Spanish-English pair used in the example below.

En Información del hipervínculo, haga clic en la dirección del hipervínculo.
Under Hyperlink Information, click the hyperlink address.

3 Alignment

We consider an alignment of two logical forms to be a set of mappings, such that each mapping is between a node or set of nodes (and the relations between them) in the source LF and a node or set of nodes (and the relations between them) in the target LF, where no node participates in more than one such mapping.

Our alignment algorithm proceeds in two phases. In the first phase, it establishes tentative lexical correspondences between nodes in the source and target LFs. In the second phase, it aligns nodes based on these lexical correspondences as well as structural considerations. It starts from the nodes with the tightest lexical correspondence (“best-first”) and works outward from these anchor points.

We first present the algorithm, and then illustrate how it applies to the sentence-pair in Figure-1.

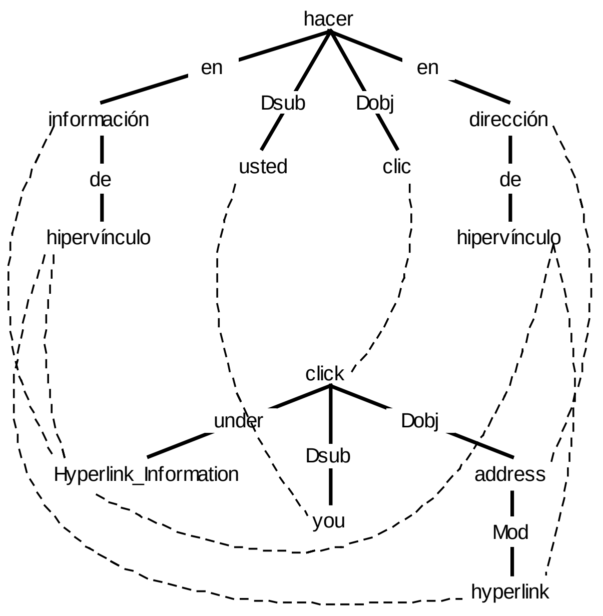


Figure 1a: Lexical correspondences

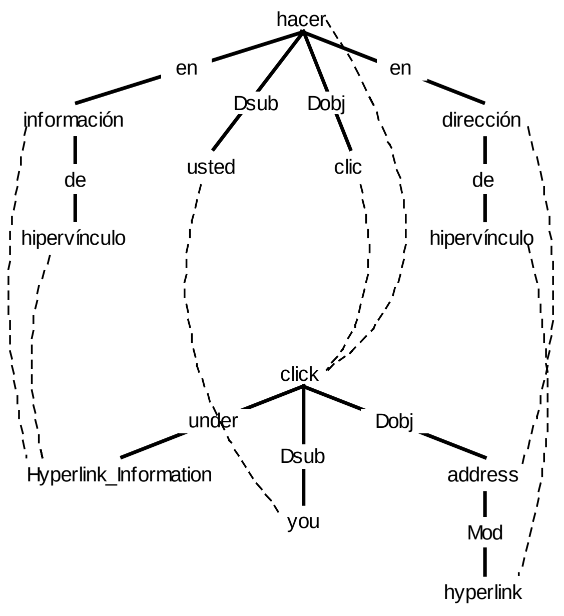


Figure 1b: Alignment Mappings

3.1 Finding tentative lexical correspondences

We use a bilingual lexicon that merges data from several sources (CUP, 1995), (SoftArt, 1995), (Langenscheidt, 1997), and invert target-to-source dictionaries to improve coverage. Our Spanish-English lexicon contains 88,500 translation pairs. We augment this with 9,563 translation correspondences acquired using statistical techniques described in (Moore, 2001).

Like Watanabe (2000) and Meyers (2000) we use this lexicon to establish initial tentative word correspondences. However, we have found that even a relatively large bilingual dictionary has only moderate coverage for our purposes. Hence, we pursue an aggressive matching strategy, using the bilingual dictionary together with the derivational morphology component of our system (Pentheroudakis, 1993). We find direct translations, translations of morphological bases and derivations, and base and derived forms of translations. We find that aggressive over-generation of correspondences at this phase is balanced by the more conservative second phase and results in improved overall alignment quality.

We also look for matches between components of multi-word expressions and individual words. This allows us to align such expressions that may have been analyzed as a single lexicalized entity in one language but as separate words in the other.

3.2 Aligning nodes

Our alignment procedure uses the tentative lexical correspondences established above and structural cues to create affirmative node alignments. A set of alignment grammar rules licenses only linguistically meaningful alignments. The rules are ordered to create the most unambiguous alignments ("best") first and use these to disambiguate subsequent alignments. The rules, intended to be language-neutral, were developed while working primarily with Spanish-English, but have also been applied to other language pairs such as French, German, and Japanese to/from English.

The algorithm is as follows:

1. Initialize the set of unaligned source and target nodes to the set of all source and target nodes respectively.
2. Attempt to apply the alignment rules in the specified order, to each unaligned node or set of nodes in source and target. If a rule fails to apply to any unaligned node or set of nodes, move to the next rule.
3. If all rules fail to apply to all nodes, exit. No more alignment is possible. (Some nodes may remain unaligned).
4. When a rule applies, mark the nodes or sets of nodes to which it applied as aligned to each other and remove them from the lists of unaligned source and target nodes respectively. Go to step 2 and apply rules again, starting from the first rule.

The alignment grammar currently consists of 18 rules, including the following

1. *Bidirectionally unique translation*: A set of contiguous source nodes S and a set of contiguous target nodes T such that every node in S has a lexical correspondence with every node in T and with no other target node, and every node in T has a lexical correspondence with every node in S and with no other source node. Align S and T to each other.
2. *Translation + Children*: A source node S and a target node T that have a lexical correspondence, such that each child of S and T is already aligned to a child of the other. Align S and T to each other.
3. *Translation + Parent*: A source node S and a target node T that have a lexical correspondence, such that a

parent P_s of S has already been aligned to a parent P_t of T. Align S and T to each other.

4. *Verb+Object to Verb:* A verb V_1 (from either source or target), that has child O that is not a verb, but is already aligned to a verb V_2, and either V_2 has no unaligned parents, or V_1 and V_2 have children aligned to each other. Align V_1 and O to V_2.
5. *Parent + relationship*: A source node S and a target node T, with the same part-of-speech, and no unaligned siblings, where a parent P_s of S is already aligned to a parent P_t of T, and the relationship between P_s and S is the same as that between P_t and T.
6. *Child + relationship*: Analogous to previous rule but based on previously aligned children instead of parents.
7. *Verb+Verb to Verb:* A verb V_1 (from source or target) that has no lexical correspondences and has a single child verb V_2 that is already aligned to a verb V_3, where V_3 has no unaligned parents. Align V_1 and V_2 to V_3

Note that rules 4—7 rely solely on relationships between nodes being examined and previously aligned nodes.

3.3 Alignment Example

We now illustrate the application of the alignment procedure to the example in Figure 1. In the first phase, using the bilingual lexicon, we identify the lexical correspondences depicted in Figure-1a as dotted lines. Note that each of the two instances of *hipervínculo* has two ambiguous correspondences, and that while the correspondence from *Información* to *Hyperlink_Information* is unique, the reverse is not. Note also that neither the monolingual nor the bilingual lexicons have been customized for this domain. For example, there is no entry in either lexicon for *Hyperlink_Information*. This unit has been assembled by general rules that link sequences of capitalized words. Lexical correspondences established for this unit are based on translations found for its individual components.

Next, the alignment rules apply as described below. The alignment mappings they create are depicted in Figure-1b as dotted lines.

Rule-1: *Bidirectionally unique translation*, applies in three places, creating alignment mappings between *dirección* and *address*, *usted* and *you*, and *clic* and *click*. These are the initial "best" alignments that provide the anchors from which we will work outwards to align the rest of the structure.

Rule-3: *Translation + Parent*, applies next to align the instance of *hipervínculo* that is the child of *dirección* to *hyperlink*, which is the child of *address*. We leverage a previously created alignment (*dirección* to *address*) and the structure of the logical form to resolve the ambiguity present at the lexical level.

Rule-1 now applies (where previously it did not) to create a many-to-one mapping between *información* and *hipervínculo* to *Hyperlink_Information*. The uniqueness condition in this rule is now met because the ambiguous alternative was cleared away by the prior application of Rule 3.

Rule-4: *Verb+Object to Verb* applies to rollup *hacer* with its object *clic*, since the latter is already aligned to a verb. This produces the many-to-one alignment of *hacer* and *clic* to *click*

4 Acquiring Transfer Mappings

Figure-2 shows the transfer mappings derived from this example.

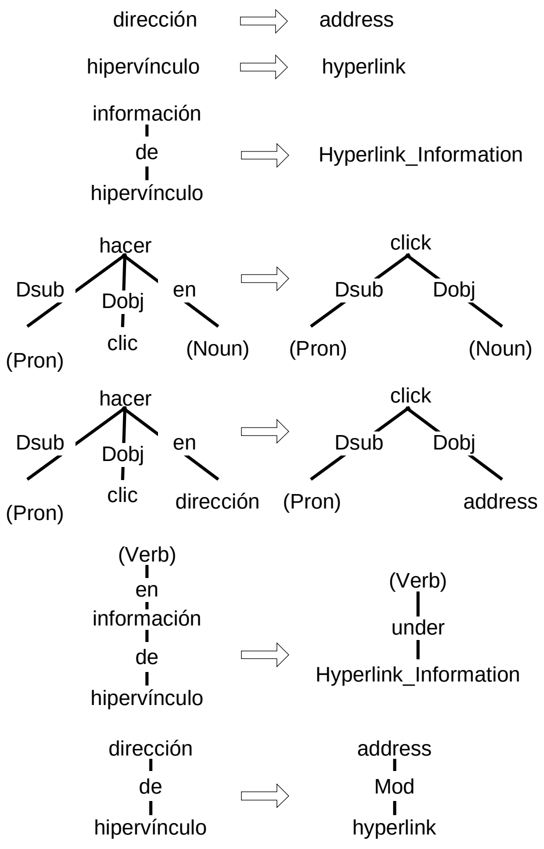


Figure-2 : Transfer mappings acquired

4.1 Transfer mappings with context

Each mapping created during alignment forms the core of a family of mappings emitted by the transfer mapping acquisition procedure. The alignment mapping by itself represents a minimal transfer mapping with no context. In addition, we emit multiple variants, each one expanding the core mapping with varying types and amounts of local context.

We generally use linguistic constructs such as noun and verb phrases to provide the boundaries for the context we include. For example, the transfer mapping for an adjective is expanded to include the noun it modifies; the mapping for a verb may include the object as context; mappings for noun collocations are emitted individually and as a whole. Some mappings may include "wild card" or under-specified nodes, with a part of speech but no lemma, as shown in Figure 2.

4.2 Alignment Post-processing

After we have acquired transfer mappings from our entire training corpus, we compute frequencies for all mappings. We use these to resolve conflicting mappings, i.e. mappings where the source sides of the mapping are identical, but the target sides differ. Currently we resolve the conflict by simply picking the most frequent mapping. Note that this does not imply that we are commited to a single translation for every word across the corpus, since we emitted each mapping with different types and amounts of context (see section 4.1). Ideally at least one of these contexts serves to disambiguate the translation. The conflicts being resolved here are those mappings where the neccesary context is not present.

A drawback of this approach is that we are relying on a priori linguistic heuristics to ensure that we have at least one mapping with the right context. Our future work plans to address this by using machine-learning techniques to find the precise context that serves to optimally disambiguate between conflicting mappings.

4.2.1 Frequency Threshold

During post-processing we also apply a frequency threshold, keeping only mappings seen at least N times (where N is currently 2). This frequency threshold greatly improves the speed of the runtime system, with minor impact on translation quality (see section 5.6).

5 Experiments and Results

5.1 Evaluation methodology

In the evaluation process, we found that various evaluation metrics of alignment in isolation bore very little relationship to the quality of the translations produced by a system that used the results of such alignment. Since it is the overall translation quality that we care about, we use the output quality (as judged by humans) of the MT system incorporating the transfer mappings produced by an alignment algorithm (keeping all other aspects of the system constant) as the metric for that algorithm.

5.2 Translation system

Our translation system (Richardson, 2001) begins by parsing an input sentence and obtaining a logical form. We then search the transfer mappings acquired during alignment, for mappings that match portions of the input LF. We prefer larger (more specific) mappings to smaller (more general) mappings. Among mappings of equal size, we prefer higher-frequency mappings. We allow overlapping mappings that do not conflict. The lemmas in any portion of the LF not covered by a transfer mapping are translated using the same bilingual dictionary employed during alignment, or by a handful of hard-coded transfer rules (see Section 5.7 for a discussion of the contribution made by each of these components). Target LF fragments from matched transfer mappings and default dictionary translations are stitched together to form an output LF. From this, a rule-based generation component produces an output sentence.

The system provides output for every input sentence. Sentences for which spanning parses are not found are translated anyway, albeit with lower quality.

5.3 Training corpus

We use a sentence-aligned Spanish-English training corpus consisting of 208730 sentence pairs mostly from technical manuals. The data was already aligned at the sentence-level since it was taken from sentence-level translation memories created by human translators using a commercial translation-memory product. This data was parsed and aligned at the sub-sentence level by our system, using the techniques described in this paper. Our parser produces a parse in every case, but in each language roughly 15% of the parses produced are “fitted” or non-spanning. Since we have a relatively large training corpus, we apply a conservative heuristic and only use in alignment those sentence-pairs that produced spanning parses in *both* languages. In this corpus 161606 pairs (or 77.4% of the corpus) were used. This is a substantially larger training corpus than those used in previous work on learning transfer mappings from parsed data. Table-1 presents some data on the mappings extracted from this corpus using Best-First.

Total Sentence pairs	208,730
Sentence pairs used	161,606
Number of transfer mappings	1,208,828
Transfer mappings per pair	7.48
Num. unique transfer mappings	437,479
Num. unique after elim. conflicts	369,067
Num. unique with frequency > 1	58,314
Time taken to align entire corpus not including parsing (on a 550MHz PC)	98 minutes
Alignment speed	26.9 pairs/s

Table-1: Best-first alignment of training corpus

5.4 Experiments

In each experiment we used 5 human evaluators in a blind evaluation, to compare the translations produced by the test system with those produced by a comparison system. Evaluators were presented, for each sentence, with a reference human translation and with the two machine translations in random order, but not the original source language sentence. They were asked to pick the better overall translation, taking into account both content and fluency. They were allowed to choose Neither if they considered both translations equally good or equally bad.

All the experiments were run with our Spanish-English system in December 2000. The test sentences were randomly chosen from unseen data from the same domain. Experiment-1 used 200 sentences and every sentence was evaluated by every rater. Sentences were rated better for one system or the other if a majority of the raters agreed. Experiments 2-4 used 500 sentences each, but every sentence was rated by a single rater.

For all experiments, the test system was the system described in section 5.2, loaded with transfer mappings acquired using the techniques described in this paper (hereafter “Best-First”).

System-A	System-B	Num. sentences System-A rated better	Num. sentences System-B rated better	Num. sentences neither rated better	Net percent improved sentences
Best-First	BabelFish	93 (46.5%)	73 (36.5%)	34 (17%)	10%
Best-First	Bottom-Up	224 (44.8%)	111 (22.2%)	165 (33%)	22.6%
Best-First	No-Context	187 (37.4%)	69 (13.8%)	244 (48.8%)	23.6%
Best-First	No-Threshold	112 (22.4%)	122 (24.4%)	266 (53.2%)	-2.0%

Table-2: Translation Quality

In the first experiment the comparison system is a highly rated commercial system, Babelfish (http://world.altavista.com).[1]

Each of the next three experiments varies some key aspect of Best-First in order to explore the properties of the algorithm. The algorithm variations are described in the next section.

5.5 Comparison alignment algorithms

5.5.1 Bottom Up

Experiment-2 compares Best-First to the previous algorithm we employed, which used a bottom-up approach, similar in spirit to that used by Meyers (1998).

This algorithm follows the procedure described in section 3.1 to establish tentative lexical correspondences, however, it does not use an alignment grammar, and relies on a bottom-up rather than a best-first strategy. It starts by aligning the leaf nodes and proceeds upwards, aligning nodes whose child nodes have already aligned. Nodes that do not align are skipped over, and later rolled-up with ancestor nodes that have successfully aligned.

5.5.2 No Context

Experiment-3 uses a comparison algorithm that differs from Best First in that it retains no context (see section 4.1) when emitting transfer mappings.

5.5.3 No Threshold

The comparison algorithm used in Experiment-4 differs from Best First in that the frequency threshold (see section 4.2.1) is not applied, i.e. all transfer mappings are retained.

5.6 Discussion

The results of the four experiments are presented in Table-2.

Experiment-1 establishes that the algorithm presented in this paper automatically acquires translation knowledge of sufficient quantity and quality as to enable translations that exceed the quality of a highly rated traditional MT system. Note however that Babelfish/Systran was not customized to this domain.

[1] We also used the packaged version of the same underlying system from Systran, but found that on our test set it produced inferior translations to those produced by Babelfish, even when using it's computer-domain dictionary. We speculate that the website may represent a newer version of the system.

Experiment-2 shows that Best-First produces transfer mappings resulting in significantly better translations than Bottom-Up. Using Best-First produced better translations for a net of 22.6% of the sentences.

Experiment-3 shows that retaining sufficient context in transfer mappings is crucial to translation quality, producing better translations for a net of 23.6% of the sentences.

Experiment-4 shows that the frequency threshold does not have a statistically significant impact on the translation quality, but as shown in Table-3, results in a much smaller (approx. 6 times) and faster (approx. 45 times) ru`ntime system.

	Num mappings	Translation speed (500 sentences)
Best-First	58,314	173s (0.34 sec/sent)
No-Threshold	359,528	8059s (17 sec/sent)

Table-3: Translation Speed (500 sentences)

5.7 Transfer mapping coverage

Using end-to-end translation quality as a metric for alignment leaves open the question of how much of the translation quality obtains from alignment versus other sources of translation knowledge in our system, such as the bilingual dictionary. To address this issue we measured the contribution of each using a 3264-sentence test set. Table-4 presents the results. The first column indicates the total number of words or relationships in each category. The next four columns indicate the percentage translated using each knowledge source, and the percentage not translated or transferred directly from source to target, respectively.

As the table shows, the vast majority of content words are translated using transfer mappings obtained via alignment. The table also shows the fraction of relationships covered by transfer mappings. Relationships, which are represented in the Logical Form as labels on arcs (see Figure-1) may be labeled with a relationship type (subject, direct object etc) and/or with a preposition.

As the table shows, though over half the relationships in the input are covered by transfer mappings, the system is currently less successful at learning transfer mappings for relationships than it is for content words. As a temporary measure we have 2 hand-coded transfer rules that apply to

	Number of instances	Transfer mappings	Dictionary	Rules	Not translated or direct transfer
Content words	21102	96.3%	2.5%	0%	1.2%
Prepositional relationships	6567	53.6%	39.5%	6.8%	0%
Other relationships	17507	54.2%	0%	0%	45.8%

Table-4: Coverage of transfer mappings, dictionary & rules

some prepositional relationships, which account for 6.9% of such transfers.

6 Absolute Quality

The experiments presented leave two open questions: What is the absolute quality of the translations produced by the system? And what is the relationship between translation quality and the transfer mappings learned by alignment?

To attempt to address these questions we conducted an absolute quality evaluation and investigated the relationship between the transfer mappings used in a translation and the absolute quality of that translation. This evaluation was conducted in April 2001 on a newer version of the same system.

6.1 Methodology

We used 5 human evaluators looking at translations produced from previously unseen data. Evaluators were presented, for each sentence, with a reference human translation and with the machine translation, but not the original source language sentence. They were asked to rate the machine translation on a scale of 1 to 4, taking into account both the fluency of the translation and the accuracy with which it conveyed the meaning (as compared with the reference human translation).

The scores were defined as follows:

4 *Ideal*: Fluent, all information included.
3 *Acceptable*: Comprehensible; all important information accurately transferred.
2 *Possibly Acceptable*: May be interpretable given context and time, some information transferred accurately
1 *Unacceptable*: Not comprehensible and/or little or no information transferred accurately.

The absolute quality evaluation was run for two systems – our own system (using "Best-First") and Babelfish. In each case, the inter-rater agreement was good, though there was some clustering of scores around 2 and 3. The results, presented in Figure-3, confirm the results fof Experiment-1, indicating that our system produces a significantly greater number of translations rated 3.5 to 4.0 than Babelfish.

6.2 Relationship between absolute quality and transfer mappings

The most interesting question for us was what relationship existed between the absolute quality of a given translation and the transfer mappings used to produce that translation.

Towards this end, we computed the following metrics for each group of sentences rated to have similar quality.

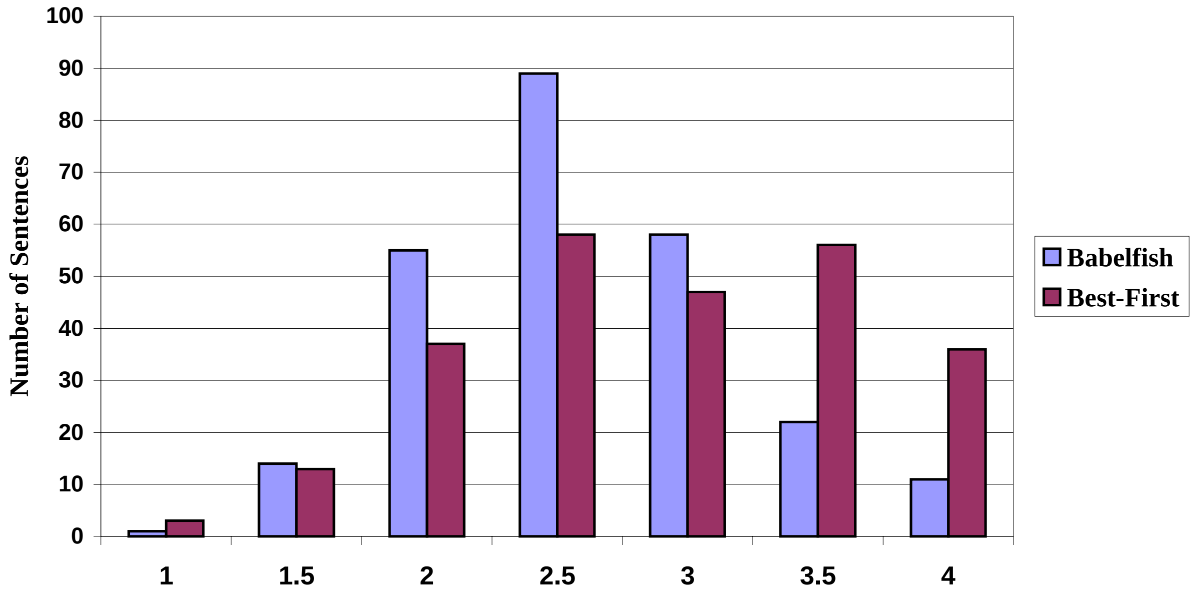


Figure-3: Absolute Quality Score

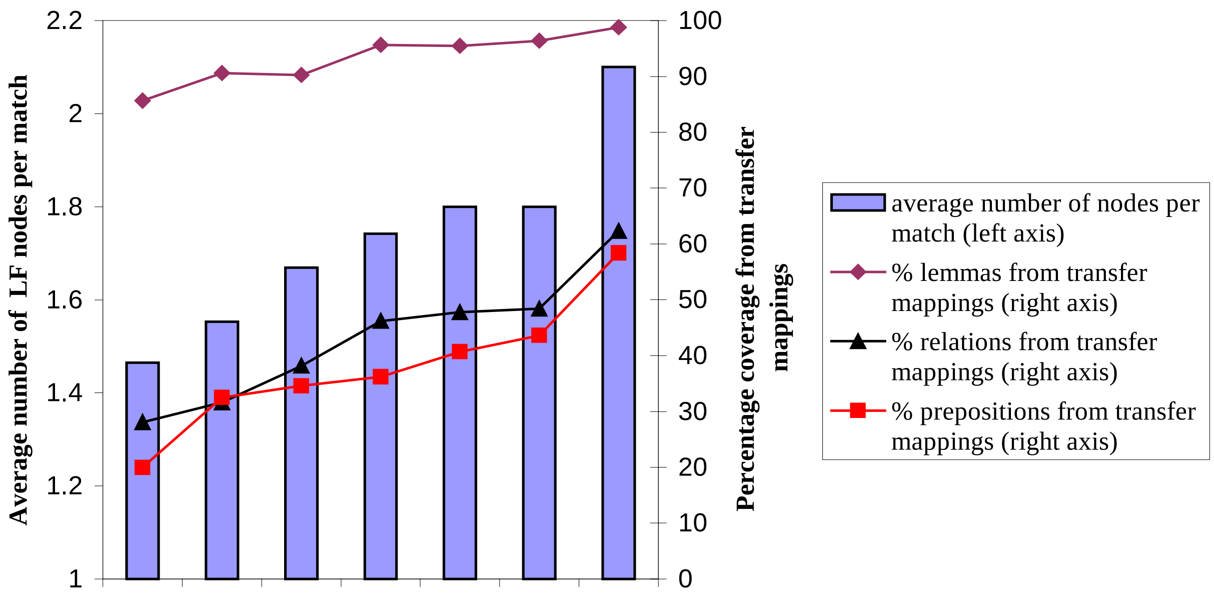


Figure-4: Transfer mapping metrics vs Absolute translation quality score

1) Average number of LF nodes in the transfer mappings used in the translation. The minimum size is 1, representing word-for-word transfers with no larger context.
2) Percentage of lemmas translated using transfer mappings (The remainder are either translated using the bilingual dictionary or left untranslated)
3) Percentage of prepositional relationships covered by transfer mappings (the remainder are translated using the bilingual dictionary)
4) Percentage of other relationships covered by transfer mappings (the remainder are, correctly or incorrectly, transferred from source to target unchanged)

Figure-4 shows these metrics plotted against the absolute quality score of the corresponding sentences. The bars represent the average number of LF nodes per match, which is plotted on the left-hand scale. The lines, plotted against the right-hand scale, represent the percentage of lemmas, prepositions and other relationships, respectively, that were translated using transfer mappings.

The chart shows a strong relationship, in particular, between size of the transfer mappings used and the quality of the translation. The relationship is especially pronounced with the perfect translations. It also indicates that for sentences where most of the prepositions and relationships were not covered by transfer mappings, the absolute quality was rated as low.

On the other hand, the relationship between quality and the percentage of content words translated via transfer mappings is weak, indicating that even within a specific domain, learning word-for-word translations is not enough to ensure quality. Instead larger, more contextual transfers must be learned.

7 Conclusions and Future Work

In this paper, we proposed an algorithm for automatically acquiring high-quality transfer mappings from sentence-aligned bilingual corpora using an alignment grammar and a best-first strategy.

We reported results applying the algorithm to a substantially larger training corpus than that used in previously reported work on learning transfer mappings from parsed data.

We showed that this approach produces transfer mappings that result in translation quality comparable to a commercial MT system.

We also showed that a best-first, alignment-grammar based approach produced better results than a bottom-up approach, and that retaining context in the acquired transfer mappings is essential to translation quality.

We investigated the relationship between absolute translation quality and transfer mappings, and showed that larger more contextual mappings are essential for higher quality.

We currently rely on a priori linguistic heuristics to determine the right context for each transfer mapping. In future work, we plan to use machine-learning techniques to discover the extent of the context that optimally disambiguates between conflicting mappings.

Acknowledgements

We gratefully acknowledge the efforts of all members of the MSR NLP group that participated in this project.

References

Cambridge University Press (1995). McCarthy, M. ed., *Cambridge Word Selector*

Peter Brown, Stephen A. Della Pietra, Vincent J. Della Pietra and Robert L. Mercer (1993). The mathematics of statistical machine translation. *Computational Linguistics*, 19:263-312

Stanley F. Chen (1993). Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st Meeting of the Association for Computational Linguistics (ACL 1993)*

Karen Jensen (1993). PEGASUS: Deriving argument structures after syntax. In *Natural Language Processing: The PLNLP Approach*. Kluwer Academic Publishers, Boston, MA.

Hiroyuki Kaji, Yuuko Kida, and Yasutsugu Morimoto (1992). Learning Translation Templates from Bilingual Text. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING 1992)*

Langenscheidt Publishers 1997, *The Langenscheidt Pocket Spanish Dictionary*

Adam Meyers, Roman Yangarber, Ralph Grishman, Catherine Macleod, and Antonio Moreno-Sandoval (1998a). Deriving transfer rules from dominance-preserving alignments. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING 1998)*

Adam Meyers, Michiko Kosaka and Ralph Grishman, (1998b). A multilingual procedure for dictionary-based sentence alignment. In *Proceedings of the 3rd Conference on Machine Translation in the Americas (AMTA 98)*

Adam Meyers, Michiko Kosaka and Ralph Grishman (2000). Chart-based transfer rule application in machine translation. In *Proceedings of 18th International Conference on Computational Linguistics (COLING 2000)*

Robert C. Moore (2001). Towards a Simple and Accurate Statistical Approach to Learning Translation Relationships among Words *Proceedings of the Workshop on Data-Driven Machine Translation at the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2001)*

Joseph Pentheroudakis and Lucretia Vanderwende (1993). Automatically identifying morphological relations in machine-readable dictionaries. In *Ninth Annual conference of the University of Waterloo Center for the new OED and Text Research*

Satoshi Sato and Makoto Nagao (1990). Towards memory-based translation. *Proceedings of 13th International Conference on Computational Linguistics (COLING 1990)*

Stephen D. Richardson, William Dolan, Monica Corston-Oliver, and Arul Menezes (2001). Overcoming the customization bottleneck using example-based MT. In *Proceedings of the Workshop on Data-Driven Machine Translation at the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2001)*

SoftArt Inc (1995) *Soft-Art translation dictionary. Version 7.*

Hideo Watanabe, Sado Kurohashi, and Eiji Aramaki (2000). Finding Structural Correspondences from Bilingual Parsed Corpus for Corpus-based Translation. In *Proceedings of 18th International Conference on Computational Linguistics (COLING 2000)*

What is Example-Based Machine Translation?

Davide Turcato, Fred Popowich

gavagai Technology Incorporated
420-6450 Roberts Street
Burnaby, British Columbia, Canada V5G 4E1
{turk, popowich}@gavagai.net

Abstract

We maintain that the essential feature that characterizes a Machine Translation approach and sets it apart from other approaches is the kind of knowledge it uses. From this perspective, we argue that Example-Based Machine Translation is sometimes characterized in terms of inessential features. We show that Example-Based Machine Translation, as long as it is linguistically principled, significantly overlaps with other linguistically principled approaches to Machine Translation. We make a proposal for translation knowledge bases that make such an overlap explicit.

Introduction

In an excellent review article about Example-Based Machine Translation (EBMT), Harold Somers (1999) provides a comprehensive classification of the broad variety of MT research falling within the example-based paradigm, and makes an attempt at capturing the essential features that make an MT system an example-based one.

The present paper takes Somers' discussion as its starting point and tries to take further steps in answering the questions posed therein. We acknowledge at this point that we also draw heavily from Somers' paper in terms of citations of previous works in EBMT.

In the broad and diversified panorama of MT, we believe that this definition task, far from being a pedantic exercise, is an important step towards separating essential differences among MT approaches from inessential ones. This effort may lead to uncovering overlaps between approaches that at first sight seem quite far apart, or conversely it may bring to light significant differences between approaches that are superficially similar. We believe that a better understanding of the relations among different approaches provides valuable insight that can guide MT researchers in their decisions about further directions to take.

Classification Criteria

In his apparently provisional conclusions about a definition of EBMT, Somers (1999) discusses three increasingly specific criteria for defining EBMT:

1. EBMT uses a bilingual corpus.
2. EBMT uses a bilingual corpus as its main knowledge base.
3. EBMT uses a bilingual corpus as its main knowledge base, at run-time.

Somers (1999) states that the first two criteria are too broad, but he argues that the third criterion may be too strict, as it rules out, for instance, statistical MT, where all the corpus-driven probabilities are computed in advance. While agreeing with Somers on the inadequacy of the first two criteria, we would like to suggest that the third criterion might also be too broad (disregarding here whether it is at the same time too strict for the reasons put forward by Somers). In the following sub-sections we discuss the proposed criteria.

Implicit vs. Explicit knowledge

One of Somers' criteria is that an example database be used at run-time. As far as we can see, there are two reasons why a corpus is used at run-time in an MT system:

1. The system uses knowledge that can only be dynamically acquired at run-time by accessing an entire corpus, or sections of it whose extent cannot be determined in advance.
2. The system uses knowledge that could be extracted in advance, but is instead left implicit in the corpus, and extracted as needed at run-time.

We argue that only the former case is relevant to the above-mentioned criterion for characterising EBMT. We adopt here the software engineering perspective of separating processes from data, and we focus on data. Obviously, there may be reasons for preferring to leave knowledge implicit rather than making it explicit (in terms of efficiency, memory requirements, time/space trade-off, etc.). However, from the point of view of characterising an approach to MT, such as taken here, what we regard as essential is ascertaining what kind of knowledge a given approach uses, as opposed to whether the same body of knowledge is explicitly or implicitly encoded.

Analogously, once a body of knowledge is explicitly encoded, we regard the source from which it was acquired as secondary, for classification purposes. Of course, we do not intend to overlook the issue of knowledge acquisition, both in terms of cost-effectiveness and system coverage. However, we maintain that this issue is not crucial in classifying an MT system. To give an example coming from direct experience, in our English-Spanish lexicalist transfer system (Popowich et al., 1997) we initially handcrafted our bilingual lexicon. Subsequently, we developed tools for the automatic acquisition of relevant terms from corpora and for the automatic or semi-automatic generation of bilingual lexicons (Turcato et al., 2000a, and references therein). Although we obviously considered this a major achievement, nevertheless we did not feel that this made our MT system more example-based than it was before.

Single vs. Multiple Knowledge Sources

Another criterion stated by Somers is that the example database be the main knowledge base used by a system. This is a point often emphasised in the EBMT literature. One of its corollaries is that a system's accuracy can be increased by simply adding more examples.

One preliminary remark is that accounts of EBMT systems tend sometimes to overlook or understate the use they make of other resources, besides an example database. For example, most EBMT systems assume the existence of a bilingual lexicon to perform substitutions in examples.

To give another example, one of the most characteristic operations performed in EBMT, the *similarity comparison*, is usually driven by a thesaurus. Of course, the availability of thesauri does not make an MT approach more data-driven or knowledge-free than if a thesaurus had to be specifically developed by hand. Moreover, such resources do not readily lend themselves to the porting of an MT system to a specific domain. Work on semantic databases like WordNet has shown that much of their information can be misleading in specific domains. For example, an MT system dealing with weather reports would have serious problems using a thesaurus where very frequent words like *snow* and *C* (for *Celsius*) were considered semantically similar because they are both synonyms for *cocaine* (Turcato et al., 2000b).

Finally, several other kinds of linguistic processing are performed in EBMT, ranging from named entity recognition (Brown, 1999), morphological analysis, and tagging to full parsing. In other words, most of the linguistic processing techniques used in conventional MT systems have been proposed in EBMT systems. Each of these processes requires some sort of linguistic resource.

A further and perhaps more crucial remark is that EBMT approaches tend to use the same resource (i.e. an example database) for different purposes, while traditional MT systems tend to use different resources. E.g., target sentences are used as sentence template for the recombination task, via some kind of substitution. The recombination task parallels generation, for which many other systems used specific target grammars. Analogously to what is argued about the implicit vs. explicit encoding of knowledge, we maintain that what is relevant here is to ascertain what kind of knowledge is used for the recombination/generation task. Again, there may be practical arguments for preferring a single resource to separate resources, or vice versa. However, what affects the output of a system is the knowledge used for a task, not the integration vs. segregation of this knowledge with respect to other knowledge sources.

A Declarative Classification Criterion

To sum up, in classifying an MT approach, we believe it is useful to separately ask the following three questions:

1. What linguistic information is used?
2. Where is linguistic information acquired from?
3. When is linguistic information acquired?

We claim that only the first question should be the primary focus of a classification, while the differences in terms of the other two questions should be regarded as secondary. In other words, we suggest a declarative criterion that looks at the knowledge being used, rather than at the processes used to obtain that knowledge.

We illustrate this point with an example. In presenting their *Gaijin* EBMT system, Veale & Way (1997:239) claim that “the only linguistics employed by Gaijin is a psycholinguistic constraint – the marker hypothesis”. However, in describing the system they explain that they use a bilingual lexicon, statistically constructed by corpus word alignment. They also explain that “Gaijin employs corpus-based statistics not as a translation strategy in themselves, but as a basis for inferring symbolic transfer rules” (p. 240). So, in answering our first question, we would say that Gaijin uses a bilingual lexicon and a set of symbolic transfer rules, besides knowledge about phrase markers. In addition, one might question whether the mere fact that Gaijin acquires its linguistic information from a corpus makes it an EBMT system.

Arguably, two approaches can be regarded as one and the same approach if they use the same knowledge in the same way, regardless of whether such knowledge is extracted in advance or at run-time, from a corpus or from other resources, or whether it is distributed over one or several resources. All variants ultimately behave in the same way.

To look at the same issue from a slightly different perspective, we tend to consider two systems that perform full syntactic analysis more similar among themselves, regardless of whether this information is encoded in a grammar or a tree-bank, than each of them is to a system that only performs, say, morphological analysis.

EBMT Re-assessment

Equipped with this criterion that prioritises the knowledge content of a system over the way knowledge is expressed or acquired, we turn to a tentative review of EBMT based on this criterion. Our goal is not so much to give an absolute definition of what ‘true’ EBMT is, but rather to assess the proximity of approaches that look superficially distant, or conversely to assess the distance between systems that look superficially similar. We will draw a comparison between linguistically principled EBMT system and other linguistically principled approaches to MT. Our comparison will sometimes linger on a specific transfer approach, the lexicalist variant, particularly when the discussion concerns translation selection. This preference is contingently motivated by the fact that lexicalist transfer is the symbolic approach we are most familiar with. However, we think that such a comparison bears a general significance, for the following reasons:

1. Lexicalist transfer, in its turn, considerably overlaps with other symbolic approaches, not limited to the class of transfer approaches.
2. If we show that EBMT (or a subclass of it) is equivalent to (or significantly overlapping with) a symbolic approach that is customarily characterised in terms that make no reference to example bases, then we are led to either:
   a) draw the somehow paradoxical conclusion that EBMT can be characterised in terms that make no reference to example bases; or
   b) conclude that the characterisation of example-based approaches rests more upon knowledge acquisition aspects than upon knowledge usage aspects. This, in turn, suggests that the example-based characterisation could be reformulated as a transversal distinction between data-driven vs. theory-driven approaches that would cut across multiple linguistically principled approaches, rather than being a separate approach on a par with the others. So, one could imagine a data-driven lexicalist transfer vs. a theory-driven lexicalist transfer, a data-driven structural transfer vs. a theory-driven structural transfer, etc., depending on how knowledge is acquired.

We finally note that in our review we will try to leave out systems that are explicitly claimed to be hybrid by their authors, narrowing down our scope to systems that are claimed to be variants of the EBMT approach.

Non-symbolic EBMT

A fundamental distinction exists between systems that use linguistic knowledge and systems that do not. Statistical MT falls in the latter class. We leave open the issue whether statistical MT is a kind of EBMT. In any case, it is clear that the methods of such approaches set them apart as much from linguistically principled EBMT as from other kinds of linguistically principled MT.

We illustrate this point by discussing a sample statistical MT system, the French-English system Candide (Berger et al., 1994). Candide is described in terms of three components resembling the traditional partition of transfer MT: analysis, transfer, and synthesis. Analysis maps source French sentences onto what the authors call "intermediate French", i.e. normalised representations of sentences. Normalisation consists of case and spelling correction, name and number detection, segmentation, morphological analysis and word reordering. Conversely, synthesis maps "intermediate English" representations onto target sentences. Although these components do use some linguistic knowledge, the knowledge they use is low-level. Their task is to normalise the input and the output, rather than to add any linguistic knowledge to be used in transfer. The input and the output of transfer are just strings, and transfer is purely statistical. Its task is described as decoding an English sentence that was transmitted over a noisy communication channel, which "corrupted" it to a French sentence.

Transfer uses two knowledge sources: a *language model* of English, and a *translation model*. A language model is used to assign probabilities to English sequences of words, and it simply consists of a set of trigrams (i.e. triples of English words), with associated probabilities (i.e. numeric values). No other knowledge about words is used. A translation model is used to compute the conditional probability of an English sentence, given a French sentence and an alignment (an exhaustive mapping of word positions between two sentences). Several translation models are proposed. The simplest consists of a set of word translation probabilities (i.e. pairs of English and French words, to which numeric values are assigned). The other translation models are more refined in terms of more sophisticated parameter estimations, but none of them makes use of any additional linguistic knowledge.

Summing up, besides the result of alignment, which can be considered a word-based probabilistic bilingual lexicon (analogously to what was previously discussed for the Gaijin system), it can be seen here that transfer uses none of the other linguistic knowledge sources (either monolingual or bilingual) used by linguistically principled EBMT or other symbolic approaches.

Symbolic EBMT

The present section expands and generalizes some remarks we made in (Turcato et al., 1999), where we attempted a comparison of different kinds of EBMT systems with lexicalist transfer MT. In that paper we remarked, as also mentioned above, that in EBMT an example database is used for different purposes at the same time: as a source of sentence frame pairs, and as a source of sub-sentential translation pairs. Accordingly, Somers (1999) points out that EBMT comprises three phases (*matching*, *alignment*, and *recombination*), and draws a parallel with analogous phases in traditional transfer MT systems (*analysis*, *transfer*, and *generation*). Sentence frames are used in matching and recombination as aligned basic sentence structures to be combined with aligned sub-sentential translation pairs. In the following two sections we discuss the two key operations of EBMT, *sentence decomposition* and *translation selection*.

Sentence Decomposition

Given the obvious fact that exact match would be too strict a requirement, a match between dissimilar sentences is generally obtained by decomposing them into some kind of constituents, in order to perform a partial match, in which dissimilar parts are substituted.

Proposals about how to decompose examples vary considerably. In most cases, some sort of linguistic analysis is used. The range of proposed techniques includes, just to name a few: segmenting sentences using markers as segment boundaries (Veale & Way, 1997), performing named entity recognition to obtain more abstract examples (Brown, 1999), using morphologically analyzed segments (Kitano, 1993), using tagged sequences as fragments (Somers et al., 1994), parsing sentences into dependency trees (Sato & Nagao, 1990). In brief, the whole range of available linguistic techniques is used for this purpose. In some cases it is proposed to explicitly store generalized examples (Furuse & Iida, 1992), in other cases examples are decomposed on the fly. In any case, the idea behind all proposals is that examples can be decomposed into smaller constituents to be processed independently.

While there is obviously a remarkable difference in analyzing power between performing a simple detection of segment boundaries or morphological analysis and performing full parsing, the common idea of decomposing sentences into constituents resembles the idea that underlies grammars. In fact, each of the proposed techniques for decomposing sentences parallels some corresponding kind of grammar in a conventional MT system. In each case, the knowledge used for decomposition could be explicitly stored as a separate set of rules. For example, in (Toole et al., 1999) we describe a grammar in terms of a flat list of tag sequences, used to cover a segmented input sentence.

In brief, we would like to argue that some of the techniques used in EBMT for sentence decomposition are farther apart from each other than each of them is from the corresponding technique used (or usable) in a conventional transfer system. This latter similarity would stand out more clearly if sentence templates were separately stored, as proposed in some EBMT systems, instead of being extracted on-the-fly. It is not apparent that there is any compelling reason ruling out this option.

We make a final remark about the advantage of easily handling structural mismatches, ascribed to EBMT. Although this is certainly true of EBMT, since no recursive transfer is performed, this is a property that EBMT shares with several other proposals not only in transfer approaches, such as lexicalist MT or semantic transfer (Dorna et al., 1998), but also in the interlingua approach, such as proposed by Traum & Habash (2000).

Translation Selection

Given that exact match of complete examples is the exception rather than the rule, and translation is usually done by decomposing a sentence into constituents, two questions arise: (i) to what extent the availability of complete examples is needed in translating constituents? (ii) how does EBMT translation of constituents differ from other symbolic approaches to MT (e.g. lexicalist transfer)? We discuss the two issues by putting forward a proposal for a translation knowledge base, then discussing how such a knowledge base would suit EBMT.

Paraphrasing the definition of grammars as descriptions of infinite sets of sentences by finite means, we can analogously define the translation task as the description of an infinite set of equivalencies by means of a finite set of equivalencies.

The translation task can be defined in terms of two fundamental properties:

1. Translation is *compositional*. The translation of an expression is a function of the translation of its constituents. E.g. the Spanish translation of *business trip* (*viaje de negocios*) is a function of the translations of *business* (*negocios*) and *trip* (*viaje*), when these appear in isolation. In turn, the translation of *a long business trip* (*un viaje de negocios largo*) is a function of the translations of *a* (*un*), *long* (*largo*) and *business trip*.
2. Translation is *non-monotonic*. In specific context, compositionality holding for narrower contexts is reversed. E.g. the Spanish translation of *field trip* (*viaje de estudio*) is not a function of the translations of *field* (*campo*) and *trip*, when these appear in isolation. However, the translation of *a long field trip* (*un viaje de estudio largo*) is indeed a function of the translations of *a* (*un*), *long* (*largo*) and *field trip*.

Accordingly, the knowledge that a bilingual knowledge base should contain can be declaratively stated as follows:

1. A repository of basic translation equivalencies, i.e. a bilingual lexicon of word-to-word equivalencies. This bilingual lexicon would define the base step in a recursive, compositional translation process. E.g.
   business ↔ negocios
   viaje ↔ trip
   field ↔ campo
   long ↔ largo
2. A repository of phrases that do not translate compositionally. This repository would define all and only translation 'turning points', which violate the monotonicity of compositional translation. E.g.
   field trip ↔ viaje de estudio
   Note that the phrases in this repository could in turn be part of a compositional translation.

This proposed distinction resembles the distinction between *general examples* and *exceptional examples* proposed by Nomyiama (1992), cited by Somers (1999:121). The sum of the two knowledge bases would account for any possible translation. In other words, it would be a finite set of equivalencies that account for an infinite set of translations.

The repository of non-monotonic contexts would serve one of the main purposes of example databases, i.e. identifying the appropriate translation of a particular word or phrase, based on its context. However, it would be more specific and explicit in two ways:

1. It would specify minimal contexts, i.e. just enough context as necessary to trigger a non-compositional translation. Consider an example database like the following:
   I read an article about your business trip ↔ Leí un artículo sobre tu viaje de negocios
   I gave up my field trip ↔ Renuncié a mi viaje de estudio
   An input sentences like *I read an article about your field trip* would probably match the first example, because of irrelevant context, and wrongly replace the translation of *business* with the translation of *field*.
2. It would eliminate redundancies (or, in EBMT terms, *example interference)*. Each context would be specified only once.

Of course, one might argue that a knowledge base of this kind is a significant departure from an example database, and requires extra work. This may well be true. However, the point we are trying to make is that there is no inherent reason why example decomposition could not be done off-line and explicitly expressed in the knowledge base. If an example database defines a finite set of equivalencies, then this set of equivalencies can be explicitly stated (with each equivalence stated only once).

A knowledge base as we propose would be compatible with different approaches to MT. It would equally be equivalent to an EBMT example database (under some specified conditions), and to a lexicalist transfer bilingual lexicon. Declaratively stated, the linguistic information used by EBMT is indistinguishable from the information used by lexicalist MT. The evidence explicitly stated in a knowledge base is the same for the two approaches. The same knowledge base can be used by both.

Where would EBMT and lexicalist MT differ then? If their knowledge base was complete, they would not differ. Where the two perspectives would differ is in dealing with incompleteness. A lexicalist, bottom up approach, emphasizes compositionality, while an example-based, top down approach, emphasizes non-monotonicity. The former is more likely to miss relevant contexts that override compositionality. The latter is more likely to miss lexical generalizations that hold across different contexts. However, the evidence provided by the way either approach deals with incompleteness is a much weaker argument than the evidence either approach could put forward if they used different kinds of knowledge.

Translation by analogy

We have intentionally postponed the discussion of translation by analogy until after the proposal put forward in the previous section. A classical example of translation by analogy is the one discussed by Nagao (1984), who shows a method for translating new sentences, based on their similarity with available examples. Similarity is measured by the distance of words in a thesaurus (although other methods could be devised). For instance, Nagao shows how the two English-Japanese examples

A man eats vegetables ↔ Hito wa yasai o taberu
Acid eats metal ↔ San wa kinzoku o okasu

can be used to translate the new sentence

He eats potatoes

provided that a bilingual lexicon at the word level is available. The problem here is to choose one of two competing translations for *eat* (*taberu* vs. *okasu*). In Nagao's approach, the translation *taberu* is correctly selected, based on the greater semantic similarity of *he* and *potatoes* with *man* and *vegetables*, respectively, than with *acid* and *metal*. Conversely, the occurrence of *eat* in

Sulfuric acid eats iron

is correctly translated by *okasu*, based on the greater semantic similarity of *sulfuric acid* and *iron* with *acid* and *metal*, respectively, than with *man* and *vegetables*.

If we rephrase the translation selection problem in the terms described in the previous section, assuming one of the two translations as the "default" translation for *eat* (most likely *taberu*, which translates the more literal sense), the task is to identify the non-monotonic contexts that require a different translation from the default one. In doing this, it would be desirable to find the minimal context that triggers a given translation, so as to use it in a broader range of cases. It turns out that things stand differently, depending on whether exact match of contexts or match by analogy is performed.

When exact match of contexts is performed, an example (e.g. *a man eats vegetables*) only accounts for itself, i.e. it is only used when the same example, or a part of it, is input (e.g. *a man eats…* or *…eats vegetables*). In other words, given a set of examples, it is known in advance what input sentences they can be useful for. In this situation it is conceivable to use a contrastive method to reduce a set of overlapping examples to a set of minimal local contexts accounting for different translations of the same term. E.g. the set of examples

A man eats vegetables ↔ Hito wa yasai o taberu
Acid eats metal ↔ San wa kinzoku o okasu
He eats potatoes ↔ Kare wa jagaimo o taberu
Sulfuric acid eats iron ↔ Ryūsan wa tetsu o okasu

could be reduced to the following set of minimal contexts (omitting function words for simplicity):

man & eat ↔ hito & taberu
acid & eat ↔ san & okasu
he & eat ↔ kare & taberu
sulfuric acid & eat ↔ ryūsan & okasu

This would be done on the basis of a generalization that ascribes the selection of one or the other translation of *eat* to its subject (alternatively, one could choose to generalize over the objects, or perhaps to do no generalization). The set of contexts can be further reduced by identifying a default translation for *eat* and only explicitly encoding non-monotonic contexts:

eat ↔ taberu
acid & eat ↔ san & okasu
sulfuric acid & eat ↔ ryūsan & okasu

When contexts are matched by analogy, a given example (e.g. *a man eats vegetables*) may account not only for itself, but also for previously unseen contexts (e.g. *he eats potatoes*). Conversely to that discussed for exact matches, given a set of examples, it is not known in advance what input sentences the examples can be useful for. In principle, an example might be used for an input sentence with no words in common, as long as there is a term-to-term semantic similarity between the example and the input sentence. Given an example like *a man eats vegetables*, how does one determine in advance what is the relevant context for an unforeseen translation? *A man eats* or *eats vegetables*? Or the two together? Because of the incompleteness assumption, extracting local contexts from examples becomes problematic. Given a set of examples, one can know what local contexts would adequately cover the examples at hand, but one cannot know whether such contexts would be adequate for all possibly relevant sentences. For instance, a sentence like *moths eat holes in clothes* may require that a larger context be taken into account, for a more informed similarity assessment. Therefore, it is advisable to have the largest possible contexts available (i.e. entire examples), so as to be able to use different portions of them depending on the input sentence to be translated.

A traditional counterpart of this selection mechanism would be to abstractly encode contexts by means of some sort of semantic features that would act as selectional restrictions. E.g. the distinction between the two senses of *eat* (respectively translated by *taberu* and *okasu*) might be captured by a [± animate] feature, which would select the appropriate subject. This approach would make a database of explicit contexts superfluous, and would also lend itself to a similarity-based approach, if selectional restrictions were used as preferences rather than hard constraints. However, this approach would be labor-intensive and, as Somers (1999:127) points out, it would be "cumbersome and error-prone".

From this informal discussion it appears that translation by analogy, which is the most characteristic technique of EBMT, is also the one where the use of entire examples is most motivated. As a final remark, we only note that an example database for the purpose of translating by analogy can be an additional resource to whatever other resources are used, along the lines discussed above. In principle, translation by analogy could also be an extension to a traditional transfer MT system, to solve cases of lexical ambiguity for which no direct evidence is found in a translation database.

Conclusions

Classifications of EBMT systems often tend to bring to the foreground the fact that such systems use a bilingual corpus as the source of their linguistic knowledge, relegating to the background the assessment of what linguistic knowledge a system actually uses.

We propose a classification criterion that reverses the terms of the question, focusing primarily on the linguistic knowledge used by a system, and considering the source of this knowledge or the format in which it is represented as secondary. Accordingly, we make a distinction between the use of a corpus as: (i) a mere source for knowledge acquisition; (ii) a mere way of storing knowledge in a compact form; (iii) a genuine repository of knowledge that the system needs to access, and that could not be easily stored in other forms. A preliminary application of this criterion to linguistically principled EBMT approaches shows that:

- An example database is usually only one of the resources used by an EBMT system.
- The allegedly holistic approach of EBMT can be equivalently decomposed in a series of distinct knowledge sources and related tasks.
- The amount of linguistic knowledge required for any given component varies considerably for different approaches, ranging from low-level information like part of speech to full syntactic information.
- For most components a counterpart can be found in more conventional approaches to MT that performs an equivalent task without the need to access a database of full examples.
- The original idea of translation by analogy stands out as truly example based.

We feel that clarifying the linguistic knowledge required by an MT approach and making the overlap with other approaches explicit is important. MT is a complex task that requires a vast amount of knowledge, linguistic and possibly extra-linguistic, to be performed adequately. We feel there is no shortcut to overcome this requirement. Hence it is important for MT practitioners to emphasize commonalities among different approaches, and put some effort towards integrating and sharing resources.

References

Berger, A. L., Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., Gillett, J. R., Lafferty, G. D., Mercer, R. L., Printz, H. & Ureš, L. (1994). The Candide System for Machine Translation. In Proceedings of the 1994 ARPA Workshop on Human Language Technology, Plainsboro, NJ, USA.

Brown, R. D. (1999). Adding Linguistic Knowledge to a Lexical Example-based Translation System. In Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (pp. 22--32). Chester, UK.

Dorna, M., Frank, A., van Genabith, J. & Emele, M. C. (1998). Syntactic and Semantic Transfer with F-Structures. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (pp. 341—347). Montréal, Canada.

Furuse, O. & Iida, H. (1992). Cooperation between transfer and analysis in example-based framework. In Proceedings of the Fifteenth [sic] International Conference on Computational Linguistics (pp. 645--651). Nantes, France.

Kitano, H. (1993). A comprehensive and practical model of memory-based Machine Translation. In Proceedings of the 13th International Joint Conference on Artificial Intelligence (pp. 1276--1282). Chambéry, France.

Nagao, M. (1984). A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. In A. Elithorn & R. Banerji (Eds.), Artificial and Human Intelligence: edited review papers at the International NATO Symposium on Artificial and Human Intelligence sponsored by the Special Programme Panel held in Lyon, France October, 1981 (173--180). Amsterdam, North-Holland, Elsevier Science Publishers.

Nomyiama, H. (1992). Machine Translation by Case Generalization. In Proceedings of the Fifteenth [sic] International Conference on Computational Linguistics (pp. 714--720). Nantes, France.

Popowich, F., Turcato, D., Laurens, O., McFetridge P., Nicholson, J. D., McGivern, P., Corzo-Pena, M., Pidruchney, L. & McDonald, S. (1997). A Lexicalist Approach to the Translation of Colloquial Text. In Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation (pp. 76--86), Santa Fe, New Mexico, USA.

Sato, S. & Nagao, M. (1990). Toward memory-based translation. In Proceedings of the 13th International Conference on Computational Linguistics (vol. 3, pp. 247--252). Helsinki, Finland.

Somers H. (1999). Review Article: Example-based Machine Translation. Machine Translation, 14(2), 113--157.

Somers, H., McLean, I. & Jones, D. (1994). Experiments in Multilingual Example-Based Generation. In Proceedings of the 3 rd Conference on the Cognitive Science of Natural Language Processing. Dublin, Ireland.

Toole, J., Popowich, F., Nicholson, D., Turcato, D. & McFetridge, P. (1999). Explanation-Based Learning for Machine Translation. In Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (pp. 161--171). Chester, UK.

Traum, D. & Habash, N. (2000). Generation from Lexical Conceptual Structures. In Proceedings of the Third Workshop on Interlinguas and Interlingual Approaches. Seattle, Washington, USA.

Turcato, D., McFetridge, P., Popowich, F. & Toole, J. (1999). A Unified Example-Based and Lexicalist Approach to Machine Translation. In Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (pp. 33--43). Chester, UK.

Turcato, D., Popowich, F., McFetridge, P. & Toole, J. (2000a). Creating high-quality, large-scale bilingual knowledge bases using minimal resources. In Proceedings of the LREC-2000 Workshop on `Developing Language Resources for Minority Languages: Reusability and Strategic Priorities' (pp. 92—99). Athens, Greece.

Turcato, D., Popowich, F., Toole, J., Fass, D., Nicholson, D. & Tisher, G. (2000b). Adapting a synonym database to specific domains. In Proceedings of the ACL'2000 Workshop on Recent Advances in Natural Language Processing and Information Retrieval, Hong Kong.

Veale, T. & Way, A. (1997). Gaijin: A Bootstrapping Approach to Example-Based Machine Translation. In Proceedings of the International Conference ‘Recent Advances in Natural Language Processing’ (239—244). Tzigov Chark, Bulgaria.

Beyond Translation Memories

Reinhard Schäler

Localisation Research Centre (LRC)
Department of Computer Science and Information Systems (CSIS)
University of Limerick
Limerick
Ireland
Reinhard.Schaler@ul.ie

Abstract

One key to the success of EBMT is the removal of the boundaries limiting the potential of translation memories. To bring EBMT to fruition, researchers and developers have to go beyond the self-imposed limitations of what is now traditional, in computing terms almost old fashioned, TM technology. Experiments have shown that the probability of finding exact matches at phrase level is higher than the probability of finding exact matches at the current TM segment level. We outline our implementation of a linguistically enhanced translation memory system (or *Phrasal Lexicon*) implementing phrasal matching. This system takes advantage of the huge and underused resources available in existing translation memories and develops a traditional TM into a sophisticated example-based machine translation engine which when integrated into a hybrid MT solution can yield significant improvements in translation quality.

Translation Memories and EBMT

Having kept translators up at night worried about the future of their profession, machine translation as we knew it can now safely be declared obsolete and dead. However, since this initial threat to well established work practices, translation has never been the same. New developments in computer assisted translation, more specifically the emergence of translation memory (TM) applications, have continued to keep translators on their guard.

TMs – sophisticated search & replace engines?

Initially, computational linguists often chose to simply ignore TM technology – considering it as some type of sophisticated search and replace engine, not a subject for serious research efforts.
The developers of these systems on the other hand, many of them coming from an academic background and therefore being familiar with the latest computational linguistic research developments, recognised the value of existing research and decided that it was time to apply it in practice.
For example: bilingual text alignment – this problem was declared as largely 'solved' in the early 90s by Gale and Church (1991) but never applied and proven in practice until the alignment utilities of translation memory developers some years later.[1]
Only recently, and driven by increased activities in the area of example-based machine translation (EBMT), has the interest shown by the linguistic tools industry in research results been reciprocated by the research community.
One possible reason for this development is that although EBMT as a paradigm has been described in research papers as far back as 1984 (Nagao etc.) and although it managed to capture the interest and enthusiasm of many researchers it has, so far, failed to reach the level of maturity where it could be transformed from a research topic into a technology used to build a new generation of machine translation engines – and new approaches, technologies and applications are badly needed in MT.

Unlocking the potential of TMs

We believe that the time is ripe for the transformation of EBMT into demonstrators, technologies and eventually commercially viable machine translation engines along the lines suggested by Schäler (1996) and Macklovitch (2000) which are both based on the believe that existing translations contain more solutions to more translation problems than any other available resource (Isabelle et al., 1993).
The key to the success of this development, we suggest, is the removal of the boundaries limiting the potential of translation memories. To bring EBMT to fruition, researchers and developers have to go beyond the self-imposed limitations of what is now traditional, in computing terms almost old fashioned, TM technology.

EBMT and the Phrasal Lexicon

EBMT has been proposed as an alternative and replacement for RBMT, initially by (Nagao, 1984), followed by extensions reported in (Sato & Nagao, 1990) and (Sadler & Vendelmans, 1990). EBMT has also been proposed as a solution to specific translation problems, as reported in (Sumita & Iida, 1991).
The enormous variety of approaches to, the focus of, and the motivations for the use of examples in natural language processing (NLP) are testimony to the high level of interest in EBMT. Taking existing parallel texts as their starting point, researchers have worked on:[2]

- Word-sense disambiguation (Brown et al., 1991) and translation ambiguity resolution (Doi, 1992) and (Uramoto, 1994);

[1] It is interesting to note here that what was deemed to be solved in theory turned out to present quite considerable problems when applied in practice (Schäler, 1994)

[2] The work quoted in the following bullet list can, unfortunately, not be fully referenced in this article, for practical reasons. Most of the reports mentioned were published in the proceedings of ANLP, COLING and ACL.

- Lexicography, e.g. the identification and translation of technical terminology (Dagan and Church, 1994), the development of an instant lexicographer (Karlgren, 1994); generally, the acquisition of lexical knowledge through the structural matching of bilingual sentences (Utsuro et al., 1994);
- Extraction of bilingual collocations or translation patterns from parallel corpora, non-aligned as in (Fung, 1995), (Rapp, 1995) and (Tanaka and Iwasaki, 1996), or aligned and using a linguistic (Matsumoto et al., 1993), statistical (Kupiec, 1993; Smadja et al., 1991; Smadja, 1993; Smadja et al.1996), or, indeed, a combined approach (Kumano and Hirakawa, 1994); special attention to the problems of extracting bilingual collocations for Asian languages is given by (Haruno, 1996) and (Shin, 1996);
- Translation Quality Measures (Su et al., 1992);
- Extensions to and variations of the basic idea of EBMT, proposing Pattern-based Machine Translation (Maruyama, 1993) and (Takeda, 1996), Transfer-Driven Machine Translation (Furuse and Iida, 1994), Statistical Machine Translation (Brown et al. 1993), Machine Translation based on Translation Templates (Kaji et al., 1992) and (Kinoshita, 1994), and Translation Patterns (Watanabe 1993) and (Watanabe, 1994).

One idea, however, which precedes all of the approaches mentioned and which, surprisingly, has so far not been taken up by researchers to any significant degree, that of the Phrasal Lexicon, described by Joseph Becker (1975).

The Phrasal Lexicon

"Like all other scientists, linguists wish they were physicists. They dream (...) of having language behave in an orderly way so that they could discover the Universal Laws behind it all. Linguists have a problem because language just ain't like that." (Becker, 1975:70)

Phrases as building blocks

Becker proposes a model radically different from that of mainstream linguistics as it is known since the mid-seventies: instead of considering language production as the process of combining units the size of words or smaller to form utterances, he heads in the opposite direction identifying phrases consisting of more than one word as the building blocks for the formation of utterances.

His thesis is that humans produce language mostly by repetition, modification and concatenation of previously-known phrases which are adapted to new situations during the productive process. While he concedes that *generative* processes ("generative gap filling") play an important role in language production, he believes that there is sufficient evidence to suggest that the use of language is at least as much based on memorization as on the generation of novel utterances because many situations do not demand novelty, but "rather an appropriate combination of formulas, clichés, idioms, allusions, slogans, and so forth".

Becker's Theory of Language Production

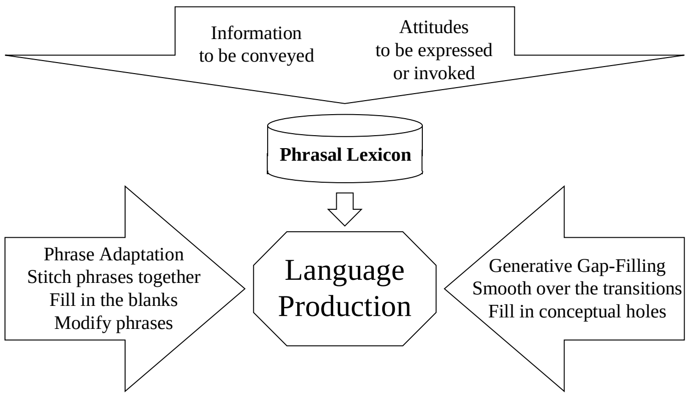


Figure 1: Becker's Theory of Language Production

For Becker, language generation is compositional[3] in the way illustrated in figure 1: the phrasal lexicon provides patterns that can provide (at least) some of the expressions needed to convey a message in a certain 'tone'. These phrases are then stitched together, blanks filled in and phrases modified where necessary. If this is not sufficient to generate the utterance, new phrases are generated to smooth over transitions and fill in conceptual holes.

Recover observed linguistic behaviour

Becker's main motivation is to recover the observed linguistic behaviour of the native speaker as a subject for linguistic research. He feels that modern linguists who are working with artificially constructed models - namely the competence ("what language would be like if a decent mathematician had drafted it") and the performance ("everything that people actually say, write or think") models - make the abstract competence language model the subject of their research and deny the existence of language, e.g. English, *as she is spoke* for the purpose of communication between ordinary human beings.

Purely mathematical approach not valid

Becker hits hard against linguists who "dream of performing classic feats like dropping grapefruits off the Leaning Tower of Pisa (...) and in general of having language behave in an orderly way so that they could discover the Universal Laws behind it all". He believes that a purely mathematical approach to language is not only scientifically dishonest - because it disregards nearly all of its subject matter - but will ultimately be self-defeating yielding the scientific initiative to "untutored computer types" trained to "confront large-scale, complex, inelegant, real-world behavioural systems such as language, and attempt to understand the workings of

[3] Becker's notion of 'compositionality' is different from that used in the context of generative linguistics or, in fact, that used in rule-based MT (RBMT), i.e. that a target structure can be *generated* in a compositional manner (combining units the size of words or smaller) following a detailed analysis of the source structure and the establishment of correspondences between grammatical descriptions of the source and the target structures.

the systems without vainly pretending that they can be reduced to pristine-pure mathematic formulations."
According to Becker, dealing with lexical phrases has the additional advantage over transformations "and other such chimeras" in that they are actually observable - because they are real. [4]

TM resources underused

In TM systems, two intellectually challenging problems have to be addressed which cannot just be solved by clever engineering.

Fuzzy matches
One is the decision of how to deal with cases where no exact matches can be found. Developers generally opt to search for similar matches and to calculate a ranking of identified 'fuzzy' matches which are then offered to the translator as a possible base for the translation of the new segment.

Initial TM (alignment)
The other problem occurs when translators want to create *initial translation memories* by aligning previous translations with their source on a segment-by-segment basis in order to import these aligned segments into a TM and then use this for the translation of a new version of the same source. Developers generally offer alignment tools which work in either interactive or fully automatic mode.

Limitations

While the former problem still remains to be solved – in theory and in practice – the latter has been solved to a large extend. Remaining problems are purely engineering problems.
The availability of alignment tools linked with the now wide-spread use of translation memory systems has lead to the creation of massive bi- and multilingual parallel corpora aligned at sentence level[5] – the only segment level currently accessible by TM systems.
However, matching segments at sentence level unnecessarily restricts the potential and the usefulness of translation memories as extremely valuable linguistic resources for the following reason.
Returning to the first problem described as intellectually challenging, i.e. fuzzy matching: if a TM system cannot find an exact match in a TM, it can only propose fuzzy matches, i.e. matches where parts of the old and new source overlap leaving the task of adapting the translation of the old source to the new source up to the translator.

[4] It should be noted that Becker's categories for phrases are different from those generally accepted by English grammarians where each phrase is named after the word class of the head of the phrase, i.e. noun phrase, verb phrase, adjective phrase, adverb phrase, and prepositional phrase. (Greenbaum, 1996:208)
[5] Segments currently used by TM systems can be defined to a certain degree by users of TM systems and can also include text strings defined in documents such as headers or members of lists. Segments, however, can never be defined using linguistic criteria.

This can be a highly complex operation and, in fact, so cumbersome that translators often opt out of the fuzzy match proposal operation by setting the percentage threshold of the fuzzy match component so high that high percentage matches which could contain matching phrases are hidden away from them. Instead, they prefer to translate the new source without the support of the TM system to save time – valuable matches at sub-sentence level are lost.
The probability of finding exact matches at a lower phrasal level (e.g. at NP, VP or PP level) is significantly higher than the probability of finding exact matches at the current sentence level (i.e. the current TM segment level) as experiments have shown.
Storing, matching and proposing segments at phrasal level has a number of advantages, among them:

- Translators will be offered a higher percentage of exact matches from translation memories.
- The use of information stored in translation memories will increase – matching phrases in otherwise fuzzy matching sentences will no longer fall below the match percentage threshold set by most translators.
- The quality of translations produced by translators using translation memories will increase.
- TM systems will be able to translate a larger amounts of source text automatically without the need to adapt fuzzy matches manually.

Translation memories implementing phrasal matching will loose much of their appeal as intelligent but basically simple search and replace engines and become sophisticated example-based machine translation engines which when integrated into a hybrid MT solution will yield significant improvements in translation quality.

TM and PL in EBMT

As outlined earlier, one approach to extending the linguistic coverage of TM systems is the redefinition of a translation unit or segment in a situation where only a fuzzy match can be identified. In such a case, translation units could be defined at phrase level. These phrasal units could then be looked up in a phrasal lexicon and be translated by combining already translated phrases stored in the phrasal lexicon – very much along the lines proposed originally by Becker.
Using a TM containing the two entries:

[1]
[ENG] The bullets move to the new paragraph.
[GER] Die Blickfangpunkte rücken in den neuen Abschnitt.
[2]
[ENG] The title moves to the centre of the slide.
[GER] Der Titel rückt in die Mitte des Dias.

Table 1: TM entries

The following new sentence could not be translated automatically:

The bullets move to the centre of the slide.

Table 2: New source sentence

At most, the system would be capable of identifying one of the two sentences in the TM as a fuzzy match and display its translation which would then have to be adapted by a translator.

Higher match values at a price

If the translation memory, however, could provide translation units at phrase level those units could be looked up in a *phrasal lexicon* and combined so that the system could produce a correct translation of the new sentence.

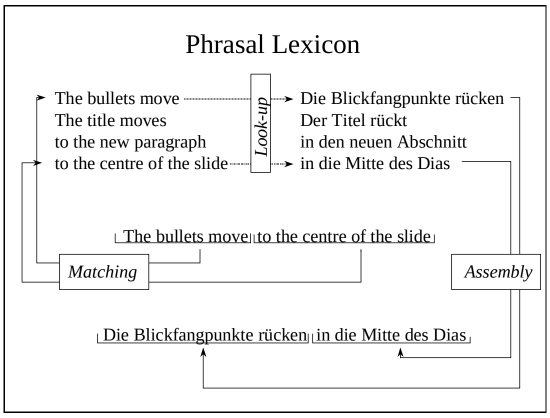


Figure 2: The Phrasal Lexicon - Overview

The phrasal lexicon has the potential to offer translators many of the advantages of a translation memory (consistency, cost savings - no coding or post-editing as with MT systems). However, because it is based on smaller translation units it can potentially identify more exact matches than the sentence based TM systems - but at a price.

Linguistic processing

TM systems do not have to carry out any significant amount of linguistic processing, e.g. they practically do not need to know anything about the target language as most of the processing (matching, calculation of fuzziness, identification of changes etc.) is being done with the source language.

By contrast, a PL would need to know - at a very minimum - enough about the source *and* the target language to identify phrases and describe the linguistic characteristics of their constituent parts. This includes the need to parse source and target language sentences and the ability of the parser used by the system to select one parse for inclusion into the PL while being able to deal with the issues inherent in such a process, including ambiguities, failing parses and wrong parses.

At the same time, the amount of linguistic processing expected to be performed by a PL systems and the human effort in maintaining it will still be significantly lower than in the case of a full MT system.

Therefore, in the context of automated translation tools, PL systems can be positioned in between MT and TM, ideally suited to deal with 'familiar' text (or 'fuzzy matches' in TM terminology) while MT would continue to deal with unfamiliar text (*no match*) and TM with unchanged text (*100% match* at sentence or paragraph level) as illustrated in Figure 3.

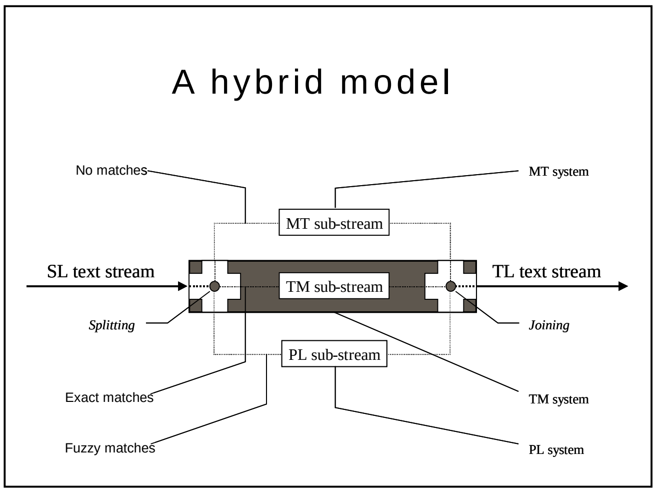


Figure 3: The Phrasal Lexicon (PL) as part of a hybrid MT environment

Frederking and Nirenburg (1994) were probably the first to propose a multi-engine MT architecture, passing the input to multiple translation technologies in parallel and combining their outputs to generate a final translation.

Using 'traditional' MT in combination with the now also established TM approach, together with the new PL engine is one implementation model for this approach.

Phrasal Matching

Different approaches are possible to implement matching at phrase level. The following paragraphs describe our approach taken for the implementation of a demonstrator, the Phrasal Lexicon (PL).

The underlying linguistic technology for this approach was developed by Allan Ramsay whose original parser and grammar for English were extended to cover both English and German with a separate dictionary for each of the languages (Ramsay & Schäler, 1995).

The demonstrator implements a number of important functions:

- The use of one parser and one grammar for English and German to analyse a source and a target text.
- The automatic or interactive/semi-automatic production of a bilingual phrasal lexicon (including linguistic annotation).
- The "translation" of a new source text by matching phrasal units from the PL (by combining of substituting known phrases).

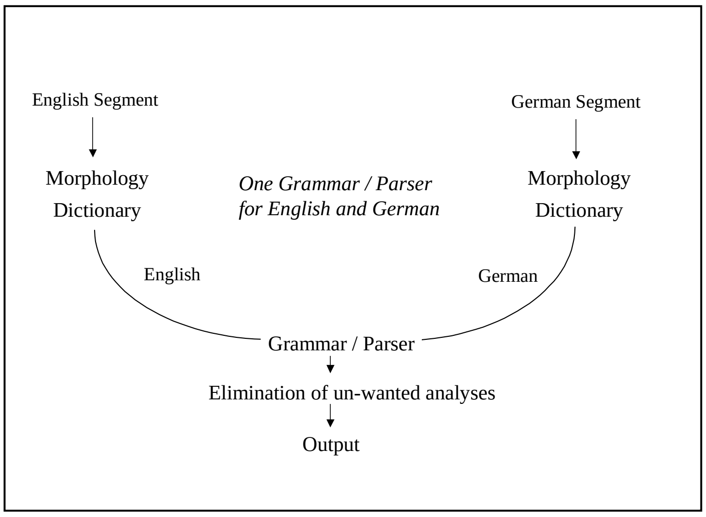


Figure 4: Parsing for the PL

Building a PL

Source language can be provided to the PL in two ways.

- As a file containing bilingual sentence pairs, e.g. exported translation memories.
- Interactively by a translator typing in source and target sentences.

Sentences are read and then analysed by the system's grammar and parser. Source and target language segments are segmented into phrasal units which are stored in a phrasal lexicon together with relevant linguistic information.

As mentioned earlier, only one parser and one grammar are used for the analysis of source and target language segments which ensures comparable output for each of the languages. The only difference is in relation to the dictionary of where there is one for each language.

This system is able to automatically process previously created translation memories without human intervention and transform it into a *linguistically enhanced* translation memory, a phrasal lexicon, thus transgressing the boundaries of traditional translation memory technology.

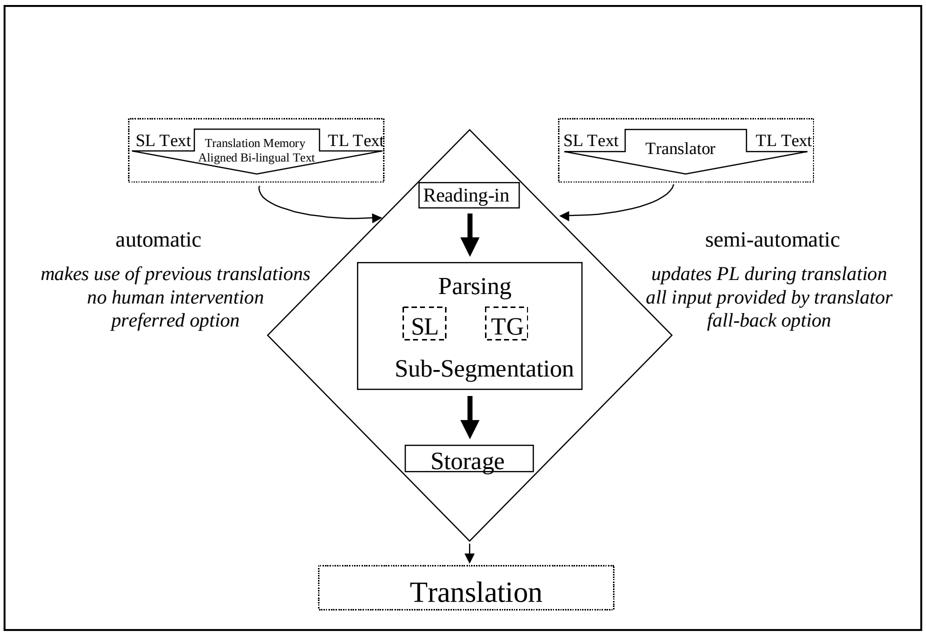


Figure 5: Building a Phrasal Lexicon

Entries in a phrasal lexicon contain multilingual phrase pairs, together with a description of some of their linguistic characteristics.

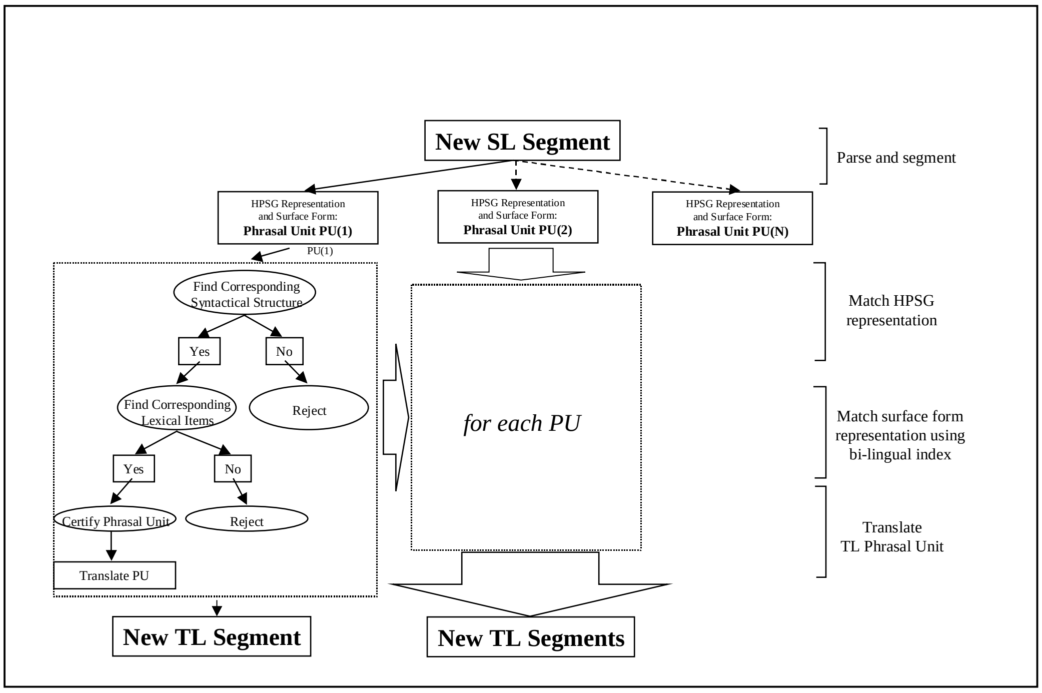
```
phrase_pair
(sign (phonology (structure
        ([he,{buy,ed},a,{car,''}],_,_,_),_,_,
                phon_type(_,_,_,_,english),_,_),
                syntax(nonfoot(major(cat(n(-),v(+)),bar(_,phrasal(+))),
                head(agree(_,_,third(sing(+),plural(-)),_,_),
                vform(vfeatures(finite
                        (tense(present(),past(+)),
                        tensed(+),participle(-),infinitive(),
                        to_form(-)),_,_,_,_,_,_),_),_),_,_),_,_),_),
sign(phonology(structure
        ([er,{kauf,te},ein,{'Auto',''}],_,_,_),_,_,
                phon_type(_,_,_,_,german),_,_),
                syntax(nonfoot(major(cat(n(-),v(+)),bar(_,phrasal(+))),
                head(agree(first(_,plural(-)),second(sing(-),plural(-
                        ),third(sing(+),plural(-)),count(individual(+),kind(-
                        ),mass(-),_),gender(neuter(-),male(+),female(-))),
                vform(vfeatures(finite
                        (tense(present(),past(+)),
                        tensed(+),participle(-),infinitive(-),
                        to_form(-)),_,_,_,_,_,_),_),_),_,_),_,_),_)).
```

Figure 6: PL sample entry

Translating new sentences

In the hybrid system described earlier, only sentences described as *familiar* (fuzzy matches in TM terminology) will be submitted to the PL component. These sentences are then parsed and analysed.

The PL checks if there is a corresponding syntactical structure in the PL. If that is the case the system matches the surface forms of both the source and the target language phrasal units using a bi-lingual index to certify the phrasal units which are then used to generate a new target language segment.



New SL Segment
Parse and segment
HPSG Representation and Surface Form: Phrasal Unit PU(1)
HPSG Representation and Surface Form: Phrasal Unit PU(2)
HPSG Representation and Surface Form: Phrasal Unit PU(N)
PU(1)
Find Corresponding Syntactical Structure
Yes
No
Find Corresponding Lexical Items
Reject
Yes
No
Certify Phrasal Unit
Reject
Translate PU
for each PU
Match HPSG representation
Match surface form representation using bi-lingual index
Translate TL Phrasal Unit
New TL Segment
New TL Segments

Figure 7: PL translation component

Conclusion

For many decades and on various occasions, MT researchers have claimed that the linguistic technology developed by them has made manual, human translation redundant. These claims have so far not had a significant impact on the reality of translation as a profession and as a business.

The one technology that did have an impact on translation has been TM – it changed the way translators work as can be seen when examining the impact it had in the localisation industry, one of the largest employers of technical translators.

Ironically, TM technology worked without any of the sophisticated linguistic technologies developed over decades by MT developers – it is little more than a sophisticated search and replace engine.

However, because of the enormous success of TM systems, there are now large amounts of TMs available – exactly how many can only be estimated: individual products, which are frequently translated into 30 languages and more, can easily contain up to one million words.

But the highly successful approach taken by TM developers is also the cause for the inherent restrictions and limitations of TMs.

To overcome these, we have proposed an implementation of EBMT based on the idea of a phrasal lexicon (or a linguistically enhanced version of a TM system working at phrase level). A first demonstrator of this system has been built and is currently being evaluated.

Acknowledgements

Initial research on this project was conducted at University College Dublin (UCD) under Prof. Allan Ramsay, now at UMIST. It was part-funded by the Irish Industrial Development Agency (IDA).

Bibliographical References

Becker (1975). Joseph D. Becker, Bolt Beranek and Newman. The Phrasal Lexicon. In Proceedings of Theoretical Issues in Natural Language Processing, pages 70-73. Cambridge, Massachusetts (10-13 June 1975).

Frederking and Nirenburg (1994). Three Heads are better than one. R Frederking and S. Nirenburg. In Proceedings of Applied Natural Language Processing (ANLP), 1994.

Gale, William A. & Church, Kenneth W. (1991). A program for aligning sentences in bilingual corpora. In Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics, Berkeley, California, June 1991.

Isabelle, P. et al. (1993). Translation Analysis and Translation Automation. In Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation, Kyoto, Japan, 1993.

Macklovitch, Elliot (2000). Two types of Translation Memory. In Translating and the Computer 22, Proceedings of the ASLIB Conference, London (16-17 November 2000).

Nagao, M. (1984). A framework of a mechanical translation between Japanese and English by analogy principle. In A. Elithorn and R. Banerji, editors, Artificial and Human Intelligence, pages 173-180, North Holland, Amsterdam, 1984.

Planas, E. & Furuse O. (1999). Formalizing Translation Memories. In Proceedings of MT Summit VII, Singapore (13-17 September 1999), pp 331-339.

Ramsay, Allan & Schäler, Reinhard (1995). *Case and Word Order in English and German*, Allan Ramsay and Reinhard Schäler, in: Ruslan Mitkov and Nicolas Nicolov (eds.): *Recent Advances in Natural Language Processing*, John Benjamins (Amsterdam/Philadelphia) 1997.

Sadler (1990). Victor Sadler and Ronald Vendelmans. Pilot Implementation of a Bilingual Knowledge Bank. Proceedings of the Conference on Computational Linguistics (COLING) 1990, pp. 449-451.

Sato, S. & Nagao,M. (1990). Toward Memory-based Translation. In Proceedings of the Conference on Computational Linguistics (COLING) 1990, pp 247-252.

Schäler, Reinhard (1994). A Practical Evaluation of an Integrated Translation Tool during a Large Scale Localisaton Project. In Proceedings of the 4th Conference on Applied Natural Language Processing (ANLP-94), Stuttgart, Germany (October 13-15, 1994).

Schäler, Reinhard (1996). Machine translation, translation memories and the phrasal lexicon: the localisation perspective. In TKE 96, EAMT Machine Translation Workshop, Vienna, Austria (29-39 August 1996), pp. 21-33.

Sumita, E. & Iida, H. (1991). Experiments and Prospects of Example-based Machine Translation. In Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL), 1991, pp 185-192.

EBMT Seen as Case-based Reasoning

Harold Somers

Centre for Computational Linguistics, UMIST,
PO Box 88, Manchester M60 1QD, England
Harold.Somers@umist.ac.uk

Abstract

This paper looks at EBMT from the perspective of the Case-based Reasoning (CBR) paradigm. We attempt to describe the task of machine translation (MT) seen as a potential application of CBR, and attempt to describe MT in standard CBR terms. The aim is to see if other applications of CBR can suggest better ways to approach EBMT.

Introduction

Case-based reasoning (CBR) is a well-established paradigm for problem solving which emerged in the 1980s as an alternative to rule-based expert systems. Instead of rules, CBR represents expertise in the form of past “cases”, and new problems are solved by finding the most similar case in the case-base, and using this as a model for the new solution through a process of “adaptation”.

EBMT is a reasonably well-established paradigm for machine translation which emerged in the 1990s as an alternative to rule-based MT systems. Instead of rules, EBMT represents its linguistic knowledge in the form of “examples” of previous translations, and new translations are made by finding the most similar example in the example-base, and using this as a model for the new translation through a process of “recombination”.

The parallel between CBR and EBMT is so obvious that one would think it perhaps unnecessary to make it. But, despite the earlier establishment of CBR as a problem-solving paradigm, very few papers on EBMT make the connection explicit, and if they do, it is only as a passing comment. With one notable exception, reference to CBR pays only lip service: no attempt is made to take what has been said about CBR to see if it applies to the problem of MT. The major exception, which we should mention very clearly, is the work of Bróna Collins and colleagues (Collins, 1998; Collins & Cunningham, 1995, 1996, 1997; Collins et al., 1996): her work was explicitly in the paradigm of CBR, since it was carried out from within a Computer Science department specialising in CBR. As for the rest of the EBMT literature, the present author has attempted (Somers, 1999) a very thorough survey of articles on EBMT: of about 130 articles collected and read, less than 10% even mentioned CBR or related paradigms, by name.

The purpose of this paper is to look at MT from the perspective of CBR, that is, to consider the CBR approach to problem solving, to see how (or whether) CBR terminology and ideas fit the particular problem of MT, and to see if we can gain any insights from this exercise. The basic assumption of this paper is that EBMT does indeed come within the general paradigm of CBR-based systems. For the purposes of discussion, however, we will use the terms “EBMT” and “CBR” distinctively: the former in its normal meaning, the latter to imply CBR seen as a generic problem-solving method.

CBR: the Paradigm

CBR emerged in the 1980s as an approach to problem solving which offered an alternative to the rule-based approach typical of “expert systems” up until that time. CBR offered a more intuitive approach, based on the way humans appear to solve problems, namely by finding previous similar examples as precedents, and using common-sense reasoning and extrapolation to adapt the precedent to the current problem. This mode of operation is extremely widespread, and can be applied to almost any imaginable human problem. As Leake (1996:3f) states, the CBR approach is based on two tenets about the nature of the world: first, “similar problems have similar solutions”, and second, “future problems are likely to be similar to current problems”. Psychological reality is claimed for CBR as a model of human cognition: “[E]xperts solve problems by applying their experience, whilst only novices attempt to solve problems by applying rules they have recently acquired.” (Watson & Marir, 1994:348)

Riesbeck & Schank (1989) suggest a trade-off between the rule-based and case-based approaches to problem solving: “A rule-based system will be flexible and produce nearly optimal answers, but it will be slow and prone to error. A case-based system will be restricted to variations on known situations and produce approximate answers, but it will be quick and its answers will be grounded in actual experience. In very limited domains, the tradeoffs favor the rule-based reasoner, but the balance changes as domains become more realistically complex.” (p.26)

This methodology applies to a variety of distinct areas of reasoning, and is widely acknowledged as closely modelling human reasoning strategies. In particular, it closely resembles the way human *translators* tend to handle a new text to be translated (Wilss, 1998), which in turn explains the huge popularity among translators of Translation Memory (TM) tools, which are of course a cousin of EBMT (in sharp contrast to the reception that other results of MT research have so far had in that community).

CBR is generally acknowledged to havie its roots in Schank & Abelson’s (1977) work on scripts, along with Medin & Schaffer’s (1978) “Exemplar-based Learning”, Stanfill & Waltz’s (1986) “Memory-based Reasoning” and Carbonell’s (1986) “Derivational Analogy”, while the term itself is probably due to Kolodner & Riesbeck (1986).

CBR is often contrasted with rule-based reasoning, in that rules are replaced by cases. By “case” we mean a "contextualized piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of the reasoner" (Kolodner, 1993:13). In fact, cases can be seen as very specific rules, that is, rules which apply to distinct situations. So CBR is a special kind of rule-based reasoning because the rules have to be interpreted "on the fly", and the same rule may be used differently from one situation to another. Thus far, the same can be said of EBMT.

One of the claimed advantages of CBR is that it overcomes the "knowledge acquisition bottleneck" (Hayes-Roth et al., 1983) of hand-coding a great number of rules, and verifying how they interact with each other. The complexity of real-world domains, according to Riesbeck & Schank (1989:26), makes it "impossible or impractical to specify fully all the rules involved." In CBR, when the existing "rules" don't work (i.e. there is no suitable case in the case-base), one simply adds a new one. Such claims have been made for CBR and related techniques (e.g. Watson & Marir, 1994:348) as well as for EBMT itself (e.g. Sato & Nagao, 1990).

Crucial Elements of CBR

Most texts discussing CBR are agreed on the essential elements that make up a CBR system, namely the database of "cases", of course, and the accompanying design format consisting of the classic "CBR cycle", as illustrated in Figure 1. A new problem must be appropriately **indexed** to enable suitable past cases to be **retrieved**. These must then be **adapted** to the new problem, and the proposed solution **tested**. If the proposed solution is inappropriate, an **explanation** is usually offered, and **repair** must be attempted; once a correct solution is found, this can be added to the memory, fulfilling a **learning** function.

In the following sections we will take each of these elements in turn to see how they relate to EBMT.

Indexing and Representation

Indexing is the term used in the CBR literature to refer to analysis of the new problem in terms of features relevant to finding and comparing cases, the features being referred to as **indexes**. The indexing scheme affects all the other parts of the system, since it reflects and determines the way the cases are represented, that is, the aspects of the cases which are relevant to the problem domain. Kolodner (1993:145) talks in terms of "the lesson(s) it [a case] teaches and the context in which it can teach its lesson(s)": the lessons are the case's **content**, the context its indexes.

Since CBR is a problem-solving methodology, the content of a case is often thought of in terms of a **problem description** coupled with its **solution** and, optionally an **outcome**. Some cases may also include an explicit **explanation** of how the solution relates to the description. Typical examples of CBR are a system which tries to find a suitable menu given some ingredients and diners' preferences and constraints (Hammond, 1986), a medical diagnosis system (Koton, 1988), an agony aunt (Domeshek, 1991). The vocabulary of problem solving permeates the CBR literature: "Cases can be represented in a variety of forms using the full range of AI representational formalisms, including frames, objects, predicates, semantic nets and rules—the frame/object

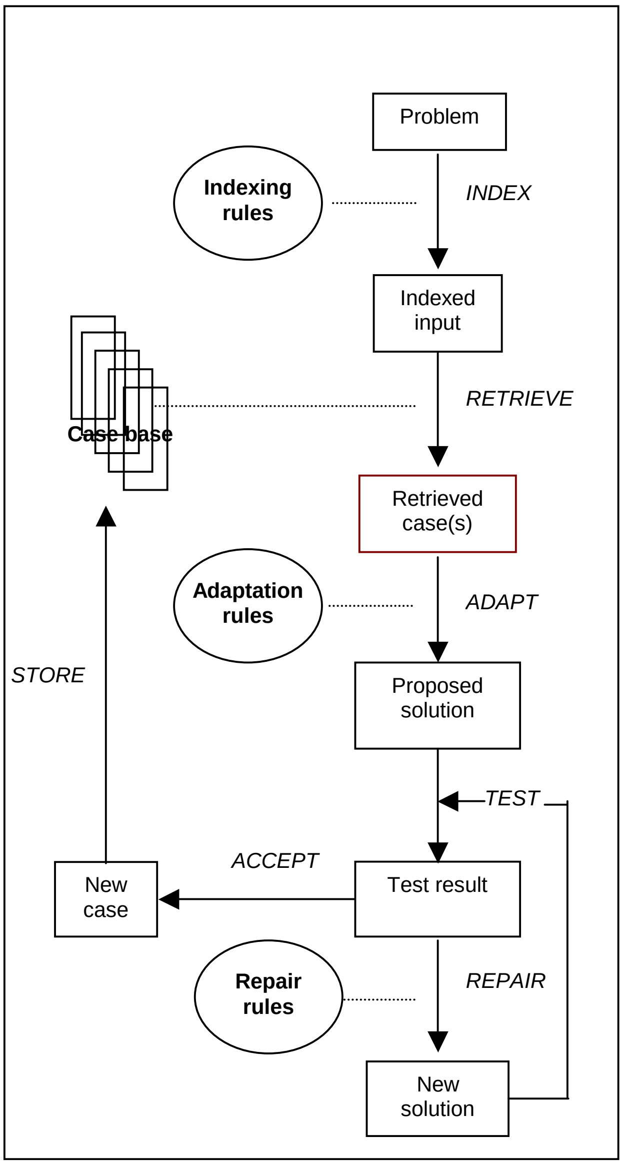


Figure 1. The CBR cycle (based on Riesbeck & Schank, 1989:32)

presentation currently being used by the majority of CBR software." (Watson & Marir, 1994:331)

Figure 2 shows an example.

Here already we see a difference between CBR and EBMT: many CBR systems address "problems" which have "solutions" which involve a sequence of **goals** to achieve, and "outcomes" or changes to the state of the "world" after the case has been invoked. In contrast, in EBMT the examples are of input–output mappings, and the means of getting from one to the other is rarely made explicit, except inasmuch as elements in the input pattern may be linked to corresponding elements in the output pattern. We will see consequences of this difference at almost every stage.

The content of the cases is distinguished in the literature from the indexing scheme used to retrieve cases. "Indexing" refers specifically to those features of the content that will be used for retrieval. As such they should correspond to the features most likely to describe and distinguish different cases. Kolodner (1993:198ff) suggest that indexes should be predictive, abstract (but not too abstract), and above all useful for the later reasoning task.

```
Problem:
   (include tofu)
   (taste hot)
   (style stir-fry)
Solution:
 (ingredients
ingr1 (tofu lb .5)
ingr2 (soy-sauce tablespoon 2)
ingr3 (rice-wine spoon 1)
ingr4 (cornstarch tablespoon .5)
ingr5 (sugar spoon 1)
ingr6 (broccoli lb 1)
ingr7 (r-pepper piece 6))
 (actions
  act1 (chop object (ingr1) size
       (chunk))
  act2 (marinate object (result act1)
  in (& (ingr2)(ingr3)(ingr4) (ingr5))
       time (20))
  act3 (chop object (ingr6) size
       (chunk))
  act4 (stir-fry object (& (result
       act2)(ingr7)) time (1))
  act5 (add object (result act3) to
       (result act4))
  act6 (stir-fry object (result act5)
       time (2)))
 (style stir-fry)
```

Figure 2. Example of a case, cited in Kolodner (1993:172)

As in many other aspects of CBR, we should be guided by the intuitive nature of this approach, and consider as indexes the kinds of features that *humans* might naturally select.

So how can we relate this terminology to EBMT? In the EBMT literature, the nature of the case base is widely discussed (cf. Somers, 1999). The cases (examples) are represented in a variety of formats, such as lexically aligned tagged sentences (e.g. Kitano, 1993), tree structures (e.g. Sato & Nagao, 1990; Al-Adhaileh & Kong, 1999), multi-level lattices (e.g. Planas & Furuse, 1999), and so on. Theoretically the cases could "speak for themselves" and be stored as unanalysed pairs of strings, though no EBMT system is reported to take this extreme step. This *is* the case with Translation Memories, a special case of EBMT which, in CBR terms, has the retrieval and storage functions, but leaves adaptation to the human user.

One of the ironies of EBMT is that the mechanisms used to produce the annotations (in CBR terms, indexes) for the cases, and also to analyse a new case into the appropriate format, are usually the same as, or very similar to, the rules found in the rule-based systems they are supposed to replace.

Many of the earliest EBMT systems were hybrids, using the example-based method for only a part of the process (e.g. Sato & Nagao, 1990; Watanabe, 1992; and several others) or for certain problems (e.g. Sumita et al., 1990). Similarly, in many CBR systems the adaptation stage (see below) is rule-based (Leake, 1996:11), while some systems have a rule-based compoinent as a back-up if not relevant cases are available (e.g. Goel et al., 1994; Koton, 1988).

A feature of recent EBMT research has been the tendency to take similar examples and store them as a single **generalized** example, sometimes so much so that they resemble traditional transfer rules (e.g. Kitano & Higuchi, 1991; Furuse & Iida, 1992; Brown, 1999). Some researchers report procedures for automatically discovering generalized patterns (Cicekli & Güvenir, 1996; Güvenir & Tunç, 1996; Güvenir & Cicekli, 1998; McTait & Trujillo, 1999). The notion of "generalization" is found in the CBR literature, but in a limited way. Riesbeck & Schank (1989:36ff) describe the dynamic formation of "new abstractions … when a number of cases are discovered to share some common set of features", and Branting (1989) describes a system which integrates generalizations into a CBR system. Hammond (1989) similarly suggests that abstract cases can be created where common sets of features are shared. Bergmann & Wilke (1996) explore the idea further. On the other hand, Kolodner (1993:7) suggests that generalization, although possible, is not a significant feature of CBR systems. One senses that general rules are in some way the antithesis of the CBR philosophy.

Representation and Retrieval

The way the cases are represented ("indexed") is of course intimately related to the method of **retrieving** cases which are similar to the given input. This mechanism involves a similarity metric which is used to rank the cases retrieved, together with a search algorithm.

Storage and retrieval is a much-visited problem in CBR. There is a tension between the semantic richness of cases and the efficiency of retrieval, and there is much literature discussing this problem. A striking feature, from the point of view of EBMT, is the small size of the case-base in many of the earliest reported systems: size does not seem to be an issue for CBR, though when it is mentioned at all we see figures in the low hundreds (though cf. Daengdej et al., 1996). In EBMT an example set of under 1,000 would be considered small, and the bigger systems might have as many as three-quarters of a million examples (cf. Somers, 1999:120).

A much used **similarity metric** in CBR is expressed as in (1), where w_i is the importance ("weight") of feature f_i and s is a similarity function for the individual features, I and R indicating input and retrieved cases, respectively.

$$(1)\quad S(I,R)=\frac{\sum_{i=1}^{n} w_i \times s\left(f^I_i, f^R_i\right)}{\sum_{i=1}^{n} w_i}$$

There are obviously three elements to this: identification of the primitive similarity function(s) s, determination of the weights w associated with each feature, and an algorithm for finding the case for which the equation in (1) gives the best value.

The **primitive similarity functions** depend on the choice of features and in particular their complexity. Where features map in an obvious way onto scalar ranges, s can involve simple arithmetic. If the features are more complex then correspondingly more complex functions have to be invoked.

More problematic is the assignment of **weights**. One method is simply to ask human experts to do this (or to weight all features equally). More adventurous systems include a component which learns which features are the most predictive of case differences, or which features are more or less likely to be adapted, and adjusts the relative weights accordingly.

Many CBR systems reportedly use a quite simple search **algorithm** which exhaustively applies the similarity function to all the cases in the case-base. Retrieval time increases linearly with the size of the case-base. One obvious way to overcome this is to organize the search-space so that the higher-weighted features are compared first.

For EBMT, these do not appear to be important issues. Even in early systems where examples are stored as tree structures, little detail is given concerning how tree structures are compared. A similarity metric which makes use of linear distance in a hierarchical thesaurus is widely used for quantifying word similarity (e.g. Nagao, 1984). For the most part, in EBMT the examples have very simple structures, typically sequences of words (this is the case with TMs), or word–tag pairs. The string-edit distance algorithm (Levenshtein, 1966) is widely used, sometimes effectively weighting certain words or categories favourably (e.g. Cranias et al., 1997; Furuse & Iida, 1994; Veale & Way, 1997).

While many CBR and EBMT systems try to retrieve the single best match, or at least to supply a ranking to a set of matches, some systems permit **multiple retrievals**, the idea being that the correct solution will result from taking the best bits of each of them. These might also be described as **partial retrievals**, where the cases are decomposed, making a collection of "substrings" (Nirenburg et al., 1993; Brown, 1997), "fragments" (Somers et al., 1994) or "chunks" (Cunningham et al., 1994; Collins, 1998) of matched material. Figure 3 illustrates this idea.

The idea of using fragments of cases is found in a number of CBR systems, including Redmond (1990), who describes how especially more complex problems can be addressed by looking at subgoals individually, and correspondingly storing cases in "snippets". Marir & Watson (1995) describe a system to estimate building and refurbishment costs, where the complex problems and solutions are all broken down into "subcases": the context information becomes all important in this case, since superficially similar solutions can be quite inappropriate if the underlying situation is different.

```
danger/NN0 of/PRP NN0 < > above/PRP
danger/NN0 of/PRP
of/PRP NN0 < > above/PRP
above/PRP CRD m/NP0
there/PNP is/VVV a/AT0
avalanche/NN0 < > above/PRP
there/PNP is/VVV
is/VVV a/AT0
danger/NN0 of/PRP avalanche/NN0
avalanche/NN0 above/PRP CRD m/NP0
avalanche/NN0 above/PRP
of/PRP avalanche/NN0
there/PNP is/VVV < > a/AT0
is/VVV < > a/AT0
there/PNP is/VVV a/AT0 < > danger/NN0 <
   > of/PRP
there/PNP is/VVV < > danger/NN0 < >
   of/PRP
there/PNP is/VVV a/AT0 < > danger/NN0
a/AT0 < > danger/NN0
there/PNP is/VVV < > danger/NN0
```

Figure 3. Fragments extracted for the input *there is a danger of avalanche above 2000m*.
The individual words are tagged; the matcher can also match tags only, and can skip unmatched words, shown as < >. The fragments are scored for relevance and frequency, which determines the order of presentation. From Somers et al. (1994).

Adaptation

A solution retrieved from a stored case is almost never exactly the same as a new case. CBR systems therefore need one or more strategies for **adapting** the old system to the new situation. This procedure involves two tasks: first deciding *what* aspect of the stored solution needs to be adapted and then *how* to carry out the adaptation. For many, this adaptive aspect is the heart and soul of CBR. Riesbeck & Schank (1989) refer to it as "the ultimate task of a case-based reasoner" (p. 41). It is important because it not only permits the reuse of existing solutions, but it also contributes to the creation of new solutions and hence to the *learning* capability of CBR.

Despite its importance, adaptation is sometimes omitted from CBR systems, or replaced with human intervention. Watson & Marir comment that "it should not be viewed as a weakness of CBR that it encourages human collaboration in decision support" (1994:330). In CLAVIER (Mark, 1989), an early commercial CBR system, it is reported by Mark et al. (1996) that, as the case-base grew through usage, adaptation and maintenance became more difficult, and eventually the designers decided to replace the adaptation component with an interactive module.

The EBMT equivalent of a system which consists essentially of a retrieval mechanism whose output is then passed to a human is a TM system; and this author (Somers, 1999:114) has explicitly tried to distinguish EBMT and TMs on precisely these grounds: what makes EBMT an interesting process is the extent to which the "hard" part is automated! Similarly, CBR can hardly be considered "reasoning" if its performance amounts to copying and pasting.

The CBR literature is in agreement that adaptation is the most taxing aspect of the paradigm. Hanney & Keane (1997) for example refer to the "adaptation knowledge bottleneck" suggesting that it is difficult to derive any knowledge about how adaptation should be conducted from the cases alone. What is needed is some prior domain knowledge which serves to contextualize the cases; and this domain knowledge is necessarily expressed as general *rules*. In this way, hybrid case- and rule-based systems are developed.

We can see this approach in some EBMT systems, where the cases are "generalized", as described above, sometimes to such an extent that they really end up as rules.

Overviews (e.g. Riesbeck & Schank, 1989:43; Kolodner, 1993:395ff; Watson & Marir, 1994:334) list up to a dozen types of adaptation, broadly divided between **structural** and **derivational** adaptation techniques. In the former, rules are applied to (a copy of) the case selected as the best match. In the latter, the algorithms, methods or rules which generated the original solution are reused to generate a new solution. Because of differing terminology, it is not always clear whether differently named methods are really distinct.

The adaptation step in EBMT is usually termed **recombination**, though this term is more specifically applicable in systems where the matching process retrieves multiple, sometimes partial, solutions, a strategy not widely used in CBR.

The simplest of the CBR adaptation methods is null adaptation; then there are substitution methods (reinstantiation, parameter adjustment, abstraction and respecialization, case-based substitution, specialized search), transformation methods (commonsense transformation and model-guided repair) and finally derivational replay.

Null adaptation

The first method in reality involves no adaptation at all. Clearly it is used when the new problem exactly matches an existing case, or it may be used when the new problem is sufficiently close to the matched case that no adaptation is necessary (bearing in mind the existence of a revision stage). In EBMT, null adaptation occurs when an exact match is found, which may be more or less common depending on the application, but for null adaptation to apply when the match is not exact would involve the system "knowing" that the differences between the input and the match were insignificant. One can imagine ways of doing this.

Reinstantiation

In reinstantiation, the old and new problems are structurally similar, but differ in the values of elements. Reinstantiation involves replacing the old values with new. This is a method often found in EBMT: for instance (2) could be used as a model for the translation of (3) by replacing *she* with *he*, *big* with *blue* and *feet* with *eyes* to give (4).

(2) *Kanojo wa ashi ga ōkii.*
SHE topic FOOT subj BIG
She has big feet.

(3) *Kare wa me ga aoi.*
HE topic EYE subj BLUE

(4) He has blue eyes.

In reinstantiation we have to know the correspondences between the elements that we are exchanging, but we also have to be sure that the simple substitution is permitted. In CBR terms, if there are implicit relationships between the slots, reinstantiating one might have repercussions. This can be easily illustrated in the case of EBMT: if we want to use (5) as a model for the translation of (6), we cannot simply replace *man–homme* with *woman–femme* (7), but also change some of other words in the sentence (7).

(5) That old man has died.
Ce vieil homme est mort.

(6) That old woman has died.

(7) a. **Ce vieil femme est mort.*
b. *Cette vieille femme est morte.*

This problem is referred to in the EBMT literature as **boundary friction** (Nirenburg et al., 1993:48; Collins, 1998:22; Somers, 1999:133). One solution to this problem in CBR terms might be to treat it as a case for …

Parameter adjustment

This is a structural adaptation technique in which specific parameters of the retrieved and new cases differ. A key element seems to be the use of "specialized adjustment heuristics" to cope with the problem (Kolodner, 1993:404). A possible interpretation of this in EBMT terms is if in (5) the representation of the French translation included an indication of the agreement requirements, so that the substitution of *man* with *woman* in (6) would trigger specific agreement rules to adapt the other words. MT experts might call such a specialized adjustment heuristic a "transfer rule".

Abstraction and respecialization

This technique, also termed "local search", is a type of substitution that allows a novel solution to be generated from an example which differs only in a small part. The idea is to take the piece of the solution that does not fit, look for abstractions of that piece, and then try other specializations of the abstraction in the current situation. This technique obviously depends on there being a hierarchically structured knowledge base behind the case base, and is very well illustrated for EBMT by Sumita & Iida's (1991) system which translates Japanese adnominal particle constructions (*A no B*) with the help of a thesaurus.

Case-based substitution

This adaptation technique comes into play when parts of a solution have to be found in additional cases.

Papaioannou (2000) adopted this as a solution to the boundary friction problem in a simulated English–Greek EBMT system. Because Greek is a highly inflected language, there is danger of recombining inappropriately inflected fragments. The examples in Papaioannou's system are tagged and lemmatised to show appropriate morphological information, as in Figure 4.

I saw the new prime-minister.
Είδα τον νέο πρωθυπουργό.

```
<s>
 <gw cat="V" attrs="Act Pst Ind Sng 1st"
  lemma="blepw/see">Eida</gw>
 <gw cat="Art" attrs="Msc Sng Acc"
  lemma="o/the">ton</gw>
 <gw cat="Adj" attrs="Msc Sng Acc"
  lemma="neos/new">neo</gw>
 <gw cat="N" attrs="Msc Sng Acc"
  lemma="prw8upourgos/prime-
  minister">prw8upourgo</gw>
 <punc>.</punc>
</s>
```

Figure 4. An example from Papaioannou (2000).

The recombination stage "knows" that certain attributes have to match up (agreement of Art, Adj and N, for instance), so when the system retrieves examples for a new input, it notes the particular details of any discrepancies and specifically searches the rest of the example-base for the missing item(s). For instance, the sentence in (8) matches the example in Figure 4 perfectly, except for the last lemma and surface word. In order to adapt the example, the system searches the example-base for another case that contains exactly the configuration in (9), and, if it is found can generate the appropriate form *πρόεδρο*.

(8) I saw the new president.

(9) `<gw cat=″N″ attrs=″Msc Sng Acc″ lemma= ″???/president″>??? </gw>`

If we now give the input (10), there are several mismatches. Two of the words have the wrong attributes, and the third word also has the wrong lemma. So the system has to search for the three items in (11).

(10) I saw the new delegates.

(11) a. `<gw cat=″Art″ attrs= ″Msc Plr Acc″ lemma= ″o/the″>??? </gw>`
b. `<gw cat=″Adj″ attrs=″Msc Plr Acc″ lemma= ″neos/new″>??? </gw>`
c. `<gw cat=″N″ attrs=″Msc Plr Acc″ lemma= ″???/ delegate″>??? </gw>`

Supposing there is no "evidence" in the case-base for one of these new combinations. If the missing case is (11), where we do not know the lemma, there is not much we can do. In the case of (11), we may be able to generate the appropriate form of the adjective by looking at other masculine plural adjectives, and comparing the lemma and the surface form, though this would be a further complexity for the adaptation phase. This might be termed "specialized search" (Kolodner, 1993:411).

Common-sense transformation

Kolodner (1993) describes two types of adaptation involving "transformations". Transformation in general involves making deletions or insertions either to the solution as a whole or to some part of it. The first of these is common-sense transformation, which makes use of "a small set of heuristics that use knowledge about the relative importance of different components of an item to determine whether deletions or substitutions should be done" (p. 420f). To be able to do this, the system must of course identify the component needing to be changed, but the representations need to indicate which components are susceptible to this kind of manipulation. In particular, the internal relationships between the components must be maintained after the transformation has taken place.

How might this be implemented in an EBMT system? The idea of deleting or inserting components is widespread in EBMT systems, and is very intuitive. If we have the Malay–English examples in (12), it is not difficult to construct the correct translations of sentences like those in (13).

(12) a. *Dia nak pĕrgi kĕ kĕdai bĕli roti.*
She is going to go to the shops to buy bread.
b. *Dia pĕrgi kĕ pasar nak bĕli baju.*
She went to the market to buy a shirt.
c. *Mĕreka pĕrgi kĕ kampung nak bĕli kereta.*
They went to the village to buy a car.

(13) a. She went to the village to buy bread.
b. They are going to the market.

In fact as humans we bring to bear a certain amount of generic (common-sense) knowledge about how languages work to do this. The work – mentioned above – to extract patterns fully automatically (Cicekli & Güvenir, 1996; Güvenir & Tunç, 1996; Güvenir & Cicekli, 1998; McTait & Trujillo, 1999) needs minimal pairs and would not be able to extract as much as humans can from the examples in (12).

This kind of activity does have its limitations though. Kolodner (1993) notes that internal relationships between elements must be maintained, and here we meet again the "boundary friction" problem already illustrated in (5)–(7). A further problem is that language is not always logical: you might guess from the examples in (14)

(14) *stryd fawr* – big street
stryd fach – small street
tŷ mawr – big house
??? – small house

that the Welsh for 'small house' is *tŷ mach* … unfortunately though you would be wrong (it should be *bach*)!

Model-guided repair

For these reasons, model-guided repair might be a better way of implementing adaptation. As its name suggests, the transformations are guided by some knowledge of the domain, rather than just common sense. In EBMT terms, this would mean verifying that the proposed adaptation is legitimate by submitting it to some kind of analysis. As we will see below, most CBR systems involve an evaluation step where this kind of verification is done, but it could be regarded as part of the adaptation procedure. An obvious way to verify a proposed translation is to try to parse it, though this step would require a grammar of the target language which, perhaps, would undermine the reason for adopting the example-based approach in the first place.

Somers et al. (1994) proposed "disalignment": the proposed output was checked by rerunning the matching algorithm that had been used on the input, this time comparing the proposed output against the target side of the examples. The ease with which the proposed output could be matched was felt to be an indication of its well-formedness.

Derivational replay

Kolodner (1993) describes "derivational replay" as follows:

Each of the adaptation methods described up to now fixes an old solution to fit a new solution. Sometimes, however, it is more appropriate to recomputed a new solution or partial solution using the same means by which the old solution was computed. (p. 435)

A problem with seeing how this technique could apply to EBMT is that there is an underlying assumption that solutions are "computed", whereas in EBMT the solutions are usually just "given". Where solutions (i.e. translations) *are* computed, this is usually in the traditional rule-based manner. So we could regard the hybrid systems that have been described in this light. In the early days of EBMT it was sometimes suggested that this method could be used just for special problem cases(cf. Sumita et al., 1990).

Adaptation-guided retrieval

An implicit assumption in many CBR systems is that the most similar case in the casebase is also the most adaptable. This is not always the case, as Smyth & Keane

(1993, 1995), Leake (1995), Rousen & Aarts (1996) and others have noted. The notion of “adaptation-guided retrieval” has been developed whereby case retrieval is based not only their similarity with the given input, but also the extent to which they represent a good model for the desired output, i.e. to which they can be adapted, that determines whether they are chosen. Collins (1998:31) gives the example of a robot using a restaurant script to get food at Macdonald’s, when buying a stamp at the post-office might actually be a more appropriate, i.e. adaptable, model. Collins et al.’s EBMT system, ReVerb, stores the examples together with a functional annotation, cross-linked to indicate both lexical and functional equivalence. This means that example-retrieval can be scored on two counts: (a) the closeness of the match between the input text and the example, and (b) the adaptability of the example, on the basis of the relationship between the representations of the example and its translation. Obviously, good scores on both (a) and (b) give the best combination of retrievability and adaptability, but we might also find examples which are easy to retrieve but difficult to adapt (and are therefore bad examples), or the converse, in which case the good adaptability should compensate for the high retrieval cost.

Evaluation and Repair

Once a solution has been proposed by the case-based reasoner, the system should then, according to many commentators, **evaluate** the proposal and, if necessary, **revise** (or repair) it. An important part of this is explanation, whereby the reasoner gives the reasons for the failure of the solution. An obvious and important element here is that the system should apparently be aware that it has failed to provide a correct solution – the “test” phase shown in Figure 1 above.

The language in which these ideas are expressed, for example in Schank & Riesbeck (1989) once again reflects the problem-solving scenario that so strongly influences researchers’ ideas in CBR. They suggest that two sorts of failures are possible: “Goals specified in the input are simply not achieved … [or] implicit goals, not specified in the input, are violated” (p. 52). Nevertheless, it quickly emerges from study of the literature that any “evaluation” of the system’s output will be done by human intervention. Any alternative introduces a familiar vicious circle: if the system knew that it had got the wrong answer, it wouldn’t have got the wrong answer.

On the other hand, one of the advantages of empirical approaches (as opposed to rule-based approaches) is the role that stochastic processing can play. Empirical methods can deliver a “certainty rating” along with any result, inasmuch as the result is based on probabilities. Thus, it is claimed, an EBMT system for example not only tells you what it thinks the translation is, but how sure it is of that answer, what its second and third choice solutions might be, and how far behind they are in the ranking.

One area that is important in the CBR community, but which seems to be lacking in EBMT, is **explanation**. Originally thought of as a step in the CBR cycle (see Figure 1 again) to facilitate repair of faulty solutions prior to learning, it has developed into a major field of its own (cf. Schank et al., 1994) related to AI tasks such as story understanding.

All in all, however, it seems that for EBMT, with its shallow representation of cases, and typically uncomplicated “reasoning”, there is not much on offer here.

Learning for Reuse

The final step in CBR is to store accepted solutions in the case-base for later use. In the same way, an approved translation can be added to the example-base. This is normal practise in the case of TMs, where the whole process is supervised by a human user. Descriptions of EBMT systems do not as a rule mention this possibility, nor is it ruled out.

The main issue here is the need for human quality control, though, conversely, some TM systems allow examples to be stored with an annotation identifying whether their source is human translation or MT (including EBMT).

Kolodner (1993:566) briefly discusses some of the issues in case-based learning. She notes that new cases can be added by the system itself and, of course manually. She also suggests that storing failures, i.e. cases where the system could not come up with the correct solution, can be beneficial: a system can learn from its mistakes as well as from its successes, especially if the failed case is annotated to show where it went wrong. There may be something in this for EBMT.

A second issue is the effect on the case-base of storing multiple similar or even identical examples. As Kolodner (1993) states, “Some cases added to a case library might add to the library without adding to the capabilities of a system. Such cases might or might not affect the efficiency of retrieval, depending on the retrieval algorithm.” (p. 567)

A similar point can be made with regard to EBMT (see Somers, 1999:121). If we construct the example-base froma corpus, sthere will be repetition and overlap. Some examples will mutually reinforce each other, being identical, or exemplifying the same translation phenomenon. But other examples will be in conflict: the same or similar phrase in one language may have two different translations for no other reason than inconsistency (e.g. Carl & Hansen, 1999:619).

Where examples reinforce each other this may or may not be useful. Some systems (e.g. Somers et al., 1994; Öz & Cicekli, 1998; Murata et al., 1999) involve a similarity metric which is sensitive to frequency. But if no such weighting is used, multiple similar or identical examples are just extra baggage, introducing a kind of spurious ambiguity because there seem to be multiple alternative solutions to the same problem.

Conclusions

The aim of this essay was to see if the similarity between EBMT and CBR was anything more than skin-deep. My hope was to find some inspiration in the CBR literature to push EBMT forward.

I think we have shown that in many respects EBMT can be described using CBR terminology, but for the most part we do not learn anything from this. The main problem seems to be the discrepancy in the complexity of representation of examples in CBR compared to EBMT. If anything, the CBR community might find EBMT an interesting special case, showing what one can do with very superficial analysis and representation of the cases.

I think the most fruitful area for further consideration is the question of adaptation, especially the way it might interact with retrieval, as the Dublin group (Collins and colleagues) have already shown us.

References

Al-Adhaileh, M.H. and E.K. Tang. 1999. Example-based machine translation based on the synchronous SSTC annotation schema. *Machine Translation Summit VII*, Singapore, pp. 244–249.

Bergmann, R. and W. Wilke. 1996. On the role of abstraction in case-based reasoning. In *EWCBR-96*, pp. 28–43.

Branting, L.K. 1989. Integrating generalizations with exemplar-based reasoning. *Case-Based Reasoning: Proceedings of a Workshop on Case-Based Reasoning*, Pensacola Beach, Florida, pp. 224–229.

Brown, R.D. 1997. Automated dictionary extraction for "knowledge-free" example-based translation. *Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation*, Santa Fe, New Mexico, pp. 111–118.

Brown, R.D. 1999. Adding linguistic knowledge to a lexical example-based translation system. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, Chester, England, pp. 22–32.

Carbonell, J.G. 1986. Derivational analogy: A theory of reconstructive problem solving and expertise acquisition. In R. Michalski, J. Carbonell and T. Mitchell (eds) *Machine Learning: An Artificial Intelligence Approach*, Vol. 2, pp. 371–392.

Carl, M. and S. Hansen. 1999. Linking translation memories with example-based machine translation. *Machine Translation Summit VII*, Singapore, pp. 617–624.

Cicekli, I., and H.A. Güvenir. 1996. Learning translation rules from a bilingual corpus. *NeMLaP-2: Proceedings of the Second International Conference on New Methods in Language Processing*, Ankara, Turkey, pp. 90–97.

Collins, B. 1998. *Example-based Machine Translation: an adaptation-guided retrieval approach*. PhD thesis, Trinity College, Dublin.

Collins, B. and P. Cunningham. 1995. A methodology for example-based machine translation. *CSNLP 1995: 4th Conference on the Cognitive Science of Natural Language Processing*, Dublin.

Collins, B. and P. Cunningham. 1996. Adaptation-guided retrieval in EBMT: A case-based approach to machine translation. In *EWCBR-96*, pp. 91–104.

Collins, B. and P. Cunningham. 1997. Adaptation guided retrieval: Approaching EBMT with caution. *Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation*, Santa Fe, New Mexico, pp. 119–126.

Collins, B., P. Cunningham and T. Veale. 1996. An example-based approach to machine translation. *Expanding MT Horizons: Proceedings of the Second Conference of the Association for Machine Translation in the Americas*, Montreal, Quebec, pp. 1–13.

Cranias, L., H. Papageorgiou and S. Piperidis. 1997. Example retrieval from a translation memory. *Natural Language Engineering* **3**, 255–277.

Cunningham, P., B. Smyth and T. Veale. 1994. On the limitations of memory based reasoning. In *EWCBR-94*, pp. 75–86.

Daengdej, J., D. Lukose, E. Tsui, P. Beinat and L. Prophet. 1996. Dynamically creating indices for two million cases: a real world problem. In *EWCBR-96*, pp. 195–119.

Domeshek, E.A. 1991. What Abby cares about. *Case-Based Reasoning: Proceedings of a Workshop…,* Washington, D.C., pp. 13–24.

EWCBR-93 = S. Wess, K.-D. Althoff and M..M. Richter (eds), *Topics in Case-Based Reasoning: First European Workshop, EWCBR-93, Kaiserslautern, Germany…*(Lecture Notes in Computer Science 837), Berlin: Springer

EWCBR-94 = J.-P. Haton, M. Keane and M. Manago (eds) *Advances in Case-Based Reasoning: Second European Workshop, EWCBR-94, Chantilly, France…* (Lecture Notes in Computer Science 984), Berlin: Springer.

EWCBR-96 = I. Smith and B. Faltings (eds), *Advances in Case-Based Reasoning: Third European Workshop, EWCBR-96, Lausanne, Switzerland…* (Lecture Notes in Computer Science 1168), Berlin: Springer.

Furuse, O. and H. Iida. 1992. An example-based method for transfer-driven machine translation. *Quatrième colloque international sur les aspects théoriques et méthodologiques de la traduction automatique; Fourth International Conference on Theoretical and Methodological Issues in Machine Translation. TMI-92*, Montréal, Québec, pp. 139–150.

Furuse, O. and H. Iida. 1994. Constituent boundary parsing for example-based machine translation. *COLING 94, The 15th International Conference on Computational Linguistics*, Kyoto, Japan, pp. 105–111.

Goel, A., K. Ali, M. Donnellan, A. Garza and T. Callantine. 1994. Multistrategy adaptive navigational path planning. *IEEE Expert*, **9**.6, 57–65.

Güvenir, H.A. and I. Cicekli. 1998. Learning translation templates from examples. *Information Systems* **23**, 353–363.

Hammond, K.J. 1986. CHEF: a model of case-based planning. *Proceedings AAAI-86 Fifth National Conference on Artificial Intelligence*, Philadelphia, Pa., pp. 267–271.

Hammond, K.J. 1989. On functionally motivated vocabularies: an apologia. *Proceedings of the International Joint Conference on Artificial Intelligence*, Milan, Italy.

Hanney, K. and M.T. Keane. 1997. The adaptation knowledge bottleneck: How to ease it by learning from cases. In *ICCBR-97*, pp. 359–370.

Hayes-Roth, F., D. Waterman and D. Lenat (eds) 1983. *Building Expert Systems*. Reading, Mass.: Addison Wesley.

ICCBR-95 = M. Veloso and A. Aamodt (eds), *Case-Based Reasoning Research and Development: First International Conference on Case-Based Reasoning, ICCBR-95, Sesimbra, Portugal…* (Lecture Notes in Computer Science 1010), Berlin: Springer

ICCBR-97 = D.B. Leake and E. Plaza (eds), *Case-Based Reasoning Research and Development: Second*

International Conference on Case-Based Reasoning, ICCBR-97, Providence, RI ... (Lecture Notes in Computer Science 1266), Berlin: Springer.

Kitano, H. 1993. A comprehensive and practical model of memory-based machine translation. *Proceedings of the International Joint Conference on Artificial Intelligence*, Chambéry, France, pp. 1276–1282.

Kitano, H. and T. Higuchi. 1991. High performance memory-based translation on IXM2 massively parallel associative memory processor. *AAAI-91 Proceedings Ninth National Conference on Artificial Intelligence*, Menlo Park: AAAI Press/The MIT Press, pp. 149–154.

Kolodner, J. 1993. *Case-Based Reasoning.* San Mateo, CA: Morgan Kaufmann.

Kolodner, J.L. and D.B. Leake. 1996. A tutorial introduction to case-based reasoning. In Leake (1996), pp. 31–65.

Kolodner, J.L. and C.K. Riesbeck. 1986. *Experience, Memory and Reasoning.* Hillsdale, New Jersey: Laurence Erlbaum.

Koton, P. 1988. Reasoning about evidence in causal explanation. *AAAI 88 The Seventh National Conference on Artificial Intelligence*, St Paul, Minnesota.

Leake, D. 1995. Adaptive similarity assessment for case-based explanation. *International Journal of Expert Systems* **8**, 165–194.

Leake, D.B. (ed.) 1996. *Case-based Reasoning – Experiences, Lessons & Future Directions*. Menlo Park, California: AAAI Press / MIT Press.

Levenshtein, V.I. (1966) Binary codes capable of correcting deletions, insertions, and reversals. *Cybernetics and Control Theory*, **10**, 707–710.

Marir, F. and I. Watson. 1995. Representing and indexing building refurbishment cases for multiple retrieval of adaptable pieces of cases. In *ICCBR-95*, pp. 55–66.

Mark, W.S. 1989. Case-based reasoning for autoclave management. *Case-Based Reasoning: Proceedings of a Workshop on Case-Based Reasoning*, Pensacola Beach, Florida, pp. 176–180.

Mark, W.S., E. Simoudis and D. Hinkle. 1996. Case-based reasoning: expectations and results. In Leake (1996), pp. 267–294.

McTait, K. and A. Trujillo. 1999. A language-neutral sparse-data algorithm for extracting translation patterns. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, Chester, England, pp. 98–108.

Medin, D.L. and M.M. Schaffer. 1978. Context theory of classification learning. *Psychological Review* **85**, 207–238.

Murata, M., Q. Ma, K. Uchimoto and H. Isahara. 1999. An example-based approach to Japanese-to-English translation of tense, aspect, and modality. *Proceedings of the 8th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 99)*, Chester, England, pp. 66–76.

Nirenburg, S., C. Domashnev and D.J. Grannes. 1993. Two approaches to matching in example-based machine translation. *Proceedings of the Fifth International Conference on Theoretical and Methodological Issues in Machine Translation TMI '93: MT in the Next Generation*, Kyoto, Japan, pp. 47–57.

Öz, Z. and I. Cicekli. 1998. Ordering translation templates by assigning confidence factors. In D. Farwell, L. Gerber and E. Hovy (eds) *Machine Translation and the Information Soup: Third Conference of the Association for Machine Translation in the Americas, AMTA'98, Langhorne, Pa*... (Lecture Notes in Artificial Intelligence 1529), Berlin: Springer, pp. 51–61.

Papaioannou, V. 2000. *Recombination in Example-based MT: A Case-Based Reasoning approach*. MSc dissertation, Department of Language Engineering, UMIST, Manchester.

Planas, E. and O. Furuse. 1999. Formalizing translation memories. *Machine Translation Summit VII*, Singapore, pp. 331–339.

Redmond, M. 1990. Distributed cases for case-based reasoning; facilitating use of multiple cases. *AAAI-90 Proceedings Eighth National Conference on Artificial* Intelligence, Menlo Park: AAAI Press / The MIT Press, pp. 304–309

Riesbeck, C.K. and R.C. Schank. 1989. *Inside Case-based Reasoning*. Hillsdale, New Jersey: Lawrence Erlbaum.

Rousen, J. and R.J. Aarts. 1996. Adaptation cost as a criterion for solution evaluation. In *EWCBR-96*, pp. 354–361.

Sato, S. and M. Nagao. 1990. Toward memory-based translation. *COLING-90, Papers Presented to the 13th International Conference on Computational Linguistics*, Helsinki, Finland, Vol. 3, pp. 247–252.

Schank, R.C. and R.P. Abelson. 1977. *Scripts, Plans, Goals and Understanding.* Hillsdale, New Jersey: Lawrence Erlbaum.

Schank, R.C., A. Kass and C.K. Riesbeck. 1994. *Inside Case-Based Explanation.* Hillsdale, New Jersey: Lawrence Erlbaum.

Skousen, R. 1989. *Analogical Modeling*. Dordrecht: Kluwer.

Smyth, B. and M.T. Keane. 1993. Retrieving adaptable cases. In *EWCBR-93*, pp. 209–220.

Smyth, B. and M.T. Keane. 1995. Experiments on adaptation-guided retrieval in case-based design. In *ICCBR-95*, pp. 313–324.

Somers, H. 1999. Review article: Example-based Machine Translation. *Machine Translation* **14**, 113–157.

Somers, H., I. McLean and D. Jones. 1994. Experiments in multilingual example-based generation. *CSNLP 1994: 3rd Conference on the Cognitive Science of Natural Language Processing*, Dublin, Ireland.

Stanfill, C. and D. Waltz. 1986. Toward memory-based reasoning, *Communications of the ACM* **29**, 185–192.

Sumita, E. and H. Iida. 1991. Experiments and prospects of example-based machine translation. *29th Annual Meeting of the Association for Computational Linguistics*, Berkeley, California, pp. 185–192.

Sumita, E., H. Iida and H. Kohyama. 1990. Translating with examples: a new approach to machine translation. *The Third International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Language*, Austin, Texas, pp. 203–212.

Veale, T. and A. Way. 1997. *Gaijin*: A bootstrapping approach to example-based machine translation. *International Conference, Recent Advances in Natural Language Processing*, Tzigov Chark, Bulgaria, pp. 239–244.

Watanabe, H. 1992. A similarity-driven transfer system. *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, Actes du quinzième colloque international en linguistique* *informatique, COLING-92*, Nantes, France, pp. 770–776.

Watson, I.D. (ed.) 1995. *Progress in Case-based Reasoning: First United Kingdom Workshop, Salford, UK*... (Lecture Notes in Computer Science 1020), Berlin: Springer Verlag.

Watson, I. and F. Marir. 1994. Case-based reasoning: A review. *The Knowledge Engineering Review* **9**, 327–354.

Wilss, W. 1998. Decision making in translation. In M. Baker (ed.) *Routledge Encyclopedia of Translation Studies*, London: Routledge, 57–60.

Translating with Examples

Andy Way,
School of Computer Applications,
Dublin City University,
Dublin, Ireland.

Email: away@compapp.dcu.ie

Abstract

Machine Translation (MT) systems based on Data-Oriented Parsing (DOP: Bod, 1998) and LFG-DOP (Bod & Kaplan, 1998) may be viewed as instances of Example-Based MT (EBMT). In both approaches, new translations are processed with respect to previously seen translations residing in the system's database. We describe the DOT models of translation (Poutsma 1998; 2000) based on DOP. We demonstrate that DOT1 is not guaranteed to produce the correct translation, despite provably deriving the most probable translation. The DOT2 translation model solves most of the problems of DOT1, but suffers from limited compositionality when confronted with certain data. Notwithstanding the success of DOT2, any system based purely on trees will ultimately be found wanting as a general solution to the wide diversity of translation problems, as certain linguistic phenomena require a description at levels deeper than surface syntax. We then show how LFG-DOP can be extended to serve as a novel hybrid model for MT, LFG-DOT (Way, 2001), which promises to improve upon DOT and other EBMT systems.

1 Problems for EBMT

One of the major advantages which is often claimed of EBMT is that the overall quality of translation increases incrementally as the set of stored translations increases. For example, Mima *et al.* (1998) report that in their EBMT system, translation quality rose in an almost linear fashion, from 30% with 100 examples to 65% with all 774 examples. They also note that there seems to be a limit beyond which adding further examples does not improve the overall translation quality.

While the chances of finding an exact match become greater as the corpus size increases, there are two knock-on effects whose impact on the EBMT system should be minimized. Firstly, adding more examples has a computational cost, especially if the examples need to be parsed: some EBMT systems (e.g. Sato & Nagao, 1990; Watanabe, 1994) store examples as annotated trees, for instance. Whether this is the case or not, adding more examples causes more storage problems, and adds to the complexity of the search and retrieval stages of the EBMT process. It is, therefore, unclear that one of the purported advantages of EBMT, namely a lessening in computational cost, is indeed a real benefit. Secondly, adding more examples may not be useful in practice. For example, a newly added translation pair may be identical to, or overlap other examples. Where a system involves the computation of a 'similarity metric' (e.g. Somers *et al.*, 1994), this may be influenced by the frequency of examples, so that the score attached to certain matches increases if a large number of similar examples are found. Alternatively, in other systems, identical or similar examples may just be redundant. Somers (1999:121) observes that "in such systems, the examples can be seen as surrogate 'rules', so that, just as in a traditional rule-based MT system, having multiple examples (rules) covering the same phenomenon leads

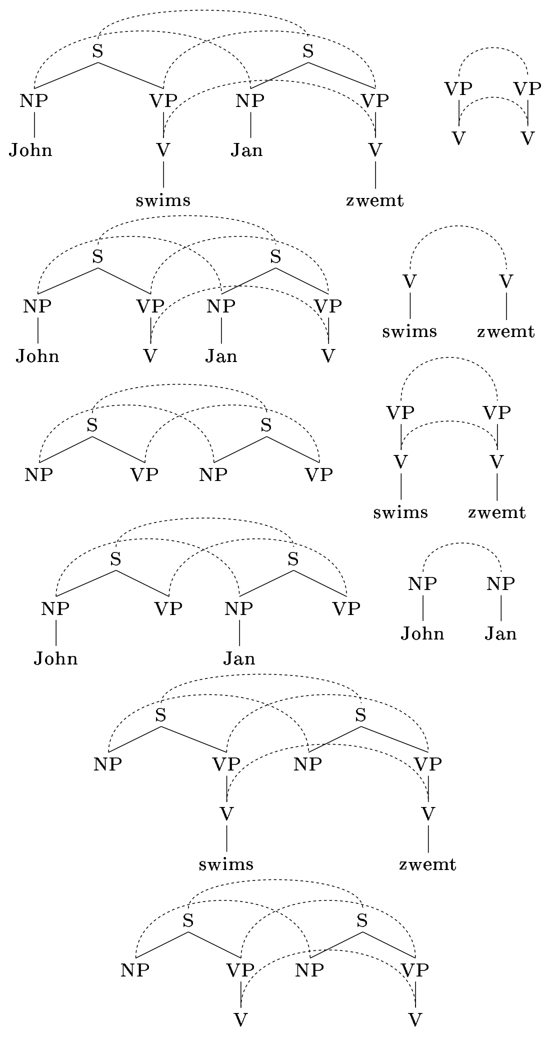


Figure 1: The complete DOT treebank for the linked sentence pair ⟨*John swims*, *Jan zwemt*⟩

to over-generation".

The two main problems for EBMT are *boundary definition* and *boundary friction*. The first of these describes the scenario where retrieved fragments may not be well-formed constituents. This is a particular problem for pure EBMT systems, where syntactic well-formedness needs to be ensured without grammatical information actually being employed. The second, boundary friction, is a problem in the retrieval process, in that context may not be taken into account. This may be illustrated by attempting to translate *I have a big dog* into German. An EBMT system might retrieve the close matches in (1):

(1) a. A big dog eats a lot of meat ⟶Ein großer Hund frisst viel Fleisch.

b. I have two ears ⟶Ich habe zwei Ohren.

From these examples, the useful translation fragments in (2) might be isolated by our EBMT system:

(2) a. A big dog ⟶Ein großer Hund

b. I have ⟶Ich habe

In this case, these fragments would be combined to give the translation *Ich habe ein großer Hund*, which is ungrammatical as we have an NP bearing nominative case in object position.

We shall show how DOT and LFG-DOT fare with these two main problems for EBMT. With strict notions of decomposition of fragments, the problem of boundary definition is unproblematic. DOT, however, may suffer from the problem of boundary friction, while given the additional syntactic information available in the Lexical-Functional Grammar (LFG: Kaplan & Bresnan, 1982) f-structures, this problem is considerably reduced in LFG-DOT.

2 Data-Oriented Translation

Poutsma (1998; 2000) has developed a model of translation based on Tree-DOP—Data-Oriented Translation (DOT). There are two different versions of DOT. We shall describe these briefly.

2.1 DOT1

Poutsma's DOT1 model (1998; 2000) is based on the methodology of Tree-DOP (cf. Bod, 1998), and relates POS-fragments between two languages (English and Dutch here), with an accompanying probability. DOT1 is parameterized on similar lines to Tree-DOP. Its **representations** are *linked* phrase-structure trees. Figure 1 shows the complete treebank for the linked sentence pair ⟨*John swims*, *Jan zwemt*⟩.[1]

[1] Here, and in future examples, we 'translate' proper names purely in order to differentiate completely source and target representations and strings.

These trees are augmented to incorporate semantic information, as a DOT1 treebank links semantically equivalent trees. It can be seen that the top left tree pair in Figure 1 represents the full parse trees for this translation pair. This tree pair is subjected to the DOT1 **decomposition** process (based on the DOP *Root* and *Frontier* fragmentation operations—essentially *Root* allows new tree fragments to be built by selecting a node to be the root node of a new tree, and deleting all other nodes except this new root and all nodes dominated by it, while *Frontier* selects a (possibly empty) set of nodes in the newly created subtree, excluding the root, and deletes all subtrees dominated by these selected nodes) so that the other tree pairs in Figure 1 are derived. Note also here that all S-rooted linked structures are derived via *Frontier*, while all others are produced via *Root*. It is these operations which delimit the boundary definitions in the DOT translation models.

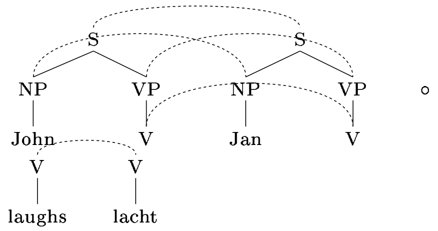
Tree pair fragments are subjected to the **recombination** process in DOT using the same composition operation, namely leftmost substitution (to ensure the uniqueness of each derivation), as Tree-DOP. The target derivations are then assembled by replacing all subtrees of the source derivation by their linked target subtrees. Any pair of fragments which can legitimately combine with others results in a well formed derivation given the corpus. This causes a particular problem of boundary friction in DOT (cf. Figure 3) which is avoided in LFG-DOT (cf. (12)). Finally, the **probability** of target trees that are candidate translations is then calculated using the Tree-DOP probability model based on their relative frequencies in a treebank such as Figure 1. Importantly, therefore, DOT treebanks are *bags* of fragments, rather than sets. Accordingly, where duplication of 'similar' examples may be an impediment in other example-based systems, in DOT (and LFG-DOT) they are essential, as an increase in frequency of particular fragments directly contributes to the weight (in terms of probability) of the derivations in which they are involved. The probability of the parse tree of a particular translation is calculated by summing the probability of all possible derivations resulting in that parse tree. If different parse trees result for a particular translation, the respective probabilities of each with respect to the others can be calculated with respect to the corpus. Similarly, if different translations are presented as candidate target strings, their respective probabilities are given. As will will see with respect to (13)–(18) below, this is useful as certain MT systems cannot prevent the output of multiple translation candidates, even where some of these may be ungrammatical target strings. In such cases, an expert user is required to sift through these output strings to select the 'best' translation. This situation is avoidable where translations are output with probabilities, as in DOT and LFG-DOT, as translations may be ranked automatically (or pruned, if only the highest ranked translation is required).

Given this trivial treebank, only the one translation pair can be processed. If we assume that the treebank fragments for the translation pair *Peter laughs* $\Longleftrightarrow$ *Piet lacht* are added to Figure 1 (these new fragments would be identical except for the different lexical material on the leaves of the tree fragments), then the two new translations in (3) can be handled on the basis of fragments already in the database:

(3) a. *John laughs* $\longleftrightarrow$ *Jan lacht*

b. *Peter swims* $\longleftrightarrow$ *Piet zwemt*

For example, one possible derivation of (3a) is that in (4):

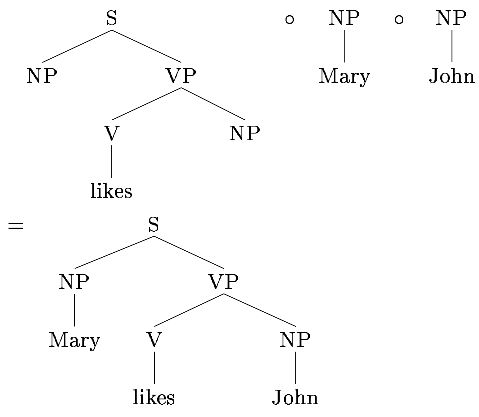
(4)


That is, the V nodes in the lower tree pair can be substituted at the appropriate ⟨*source, target*⟩ V nodes in the upper tree pair. The full parse trees for the two sentences ensue, resulting in a *bona fide* translation given this treebank. The two translations in (3) will have a slightly smaller probability than the two original translation pairs given the presence of their full linked parse trees in the treebank, each of which has a probability of $\frac{1}{12}$. That is, there is one occurrence of the actual parse trees themselves, and there are 12 trees with root `cat=s`. However, given the trivial nature of the examples, there will be no other translation candidates output so each translation will have a probability of 1. This will, of course, not always be the case with more complex examples. For instance, in section 4 we examine in two experiments the situation where multiple translations are output in LFG-DOT with accompanying probabilities given the existence of both 'default' and 'specific' fragments in the database.

2.1.1 A Problem for DOT1

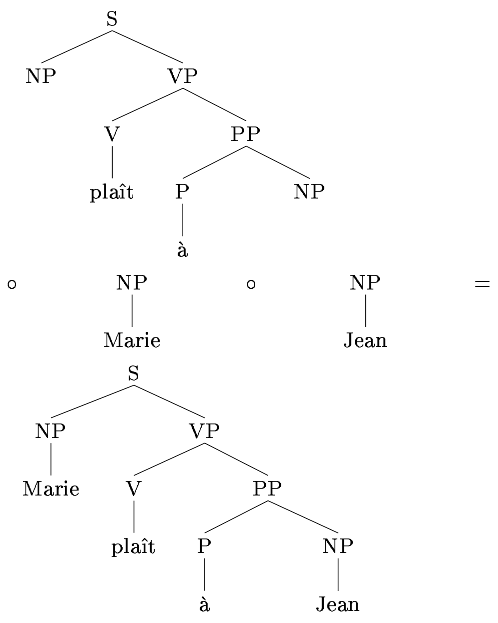
The DOT1 composition operation, based on leftmost substitution in the source trees, does not deal properly with translation cases where the word order differs significantly between two languages. Such an example is the *like* $\longleftrightarrow$ *plaire* relation-changing case. As soon as derivation (5) of the source sentence is arrived at, the desired linking of the English SUBJ with the French prepositional OBJ, and that of the English OBJ with the French SUBJ, are overridden by the composition operation of DOT1:

(5)



In this case the wrong translation of (5) is derived, as in (6):

(6)



It would appear that the adherence to leftmost substitution in the target given *a priori* leftmost substitution in the source is too strictly linked to the linear order of words, so that as soon as this deviates to any significant extent even between similar languages, DOT1 has a huge bias in favour of the incorrect translation. Even if the correct, non-compositional translation is achievable in such circumstances, it is likely to be so outranked by other wrong alternatives that it will be dismissed, unless all possible translations are maintained for later scrutiny by the user.

2.2 DOT2: An Improved Model of Translation

Poutsma overcomes this problem in DOT2 by redefining the composition operation of DOT1 to operate on *pairs* of trees, rather than on single trees. This new definition of composition ensures, among other things, that relation-changing cases such as *like* $\longleftrightarrow$ *plaire* are handled correctly: instead of the wrong derivation (6) of the source (5), we now obtain the correct translation *Jean plaît à Marie.* In DOT1, leftmost substitution in the source overrides the desired DOT-links between the French subject NP and the English object, as well as those between the French prepositional object and the English subject, given that composition is defined on the source tree *only*. Once pairs of trees are taken into account in DOT2, these links ensure the correct translation. Way (2001) shows that while DOT2 is able to handle certain 'hard' cases correctly, other examples, such as headswitching, are dealt with in a 'semi-compositional' manner. Consider the data in (7):

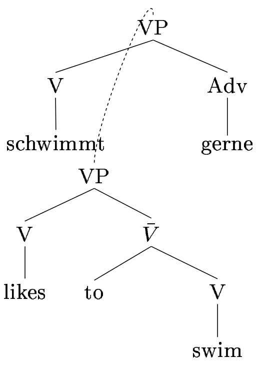
(7) a. DE: Johannes schwimmt gerne ⟷ EN: John likes to swim.
    b. DE: Josef läuft zufällig ⟷ EN: Joseph happens to run.

Presupposing the derivation of a monolingual treebank constructed from the German examples in (7), with two different NPs, verbs and adverbs, eight sentences are possible and can receive analyses with associated probabilities with respect to that DOP corpus. However, only four of these possible sentences can receive translations in a DOT corpus, namely the examples in (7) as well as those in (8), by simple substitution of the alternate NPs into the respective subject slots:

(8) a. DE: Josef schwimmt gerne ⟷ EN: Joseph likes to swim.
    b. DE: Johannes läuft zufällig ⟷ EN: John happens to run.

The other four possible sentence pairs cannot be handled at all in a DOT treebank built from analyses of the strings in (7). This is due to the fact that the linked VP pairs are not broken down any further than at the root level. The contrast can be seen by examining the *schwimmt gerne* VP and its constituent DOP-fragments in Figure 2 and (9), which contains the single DOT linked VP pair:

(9)



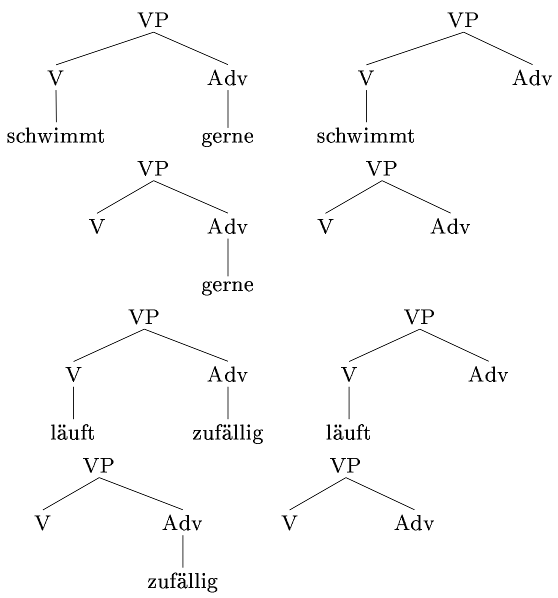


Figure 2: The monolingual DOP VP-fragments for a treebank built from the German examples in (7)

We cannot draw a link between *schwimmt* and *swim* in (9) as they are not translationally equivalent: one is a finite verb and the other an infinitive. We cannot, therefore, express the basic translation relations as ⟨*gerne, likes to*⟩ and ⟨*zufällig, happens to*⟩. Given this restriction, the only way that the other sentence pairs can be handled is if there is some prior linked pair *läuft gerne* ⟷ *likes to run* as well as a prior instance of the linked pair *schwimmt zufällig* ⟷ *happens to swim*. This is because these linked VP pairs are handled non-compositionally in DOT2 between German and English, but the monolingual VPs are treated compositionally in DOP. Contrast this situation with a DOT treebank designed to translate these 8 German strings into Dutch. Our starting point could be the German strings in (7) with their Dutch translations, as in (10):

(10) a. DE: Johannes schwimmt gerne ⟷ NL: Johan zwemt graag.
     b. DE: Josef läuft zufällig ⟷ NL: Josef loopt toevallig.

Given these simple transfer (i.e. 'word for word') examples, a DOT treebank would resemble much more closely the monolingual DOP treebanks from which it is derived for the respective sentences in (10), as *every* constituent part of the German strings corresponds exactly to a constituent part of the Dutch strings. In the DOT treebank these links are made explicit. When we have a headswitching case, however, it is apparent that both DOT models would translate the sentences correctly *iff* prior examples of linked headswitching VPs exist in the treebank. Such translations would receive extremely low probabilities with respect to the corpus in the normal case as they are built with a minimal degree of compositionality (substitution of subject NP, no other derivations being possible). As these examples only ever occur rarely, the chances of DOT2 managing to translate these in practice becomes significantly lower than might otherwise be expected, as we require not only the presence of the adverb, but also its occurrence to be correlated exactly with the verb in question for translation to succeed. We shall show that the LFG-DOT4 model of translation is able to provide generalized translation fragments which enable fully compositional translation in these cases, as required.

2.3 Boundary Friction in DOT

It is clear that DOT2 is an improvement on DOT1. DOT1 cannot always explicitly relate parts of the source language structure to the corresponding, correct parts in the target structure, so fails to translate correctly where source and target strings differ significantly with respect to word order. In DOT2, correct translations are obtained along with some possible wrong alternatives. Notwithstanding the improved composition operation and probability model of DOT2, its ability to achieve the correct translation is compromised by a lesser amount of compositionality in the translation process. Given the small number of fragments playing a role in the derivation of some translations involving complex phenomena, almost the exact linked sentence pair may need to be present in order for a translation to be possible.

A further problem is that Poutsma's DOT models cannot distinguish ill-formed from well-formed

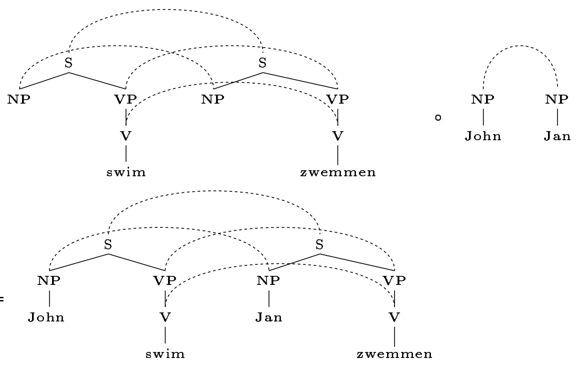


Figure 3: The Boundary Friction Problem in DOT

input. For example, both DOT models of translation would permit the derivations in Figure 3. That is, with no stipulation on subject-verb agreement, it is perfectly legitimate in DOP-based models (and EBMT systems based purely on trees, such as Watanabe 1992, as well, of course, as systems where examples are stored as pairs of strings, e.g. Somers *et al.*, 1990) to combine a singular subject with a plural verb and end up with a well-formed object. In DOT, we end up with a translation which is well-formed given the corpus. This example illustrates that while the presence of category information alleviates the problem of boundary friction to a certain extent (compared to 'pure' example-based methods, for instance, where the examples are stored as strings with no linguistic information present at all), this continues to be problematic for DOT in certain circumstances. We show in (12) that the analogous derivation in LFG-DOT would be deemed ungrammatical. That is, as soon as grammatical information is available via the accompanying f-structures, such a combination is impossible given the clash in NUM values for the subject and verb. Such ill-formed input can still be handled by relaxing grammaticality constraints such as these via *Discard*, but such translation pairs will be regarded as ungrammatical with respect to the corpus given their derivation via *Discard*; in DOT models, we have no such distinction.

Finally, of course, translation systems which are based purely on PS-trees will ultimately not be able to handle certain linguistic phenomena. DOP-based approaches are necessarily limited to those contextual dependencies actually occurring in the corpus, which is a reflection of surface phenomena only. Purely context-free models are insufficiently powerful to deal with all aspects of human language. Lexical Functional Grammar is known to be beyond context-free. It can capture and provide representations of linguistic phenomena other than those occurring at surface structure. With this in mind, the functional structures of LFG have been allied to the techniques of DOP to create a new model, LFG-DOP (Bod & Kaplan, 1998), which adds a measure of robustness (both with respect to unseen as well as ill-formed input) not available to models based solely on LFG.

3 LFG-DOT Models of Translation

Way (2001) presents four possible LFG-DOT translation models, all of which use LFG-DOP as their language models. LFG-DOP models are defined using the same four parameters as in Tree-DOP. Its **representations** are lifted *en bloc* from LFG theory, so that each string is annotated with a c-structure, an f-structure, and a mapping ϕ between them. Since we are now dealing with $\langle c, f \rangle$ pairs of structure, the *Root* and *Frontier* **decomposition** operations of DOP need to be adapted to stipulate exactly which c-structure nodes are linked to which f-structure fragments, thereby maintaining the fundamentals of c- and f-structure correspondence. Given that LFG c-structures are little more than annotated PS-trees allows us to proceed very much on the same lines as in Tree-DOP.

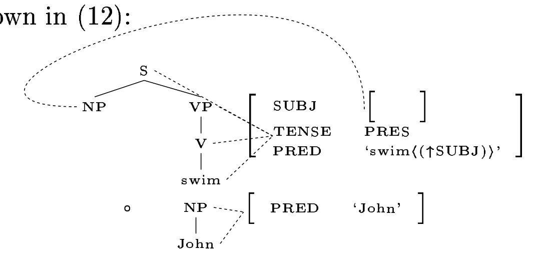
Root erases all nodes outside of the selected node, and in addition deletes all ϕ-links (informally, parts of the f-structure linked to a c-structure node, cf. the dotted lines in (12), for example) leaving the erased nodes, as well as all f-structure units that are not ϕ-accessible from the remaining nodes (for instance, features such as TENSE are deleted when the main verb is removed, as this feature—in English, at least—is inextricably linked to the verbal PRED). *Frontier* operates as in Tree-DOP, deleting all subtrees of the selected frontier nodes. Furthermore, it deletes all ϕ-links of these removed nodes together with any semantic form corresponding to the same nodes. Finally, a third, new operation, *Discard*, provides generalized fragments from those derived via *Root* and *Frontier* by freely deleting any combination of attribute-value pairs from an f-structure except those that are ϕ-linked to some remaining c-structure node, or that are governed by the local predicate. Its introduction also necessitates a new definition of the grammaticality of a sentence *with respect to a corpus*, namely any sentence having at least one derivation whose fragments are produced only by *Root* and *Frontier* and not by *Discard*. **Composition** is also a two-step operation. C-structures are combined by leftmost substitution, as in Tree-DOP, subject to the matching of their nodes. F-structures corresponding to these nodes are then recursively unified, and the resulting f-structures are subjected to the grammaticality checks of LFG. Finally, the **probability model** of LFG-DOP is again based on the relative frequency of a fragment, which in this case is a $\langle c, f \rangle$ pair. Bod & Kaplan give definitions of three possible competition sets from which fragments are sampled, depending on which point during the sampling stage the well-formedness conditions (the *Root* matching condition of DOP, or the Uniqueness, Completeness and Coherence conditions of LFG) are invoked. Note that given the non-monotonic property of the Completeness check, this can only be enforced after all other validity sampling has taken place.

3.1 LFG-DOT Model 1: Translation via τ

The first two LFG-DOT models propose the use of LFG's τ-equations to relate translation fragments between languages, the second in combination with γ, the function that links DOT source and target subtree fragments. Using τ-equations overcomes some of the problems of the DOT1 translation model. For instance, the LFG-MT solution (11) to the *like* $\longleftrightarrow$ *plaire* relation-changing case can be availed of quite straightforwardly:

$$\text{(11)} \quad \textit{like}: \quad \begin{aligned} &(\tau\!\uparrow \text{PRED}) = \text{plaire} \\ &\tau(\uparrow \text{SUBJ}) = (\tau\!\uparrow \text{OBL}) \\ &\tau(\uparrow \text{OBJ}) = (\tau\!\uparrow \text{SUBJ}) \end{aligned}$$

That is, the subject of *like* is translated as the oblique argument of *plaire*, while the object of *like* is translated as the subject of *plaire*. In addition, DOP adds robustness to LFG-MT. LFG-DOT models of translation contain two monolingual LFG-DOP language models, so *Discard* can be run on both source and target sides. This means that LFG-DOT can cope with ill-formed or previously unseen input which LFG-MT would not be able to handle at all. Suppose that *John swim* is encountered as the source string, as in Figure 3. In LFG-DOP, this can only be interpreted if *Discard* relaxes certain constraints in the f-structures. One such derivation is shown in (12):

(12)


That is, the SUBJ:NUM:PL path will have been relaxed in the sentential f-structure, with the NUM = SG feature removed from the lower NP f-structure. The NP c-structure can be substituted at the NP node in the upper c-structure, and the f-structures unified. The resultant f-structure would be input into the translation phase, which in LFG-DOT1 is quite simply the LFG-MT τ function. Taking the f-structure resulting from the derivation in (12) as input, the appropriate τ-equations would build the corresponding target Dutch f-structure, which would be linked by the target language LFG-DOT model to the appropriate c-structure tree, as in Figure 4.

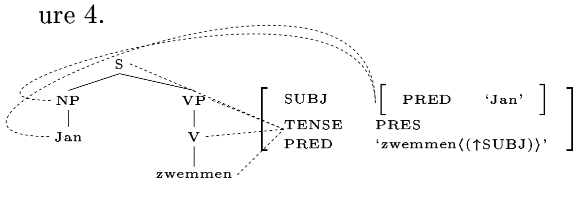


Figure 4: Robustness via *Discard* in LFG-DOT

The 'translation' of the ill-formed string *John swim* would therefore be *Jan zwemmen*. This shows that this particular DOT problem of boundary friction is resolved in LFG-DOT, owing to the presence of the syntactic f-structure information. We report further on the outstanding effect of boundary friction in LFG-DOT in section 4.

3.2 LFG-DOT Model 2: Translation via τ and γ

LFG-DOT2 requires the integration of the τ and γ mappings. Maintaining an f to f' translation engine in addition to γ increases the likelihood of achieving the correct translation—even if this is not proposed as the most probable translation via τ, given that this function will only ever produce very few translation candidates, we can guarantee in almost all cases that it is suggested as one of a small set of candidate translations. These can be compared to the best translation generated by γ and the highest ranking overall translation selected as output.

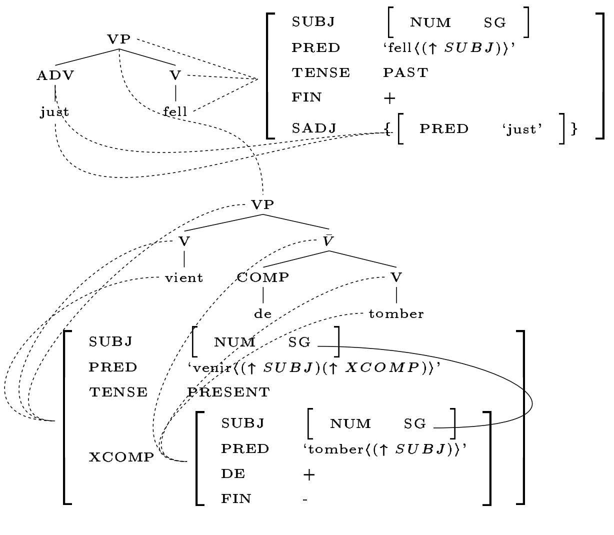


Figure 5: The *just* $\longleftrightarrow$ *venir de* case in LFG-DOT3

However, the τ mapping cannot always produce the desired translation, so that most of the LFG-MT problems (notably, failure to deal successfully with certain headswitching examples, cf. Figure 5–Figure 7) are imported into both LFG-DOT1 and LFG-DOT2 models of translation.

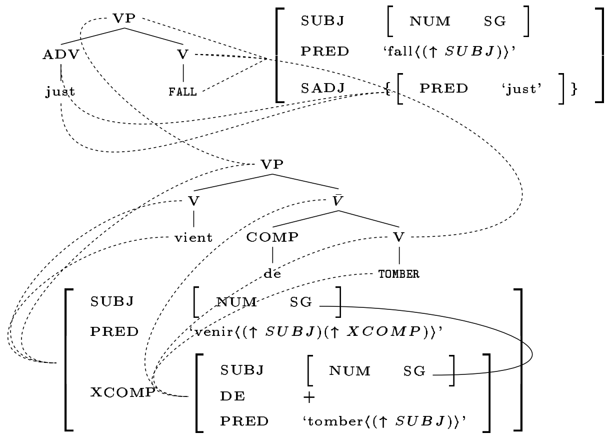


Figure 6: Lemmatization in LFG-DOT4

In contrast, Way (2001) shows that γ is always (depending, of course, on the coverage in the treebank) able to produce the correct translation, along with some possible wrong alternatives. The next two LFG-DOT models, therefore, abandon τ-equations and rely solely on γ to express the translation relation.

3.3 LFG-DOT Model 3: Translation via γ with Monolingual Filtering

The LFG-DOT3 model contains the DOT2 links between source and target c-structures, but with additional syntactic functional constraints which prevent ungrammatical structures such as Figure 3 from being formed (except via *Discard*, in much the same way as in (12) above), thereby enabling truly grammatical translations to be output, as opposed to translations which are grammatical only 'with respect to the corpus'. The f-structure information can be seen, therefore, as useful for monolingual disambiguation in both source and target sides. Ill-formed or unknown input is still processable by running *Discard* over the set of linked source and target $\langle c$, LFG-DOP-ϕ, $f\rangle$ fragments.

As an example, consider the *just* $\longleftrightarrow$ *venir de* headswitching case. In terms of LFG-DOT3, the translation relation is shown in Figure 5.

The semantically equivalent source and target c-structures are linked at the VP level via γ (omitted here for reasons of clarity). We do not consider *fell* to be semantically equivalent to *tomber* owing to their different FIN(ite) values, added to the fact that *fell* has a TENSE value whilst *tomber* does not. Hence this translation fragment can only be reused by substituting this pair with associated singular NP subjects at the appropriate nodes in an S-linked fragment. In this respect, as with DOT2, this LFG-DOT3 model continues to suffer from limited compositionality. We address this concern further in the next section which describes the LFG-DOT4 model.

3.4 LFG-DOT Model 4: Translation via γ and 'Extended Transfer'

In the previous section, we observed that the outstanding problem with LFG-DOT3 is its retention of the DOT2 problem of limited compositionality. Returning to the *just* $\longleftrightarrow$ *venir de* headswitching case in Figure 5, we would like to be able to 'relax' some of the constraints in order to map $\langle fell, tomber\rangle$ to make these linked fragments more general, and hence more useful. In so doing, we would remove this problem of limited compositionality.

In LFG-DOT4, the basic translation relation is expressed by γ, as in LFG-DOT3. In LFG-DOT4, however, there is a second application of *Discard*, by which 'lemmatized' forms are arrived at on which 'extended transfer' can be performed. *Discard* relaxes constraints in order to produce a set of generalized fragments with the potential to deal with ill-formed or unknown input. Once the `TENSE` and `FIN` features have been relaxed on the lowest verbs in both fragments in Figure 5, they can be regarded as translationally equivalent. Given this, $\langle fell, tomber\rangle$ are linked and lemmatized, as in Figure 6.

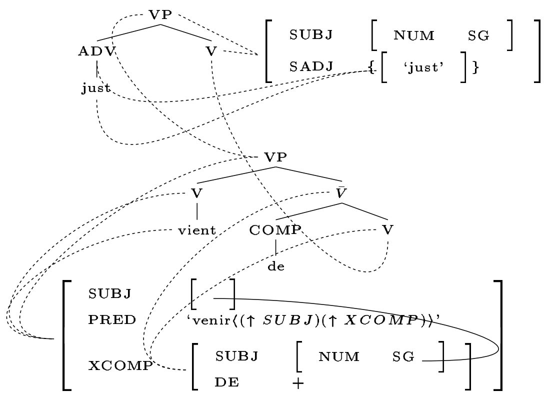


Figure 7: Generalized form of the *just* ⟷ *venir de* translation relation in LFG-DOT4

Now that ⟨FALL,TOMBER⟩ are linked, they can be deleted to produce the generalized form of the translation relation in Figure 7, as required. If fragment pairs such as those in Figure 7 prove subsequently to be of use in combining with other fragments, any resultant translation will be marked as ungrammatical with respect to the corpus, given that *Discard* was used in its derivation. Nevertheless, even if we restrict the impact of *Discard* on the probability space (cf. Way 2001, in order to ensure a preference for well-formed analyses derived via *Root* and *Frontier* over those produced by *Discard*), such translations will receive *some* probability, whereas the semi-compositional variants from which they were derived may not be able to produce *any* translation in practice.

4 Boundary Friction in LFG-DOT: two Experiments

Both DOT and LFG-DOT have strict definitions of fragment boundaries. We showed that whereas DOT cannot distinguish well-formed from ill-formed structures, LFG-DOT has an intuitively correct notion of grammaticality. Nevertheless, the thorny issue of boundary friction does raise its head in LFG-DOT to a degree.

All MT systems have to decide what are legitimate translation candidates. In most rule-based systems, default rules are differentiated from specific rules, with the former applying only in those cases where a specific rule cannot. Watanabe (1994) discusses the problem of boundary friction (he calls it 'example interference'), and provides a method of distinguishing exceptional from general examples in EBMT on the basis of *similarity* of examples (cf. Sato & Nagao, 1990, who use a similar technique based on thesaurus relations). Once patterns are identified as general, exceptional or neutral, some of the side-effects of boundary friction may be overcome.

Not all rule-based systems can prevent the output of a wrong, compositional translation once a specific translation has been obtained. For instance, in LFG-MT, satisfying the requirement that only possible translations are produced is problematic where the translation of a lexical head is conditioned in some way by one of its dependants, as in (13):

(13) commit suicide ⟷ se suicider

The problem is that in these cases, suppressing the wrong, compositional translation in LFG-MT is impossible. For instance, we require the default rules in (14):

(14) a. commit ⟷ commettre

b. suicide ⟷ suicide

Such rules are expressed in LFG-MT by the lexical entries in (15):

(15)

$$\begin{array}{ll} \textit{commit}: & (\tau\uparrow \text{PRED}) = \text{commettre} \\ & \tau(\uparrow \text{SUBJ}) = (\tau\uparrow \text{SUBJ}) \\ & \tau(\uparrow \text{OBJ}) = (\tau\uparrow \text{OBJ}) \\ & \\ \textit{suicide}: & (\tau\uparrow \text{PRED}) = \text{suicide} \end{array}$$

These entries show how *commit* and *suicide* are to be translated under normal circumstances, such as in (16):

(16) a. Jean commet un crime $\longleftrightarrow$ John commits a crime

b. Le suicide est tragique $\longleftrightarrow$ Suicide is tragic

Nevertheless, given the default, compositional entries in (15), LFG-MT produces the wrong translation in (17):[2]

(17) John commits suicide $\longleftrightarrow$ *Jean commet le suicide

LFG-MT can, however, derive the correct translation *John se suicide* in such cases via the solution in (18):

(18)
$$\begin{array}{ll} \textit{commit}: & (\tau\uparrow \text{ PRED}) = \text{se suicider} \\ & \tau(\uparrow \text{ SUBJ}) = (\tau\uparrow \text{ SUBJ}) \\ & (\uparrow \text{ OBJ PRED}) =_c \text{suicide} \end{array}$$

Here the collocational units '*commit + suicide*' are linked as a whole to *se suicider*. The $=_c$ equation is a constraining equation: rather than expressing mere equality, it constrains the PRED value of the OBJ of *commit* to *suicide* when it is to be translated as a whole into *se suicider*. The selective use of constraining equations enables correct translations to be derived which would only be possible in other systems by tuning. Nevertheless, the point remains that in LFG-MT we would get both translations here, i.e. a correct one and a wrong one, since it is not possible to enforce the requirement that specific rules ought to override the default translational rules where applicable.

4.1 Experiment 1

We tested the issue of default versus specific translations in LFG-DOT3. We produced an LFG-DOT3 treebank containing all the linked fragments from the sentences in (19):

[2]Note that the rules in (14) are *bona fide* translation rules that any rule-based English-French MT system will require. It is, therefore, the task of the French generation component to explicitly rule out the incorrect translation in (17), *not* the transfer component.

(19) a. Le suicide est tragique $\longleftrightarrow$ Suicide is tragic.

b. Jean commet le crime $\longleftrightarrow$ John commits the crime.

c. Jean commet le meurtre $\longleftrightarrow$ John commits the murder.

d. Jean dort $\longleftrightarrow$ John sleeps.

e. Marie se suicide $\longleftrightarrow$ Mary commits suicide.

f. Marie commet un attentat $\longleftrightarrow$ Mary commits an attack.

g. Marie commet la faute $\longleftrightarrow$ Mary commits the mistake.

h. Pierre commet un arbitre $\longleftrightarrow$ Peter nominates an arbitrator.

i. Pierre commet une erreur $\longleftrightarrow$ Peter commits an error.

j. Pierre commet une injustice $\longleftrightarrow$ Peter commits an injustice.

Here there are seven instances of *commettre* (six of which translate as *commit*) as opposed to only one instance of *se suicider*.

Before examining results obtained with LFG-DOT, it is insightful to point out that in the monolingual French LFG-DOP (no *Discard*) treebank built from the French strings in (19), *Marie se suicide* is preferred about 2.6 times over the compositional alternative *Marie commet le suicide*. If these were the output translations, then ranking them against one another would favour *Marie se suicide* with about 72% versus *Mary commet le suicide* with about 28%. In the LFG-DOT3 treebank produced from the English and French sentences in (19), the specific translation is preferred even more than in the French monolingual LFG-DOP treebank. We set out to test the weight of the specific over the compositional translation for the sentences in (20):

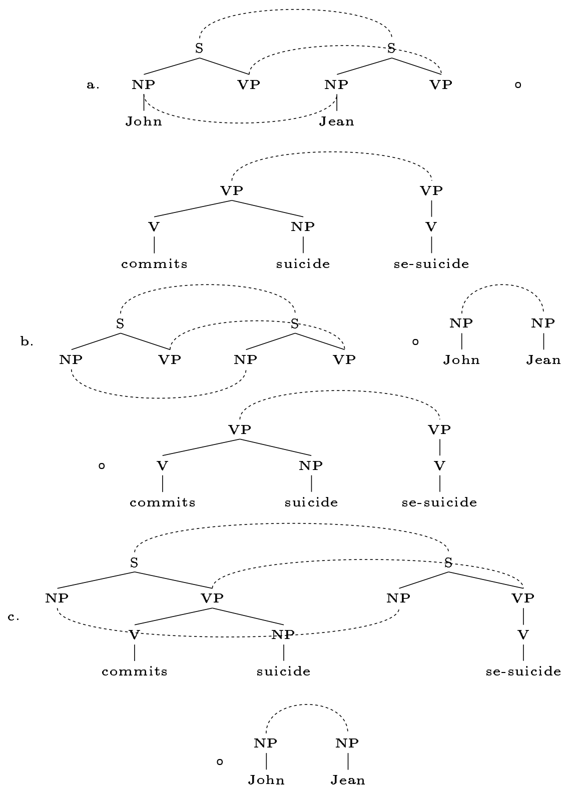


Figure 8: C-structure derivations for *John commits suicide* ⟺ *Jean se suicide*

(20) a. John commits suicide ⟷ Jean se suicide

b. Mary commits suicide ⟷ Marie se suicide

c. John commits suicide ⟷ *Jean commet le suicide

d. Mary commits suicide ⟷ *Marie commet le suicide

Translation (20a) can be built using the three derivations in Figure 8.[3] (20b) has the additional derivation of the full trees (and accompanying f-structures) for this sentence pair. The probabilities of (20a-b) are shown in (21):

(21) a. P(John commits suicide ⟷ Jean se suicide) = 0.000705 ($\simeq \frac{1}{1419}$)

b. P(Mary commits suicide ⟷ Marie se suicide = 0.006229 ($\simeq \frac{1}{161}$)

[3]We have omitted the accompanying f-structure fragments for reasons of space.

For each of the translations in (20c-d) there are 7 derivations with total probability 0.000501 ($\simeq \frac{1}{1998}$). Now we can rank each translation with respect to the other in (22):

(22) a. P(John commits suicide $\overset{T}{=}$ Jean se suicide) = 705/1206 = 0.5846

b. P(John commits suicide $\overset{T}{=}$ Jean commet le suicide) = 501/1206 = 0.4154

c. P(Mary commits suicide $\overset{T}{=}$ Marie se suicide = 6229/6750 = 0.923

d. P(Mary commits suicide $\overset{T}{=}$ Marie commet le suicide) = 521/6750 = 0.077

Here $\mathcal{T} \overset{T}{=} W$ means that $\mathcal{T}$ is a translation of a word string W. Therefore we can see that for *John commits suicide*, the correct, specific translation is about 1.4 times more likely than the wrong, default, compositional translation, whereas for *Mary commits suicide* the specific translation is preferred about 12 times more than the default translation. We see in (22) the dominance of the exact linked translation pair over the alternative translation. The presence of the exact translation (19e) is insufficient to explain the preference for the specific translation for *John commits suicide*: despite the presence of six *commit* ⟷ *commettre* examples in (19) compared to only the single instance of *commits suicide* ⟷ *se suicide*, the specific translation is nonetheless preferred.

4.2 Experiment 2

How many more times empirically we can expect to see *commit* ⟷ *commettre* compared to *commits suicide* ⟷ *se suicide*? In the LOB Corpus, there are 66 instances of *commit* (including its morphological variants), only 4 of which have *suicide* as its object, out of the 15 occurrences of *suicide* as an NP. Consequently, even for this small sample, we can see that 94% of these examples need to be translated compositionally (by *commettre* + NP), while only the *commit suicide* examples require a specific rule to apply (i.e. *se suicider*).

In the on-line Canadian Hansards covering 1986-1993, there are just 106 instances of *se suicider* (including its morphological variants). There will, of course, be many more instances of *commettre.* Given occurrences of *suicide* as an NP in French corpora, it is not an unreasonable hypothesis to expect that wrong translations such as (17) will be much more probable than those derived via the specific rule. However, this hypothesis is shown to be inaccurate in the above experiment.

Furthermore, it is clear from the results in (22) that a ratio of 6:1 is insufficient to achieve a bias in favour of the wrong, compositional translation in LFG-DOT. Running a new experiment with a treebank built from 5 instances of each translation pair (19a-d) and (19f-j) and just the one instance of (19e), making a total of 46 sentences in all, produces the results in (23):

(23) a. P(John commits suicide $\overset{T}{=}$ Jean se suicide) = 132/635 = 0.208

b. P(John commits suicide $\overset{T}{=}$ Jean commet le suicide) = 503/635 = 0.792

c. P(Mary commits suicide $\overset{T}{=}$ Marie se suicide = 1206/1758 = 0.686

d. P(Mary commits suicide $\overset{T}{=}$ Marie commet le suicide) = 552/1758 = 0.314

Now, with 30 instances of *commit* $\longleftrightarrow$ *commettre* and only the one *commits suicide* $\longleftrightarrow$ *se suicide* example, we see that the wrong, default, compositional translation for *John commits suicide* is now preferred by about 3.8 times, but the presence of the exact translation (19e) maintains the preference for the specific translation for *Mary commits suicide* by about 2.2 times. Consequently we can see that it will take many more instances of *commit* $\longleftrightarrow$ *commettre* before the specific translation for *Mary commits suicide* is outranked by the wrong, compositional alternative.

5 Conclusions and Future Work

Models of translation based on DOP and LFG-DOP translate new strings on the basis of linked ⟨*source, target*⟩ fragments already located in their databases. Accordingly, such systems may be viewed as example-based systems.

We described the DOT models of translation based on DOP. DOT1 is not guaranteed to produce the correct translation when this is non-compositional and considerably less probable than the default, compositional alternative. DOT2 addresses the failings of DOT1 by redefining the composition operation. In contrast to DOT1, DOT2 cannot fail to produce correct candidate translations, along with some possible wrong alternatives, depending of course on the corpus from which fragments are derived. Despite the presence of syntactic information in the tree-structure fragments, we showed that both DOT models continue to suffer from the problem of boundary friction in cases where singular and plural fragments are combined.

We also described a number of new hybrid models of translation which use LFG-DOP as their language models. The first, LFG-DOT1, imports the τ-equations from LFG-MT as the translation relation. LFG-DOT1 improves the robustness of LFG-MT through the use of the LFG-DOP *Discard* operator, which produces generalized fragments by discarding certain f-structure features. It can, therefore, deal with ill-formed or previously unseen input where LFG-MT cannot. Unsurprisingly, however, all of the other problems of LFG-MT are maintained in LFG-DOT1.

Given this, we augmented LFG-DOT1 with the γ function from DOT2 to give an improved model of translation. LFG-DOT2 maintains the τ translation relation to increase the chances of the correct translation being produced. Nevertheless, given that the τ-equations fail to derive the correct translation in all cases, we omitted the τ translation relation from our subsequent models.

LFG-DOT3 relies wholly on γ to express the translation relation, and uses f-structure information purely for monolingual filtering. The presence of this functional information prevents the formation of certain ill-formed structures which can be produced in DOT. LFG-DOT models, therefore, have a notion of grammaticality which is missing from DOT models. While both DOT and LFG-DOT contain strict notions of boundary definition, DOT allows the output of structures which are well-formed according to the corpus, but which are syntactically ungrammatical. The definition of well-formedness in LFG-DOT, in contrast, corresponds exactly to our understanding of grammaticality in the wider sense. However, both DOT2 and LFG-DOT3 models suffer from limited compositionality, so that in some cases the minimal statement of the translation relation is impossible.

LFG-DOT4 adds an 'Extended Transfer' phase to LFG-DOT3 by producing lemmatized forms using a second application of *Discard*. This extension overcomes the problem of limited compositionality, enabling the statement of the translation relation in an intuitive, concise fashion.

Finally, we demonstrated that LFG-DOT models of translation suffer less from the problem of boundary friction than DOT models given the presence of the additional syntactic f-structure information. In addition, we showed in two small experiments that despite attempting to 'load the dice' in favour of the wrong, compositional translation over the correct, specific alternative, LFG-DOT continues to translate in a robust fashion.

The work described here and in Way (2001) uses as its evaluation metric the ability to cope with 'hard' translation cases, such as relation-changing (cf. (5)–(6)) and headswitching (cf. (7)–(10)). Special LFG-DOT corpora such as those derived to test the effect of Boundary Friction in section 4 needed to be created. The translation effects examined here need to be tested further on larger corpora, and the work of Frank *et al.* (2001) on semi-automatic derivation of LFG corpora from treebank resources would appear promising in this regard.

Furthermore, the hypotheses developed in Way (2001) need to be further explored with respect to simpler translation data, such as *Fido barks* sentences. Different probability models will also be evaluated (cf. Bonnema *et al.*, 2000), as will the possibility of pruning the search space, by cutting down the number of fragments produced (cf. Sima'an, 1999) in order to improve the efficiency of the models proposed.

References

[1] Bod, R. (1998): *Beyond Grammar: An Experience-Based Theory of Language*, CSLI, Stanford, CA.

[2] Bod, R. and R. Kaplan (1998): 'A Probabilistic Corpus-Driven Model for Lexical-Functional Analysis', in *Proceedings of the 17th International Conference on COLING and 36th Conference of the ACL*, Montreal, Canada, **1**:145–151.

[3] Bonnema, R., P. Buying and R. Scha (2000): 'Parse Tree Probability in Data-Oriented Parsing', in *Proceedings of PACLING*, Mexico City.

[4] Frank, A., J. van Genabith and A. Way (2001): 'Treebank vs. X-BAR based Automatic F-Structure Annotation', in *Proceedings of 6th International Conference on Lexical Functional Grammar (LFG-2001)*, Hong Kong.

[5] Kaplan, R. and J. Bresnan, (1982): 'Lexical Functional Grammar: A Formal System for Grammatical for Grammatical Representation', in J. Bresnan (ed.) *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, Mass., pp.173–281.

[6] Kaplan, R., K.. Netter, J. Wedekind and A. Zaenen (1989): 'Translation by Structural Correspondences', in *Fourth Conference of the EACL*, Manchester, pp.272–281.

[7] Mima, H, H. Iida and O. Furuse (1998): 'Simultaneous Interpretation Utilizing Example-Based Incremental Transfer', in *Proceedings of the 17th International Conference on COLING and 36th Conference of the ACL*, Montreal, Canada, pp.855-861.

[8] Poutsma, A. (1998): 'Data-Oriented Translation', in *Ninth Conference of Computational Linguistics In the Netherlands*, Leuven, Belgium.

[9] Poutsma, A. (2000): *Data-Oriented Translation: Using the Data-Oriented Parsing framework for Machine Translation*, MSc Thesis, University of Amsterdam, The Netherlands.

[10] Sato, S. and M. Nagao (1990): 'Towards Memory-based Translation', in *13th International Conference on COLING*, Helsinki, Finland, **3**:247–252.

[11] Sima'an, K. (1999): *Learning Efficient Disambiguation*, PhD Thesis, University of Utrecht, The Netherlands.

[12] Somers, H. (1999): 'Example-based Machine Translation', *Machine Translation* **14**(2):113–157.

[13] Somers, H., I. McLean and D. Jones (1994): 'Experiments in Multilingual Example-Based Generation', in *Proceedings of the 3rd Conference on CSNLP*, Dublin, Ireland.

[14] Somers, H., J. Tsujii and D. Jones (1990): 'Machine Translation without a source text', in *13th International Conference on COLING*, Helsinki, Finland, **3**:271–276.

[15] Watanabe, H. (1992): 'A Similarity-Driven Transfer System', in *14th International Conference on COLING*, Nantes, Frances, pp.770–776.

[16] Watanabe, H. (1994): 'A Method for Distinguishing Exceptional and General Examples in Example-based Transfer Systems', in *15th International Conference on COLING*, Kyoto, Japan, pp. 39–44.

[17] Way, A. (2001): *LFG-DOT: A Hybrid Architecture for Robust MT*, PhD Thesis, University of Essex, UK.