Translation Pattern Extraction and Recombination for Example-Based Machine Translation

A thesis submitted to the University of Manchester Institute of Science and Technology for the degree of Doctor of Philosophy

Kevin McTait
Centre for Computational Linguistics
Department of Language Engineering
UMIST
September 2001

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or institute of learning.

Abstract

An approach to Example-Based Machine Translation is presented which operates by extracting and recombining *translation patterns* from a bilingual corpus aligned at the level of the sentence. The translation patterns are extracted using a recursive machine-learning algorithm based on the principle of similar distributions of strings: source and target language lexical items that co-occur in the same two sentence-pairs are likely to be translations of each other. The translation patterns extracted represent generalisations of sentences that are translations of each other in that certain sequences of words are replaced by variables. The translation patterns resemble, to a certain extent, transfer rules but with less constraints since there is no concept of syntactic structure in this approach: translation patterns are extracted based on the information inherent in the corpus. The strings and variables of which the translation patterns are composed are aligned in order to provide a more refined bilingual knowledge source, necessary for the recombination phase where target language translations are produced. They are aligned by means of language-neutral techniques – cognates and bilingual lexical distribution – in order to maintain language-neutrality and portability.

A non-structural approach based on the distributions of surface forms in a corpus is error prone and liable to the extraction of translation patterns that are false translations. This thesis highlights some of the sources of those errors and proposes solutions to them based on the addition of external linguistic knowledge. First, the basic approach is augmented with morphological analysis. Second, part-of-speech tagging is incorporated. This results in three variants of the approach, each with increasing amounts of external linguistic resources included. The performance of each variant is assessed and a comparison is made between them in order to test the hypothesis that an increase in linguistic knowledge improves performance, in terms of both recall and precision.

Acknowledgements

First, I would like to thank Dr Arturo Trujillo, my initial supervisor, for assisting me in getting started with this project and for his efforts, comments, ideas and advice that have assisted me throughout his time as supervisor. I would also like to thank Harold Somers for supervising the remainder of this thesis and providing input, advice and ideas where necessary in order to complete the project.

At the Centre for Computational Linguistics, I would like to thank John McNaught for his advice and input with respect to morphology. Thanks are also due to Bill Black for his input on implementation issues, as well as his role as internal examiner. Many thanks to Paul Johnston for his efforts at ensuring that I had the necessary computing facilities as well as his help with mathematical descriptions. Also, thanks to all staff and students at CCL who contributed ideas or advice in departmental seminars and thanks to the departmental administration staff for their assistance.

I acknowledge and thank the Economic and Social Research Council (ESRC) who funded this research (studentship award number: R00429834560).

Finally, I would like to thank my partner and my parents for all their support and encouragement throughout my Ph.D.

Table of Contents

Table of Figures

Table of Tables

List of Abbreviations

DP	Dynamic Programming
EBMT	Example-Based Machine Translation
EM	Expectation-Maximisation
FST	Finite State Transducer
LD	Levenshtein Distance
ML	Machine Learning
MT	Machine Translation
NLP	Natural Language Processing
POS	Part-of-Speech
RBMT	Rule-Based Machine Translation
SL	Source Language
SM	Simulated Annealing
SMT	Statistical Machine Translation
TL	Target Language
TM	Translation Memory

1 Introduction and Thesis Overview

The advantages of corpus-based approaches to machine translation (MT), known broadly as Example-Based Machine Translation (EBMT), have been made clear repeatedly in the literature: they are portable to new domains and language-pairs, more extensible than rule-based systems, often rapidly deployable and are tuned to a particular text-type or domain. A number of EBMT systems extract *translation patterns* or *templates* from bilingual texts (Kaji et al. 1992; Güvenir & Cicekli, 1998; Brown, 1999; Carl, 1999; McTait & Trujillo, 1999), which, to a certain extent, resemble transfer rules in 'traditional' Rule-Based Machine Translation (RBMT) systems. Translation patterns represent generalisations of sentences that are translations of each other in that sequences of one or more words are replaced by variables, possibly with alignments between the resulting word sequences and/or variables made explicit. The amount of constraints in the translation patterns varies with each approach. In order to produce target language (TL) translations, source language (SL) input sentences are matched against the patterns or templates, the TL equivalents of which combine to form an TL string.

Existing approaches to extracting translation patterns for EBMT are limited in certain respects. Many approaches require significant linguistic knowledge sources, such as parsers and bilingual dictionaries, in order to generalise translation examples (Kaji et al., 1992), with consequence for portability. Those that generalise translation examples by language-neutral methods are limited to comparing *pairs* of translation examples. This means that patterns are formed using textual evidence from a maximum of 2 sentence pairs (Güvenir & Cicekli, 1998; Echizen-ya, 2000; Malavazos & Piperidis, 2000). Furthermore, alignments between the strings and variables of which translation patterns are composed are often limited to a bijective or 1:1 type. Some methods are further limited by the fact that they extract translation patterns which contain only one generalisation or variable (Furuse & Iida, 1992). Translation patterns containing multiple alignments between text fragments and variables, as well as alignments of a more flexible type i.e. 2:1, 1:0, are more desirable and thus more conducive to describing translation phenomena between two languages. Structural divergences also present problems (Carl, 1999).

This thesis describes an approach to EBMT that extracts translation patterns from a bilingual corpus by language-neutral methods. The fact that translation patterns are formed from lexical items that occur a minimum of twice in the corpus means that the algorithm is useful in instances of sparse data. The alignment phase computes (possibly multiple) correspondences between the sub-sentential text fragments and variables of which the patterns are composed. Alignments may be of a bijective or non-bijective nature. Alignments are computed efficiently by a two-pass algorithm which finds not only alignments of an adjacent nature, but also non-adjacent alignments or long-distance dependencies, necessary for language pairs that are structurally divergent. The language-neutral nature of the translation pattern extraction algorithm is, however, error-prone and liable to the extraction of translation patterns that are 'false translations'. Solutions are proposed which involve adjusting frequency thresholds in the extraction algorithm and the addition of a minimal amount of linguistic knowledge. The knowledge added is kept to a minimum and of a variety that is easy to acquire in order to maintain portability.

In more detail, translation patterns are extracted from a bilingual corpus aligned at the level of the sentence (section 3.2). They are extracted by means of a language-neutral recursive machine-learning algorithm based on the principle of similar distributions of strings: SL and TL strings that co-occur in the same two (or more) sentence pairs of a bilingual corpus are likely to be translations of each other (3.2.1). The SL and TL strings that make up the translation patterns are aligned (3.2.3) 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 (3.3). 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*.

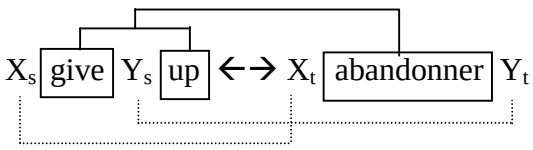

Figure 1: A Simple Translation Pattern

The translation pattern in Figure 1 contains a bijective 1:1 alignment between the variables, denoted by *X* and *Y* and also contains a non-bijective 2:1 alignment between the text fragments *give…up* and *abandonner*.

It is possible to extract translation patterns and align the text fragments of which they are composed when the data is sparse. Strings need to co-occur in a minimum of only 2 sentence pairs. Furthermore, language-neutral techniques, such as cognate matching and bilingual lexical distribution, are used to align the strings or text fragments of which the patterns are composed. Cognates facilitate alignment when the data is sparse.

An approach such as this, which is 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. The system architecture is depicted in Figure 2. The dotted lines indicate that the use of the knowledge sources is optional. The addition of linguistic resources is intended to improve both accuracy and recall.

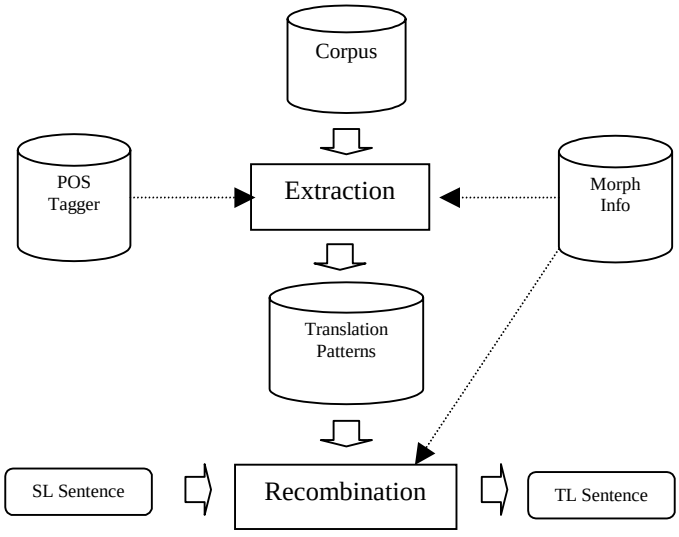

Figure 2: System Architecture

The linguistic knowledge is added in varying amounts. This results in three variants of this EBMT methodology. The first, as described above, is the basic language-neutral

variant based on the distributions of surface forms in a bilingual corpus, aided by a stop list of SL and TL closed-class words. The second variant is augmented further to include morphological analysis. The third variant is augmented further to include both morphological analysis and POS tagging. The performance of each variant is compared in order to test the hypothesis that an increase in linguistic knowledge improves performance.

1.1 Thesis Overview

Chapter 2 positions this thesis among the set of existing research projects in the EBMT literature. It highlights the main methodologies that this approach draws upon. For a comprehensive review of EBMT, the reader is referred to a recent critique of the state-of-the-art in EBMT by the author's supervisor (Somers, 1999). Therefore, this section will remain narrow and specific: most attention is paid to EBMT systems that extract translation patterns as these are the most relevant and similar research projects to this approach. Particular attention will be paid to systems that extract translation patterns by language-neutral methods. Of peripheral concern are other corpus-based approaches to translation, namely Statistical Machine Translation (SMT) and Translation Memory (TM) as well as word, term, phrase and sequence alignment algorithms.

In Chapter 3, the core algorithms of the basic approach for extracting and recombining translation patterns are presented. First, the language-neutral recursive machine-learning algorithm for extracting translation patterns is presented. Second, the proposed method for aligning the text fragments and variables of which they are composed is described. Finally, the recombination phase, which takes as input an SL input sentence and a set of translation patterns and outputs an TL translation, is described. Initial results are also presented.

The shortcomings of an approach based on the distributions of surface forms in a bilingual corpus are presented in Chapter 4. Solutions to these problems, such as the extraction of translation patterns that are false translations, are proposed by the addition of external linguistic knowledge sources. This results in three variants of this approach to EBMT, as described above. A comparison of the three variants is carried out to test the

hypothesis that an increase in linguistic knowledge leads to an increase in both accuracy and coverage.

Chapter 5 presents a more thorough evaluation of the three variants described where certain system parameters – i.e. frequency thresholds, corpus size, text type – are altered in order to observe whether any appreciable gains in performance can be achieved. A comparison with a well-known commercial MT system is also included.

Chapter 6 concludes the thesis with a summary and discussion of the main advantages of this approach and what it has succeeded in showing. Furthermore, the principal limitations are highlighted. In the light of this information, feasible future directions of research are proposed.

2 Relevant Research

The aim of this chapter is to provide an analysis of the existing approaches to EBMT that are in some way relevant to the approach presented in this thesis. In so doing, the position of this approach in relation to existing approaches is established. Other areas of Natural Language Processing (NLP) that are in some way relevant, such as alignment, are also presented. The result is a presentation and critical analysis of the methodologies that this approach draws upon, concluded by a summary of the areas in which improvement or extension is required.

2.1 EBMT System Types

The construction of 'traditional' rule-based (RBMT) or knowledge-based MT systems, such as the KANT system (Mitamura & Nyburg, 1992) and the EUROTRA project (Steiner, 1991), is a lengthy, laborious and error-prone process. It is difficult to produce hand-crafted transfer rules to cover a wide variety of input. Frequently, rule conflict can produce unpredictable side effects when new rules are added. In addition, there is no well-known linguistic theory of transfer (Melby, 1986). This led to the idea of translation by the analogy principle (Nagao, 1984; Sato & Nagao, 1990), which makes use of a set of previously translated sentences as opposed to the construction of hand-crafted monolingual grammars, bilingual lexicons and transfer rules. Since then, there has been an explosion of interest in approaches which use a bilingual corpus as the principal bilingual knowledge source. Such approaches use subtly different techniques and consequently take names to reflect this i.e. Statistical Machine Translation (SMT) etc. However, all are grouped together largely under the term EBMT, as new translations are computed on the basis of previous examples of translated text.

The resulting set of EBMT systems can be classified in many different ways. One recent comprehensive analysis is given in Somers (1999). Therefore, this chapter will concentrate on the EBMT methodologies that are of specific interest to the approach taken in the remainder of this thesis, namely the EBMT systems that operate specifically by extracting and recombining *translation patterns* from bilingual corpora.

The concept of EBMT based on the extraction and recombination of translation patterns presented in this thesis can be placed somewhere between 'traditional' linear EBMT systems – 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; Brown & Frederking, 1995; Brown 1996) – and systems that extract patterns that bear more resemblance to structural transfer rules (Kaji et al. 1992; Maruyama & Watanabe, 1992; Watanabe, 1995). This can be explained in more detail with reference to the three component stages of EBMT, as outlined by Nagao (1984), and the two-phase translation process (extraction and recombination of translation patterns) used by the author and the research groups presented in section 2.1.3.

According to Nagao, the translation process is divided into the three tasks of matching SL fragments of an input against a database of translation examples, identifying the corresponding TL fragments and then combining them appropriately to produce an TL string. These stages can be illustrated by means of the familiar pyramid diagram (Vauquois, 1968) with the tasks of EBMT superimposed in capitals (Figure 3). Analysis in conventional MT is replaced by the process of matching the SL input against the database of examples. The EBMT literature reports a wide variety of matching techniques (Matsumoto et al., 1993; Nirenburg et al., 1993; Cranias et al., 1994; Zhang & Shasha, 1997) . Once the set of example fragments have been selected the corresponding TL fragments are identified (alignment). This is the stage at which bilingual knowledge is generally extracted. The alignment of lexical items between two languages is also an area that has received considerable attention from the NLP community (section 2.2). Finally, the recombination stage assembles the TL fragments in an appropriate manner to produce an TL string, just as the generation stage in conventional MT puts together a string from a syntactic/semantic representation. Recombination techniques are generally specific to the EBMT approach taken and is designed according to the previous matching and alignment steps. This stage has generally received less attention in the literature and is often limited to a short description tacked onto the end of a research paper. Furthermore, the matching and recombination techniques used in EBMT can often be very similar to those used for analysis and generation in conventional MT. In hybrid systems, they may be identical.

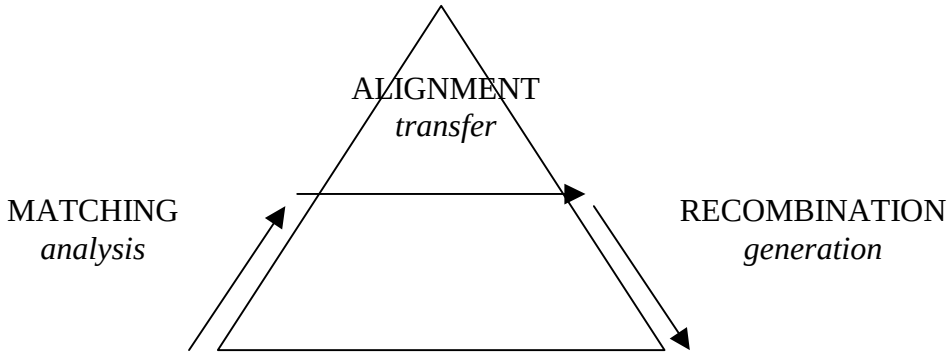

Figure 3: "Vauquois Pyramid" with EBMT Tasks Superimposed

The pyramid diagram (Figure 3) is slightly at odds with the two-phase model of translation, given in Kaji et al. (1992) and reproduced in Figure 4, in that there is an entire pre-stage to the translation process. The SL input is no longer matched against a database or corpus of raw translation examples, but is matched against a set of translation patterns or templates (exemplars) that have been automatically extracted or learned from the corpus. Exemplar-based representation has been widely used in the field of Machine Learning (ML). Medin & Schaffer (1978) originally proposed exemplar-based learning as a model of human learning. This pre-stage represents, to varying extents, an effort at automatically extracting a set of transfer rules similar to those in traditional RBMT systems. Kitano (1993) attempted to manually create such "transfer rules", but this process is error-prone for anything more than very small corpora.

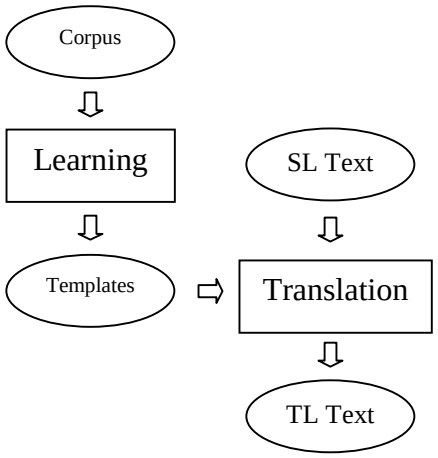

Figure 4: Two-Phase EBMT (Kaji et al., 1992: 672)

Depending on the approach, the SL and TL text fragments that make up the translation patterns may be aligned in order to provide a bilingual knowledge source or set of complete translation patterns that, to a certain extent, resemble transfer rules (the amount of constraints in the resulting translation patterns depends on the linguistic knowledge sources used in the extraction phase). Therefore, the alignment of subsentential fragments – if such a stage takes place – remains a distinct process and often occurs as part of the learning phase where translation patterns are extracted.

Matching and recombination are often conflated, or are at least represented that way in terms of the diagrams in the literature, into a single phase which is termed 'recombination' or 'translation'. However, this involves both matching the SL input against the translation patterns, instantiating any variables, and producing the TL string according to the pre-computed alignments between the text fragments of which the translation patterns are composed. Due to the fact that bilingual alignments have been made explicit or the fact that ordering information in the TL has been predetermined by the use of indices between SL and TL chunks, recombination is a simpler process than recombining the translations of partially overlapping exact matches of the SL input (see section 2.1.1).

In an attempt to position the approach in this thesis among the set of existing EBMT systems, section 2.1.1 briefly outlines some of the linear EBMT systems that adhere to the architecture given in Figure 3, while sections 2.1.2 and 2.1.3 describe in detail and provide a critical analysis of the extraction of translation patterns in the learning phase of systems that conform more to the two-stage translation process given in Figure 4. The extraction of translation patterns represents, to a certain extent, the extraction of a rule base which is then used as the primary bilingual knowledge source, as in traditional RBMT. The approaches in section 2.1.2 most closely represent the idea of using structural transfer rules and hence require linguistic knowledge about the language pairs in question. The systems in 2.1.3 are of interest since they extract translation patterns by generalising sentence pairs and very closely resemble the approach presented in this thesis. Those in section 2.1.3.1 are of particular interest since they generalise sentence pairs by language-neutral methods. The strengths and weaknesses of such approaches are discussed. Section 2.2 describes the array of text alignment techniques in

the literature that are of relevance to the alignment procedure utilised in this approach when aligning the text fragments (and variables) of which the translation patterns are composed. Furthermore, relevant recombination techniques are discussed in section 2.3. This chapter concludes with a summary of the strengths and weaknesses of existing approaches to extracting and recombining translation patterns and makes recommendations for future implementations such as the one presented in this thesis.

2.1.1 Linear EBMT

Traditional linear or non-structural EBMT systems that do not extract a rule base nor model themselves on transfer-based systems typically extract TL equivalents of overlapping partial exact matches of the SL input dynamically and recombine them in an appropriate manner to produce TL translations. There is frequently no or very little pre-processing of the corpus. Consequently, matching takes place against raw translation examples rather than against abstract representations of them. Furthermore, the majority of, but by no means all, bilingual relationships are computed at run-time. This compares with rule or pattern-based systems which compute all bilingual knowledge in a pre-matching extraction phase. Moreover, recombination represents more of a challenge as there are no sentential patterns of translation or translation rules to determine the order of items in the TL.

One such case is the PanEBMT system (Brown, 1996; Frederking & Brown, 1996) which constitutes the example-based component in the Pangloss multi-engine MT system (Frederking et al. 1994). In this system, all possible segmentations (sub-strings) or chunks of the SL input are found. This differs from the previous approach of Nirenburg et al. (1993) who attempt to compute the optimal partitioning or cover of the SL input. Subsentential alignment is performed at run time for each translation example in which the SL chunks are found in order to determine their TL equivalents.

No pre-processing of the corpus is necessary as matching of the SL input takes place against raw translation examples. However, a minor and optional knowledge source is included in the form of equivalence classes equating words within classes such as weekdays, closed-class words and words which may be inserted. As a pre-stage, the translation examples are ‘tokenised’ so that words in any of the equivalence classes are

indexed under the equivalence class name in order to increase the likelihood of matching an SL input. Later enhancements include the automatic extraction of the bilingual dictionary from the corpus (Brown, 1997) with the result that the lexicon is more 'tuned' to the corpus and portability to new language pairs or text types is increased.

A similar approach is the MEG system (Somers et al. 1994). This approach is claimed to be a 'pure' EBMT system in that no external linguistic knowledge, no matter how minimal, is used. Only information gleaned from the corpus itself is used. However, in contrast with the PanEBMT system, there is more pre-processing of the corpus before matching takes place. First, the corpus is POS tagged, although this is undertaken using a tag-set derived entirely from the corpus (Schütze, 1993) to maintain maximum portability. Subsequently, word-level alignment is carried out. Matching an SL input against the corpus is, however, carried out at run-time. The tagged SL input is matched against each relevant SL sentence in the corpus to produce a possibly non-contiguous fragment which the two sentences have in common. Strong (word and tag) or weak (tag only) correspondences are computed. By a similar method, the TL equivalent of each SL fragment is computed. The assumption is that, given a set of SL sentences in which an SL chunk is found, the corresponding TL chunk will be the sequence that is common to all of the corresponding TL sentences.

Another system that pre-processes the corpus is that of Andriamanankasina et al. (1999). The French-Japanese corpus they use is POS tagged. Alignments at the word level are computed by statistical means. This already represents a significant amount of bilingual knowledge, but does not include information about sentential patterns of translation. The translation process is based on a process of dynamic recursive division of the SL input. A translation example is retrieved whose SL sentence closely matches the SL input. The links between the words in the translation example predict the best point – the common segment for both sentences – at which to divide the SL input, thus providing the first TL word in the translation. The words to the left and right of this segmentation point are translated by the same process of recursive division, until there are no more words in the SL input to translate.

2.1.2 Structural EBMT Systems

Some of the first approaches to EBMT involved storing translation examples as fully annotated tree structures with alignments at the lexical and structural level. These aligned tree structures served as the rule base against which parsed SL input sentences were matched. Typically, the closest matching SL structure to the parsed SL input is retrieved. The alignments at the lexical and structural level between translation examples enable the retrieval of translations of segments of the SL input from other translation examples in the corpus. The corresponding TL tree is then constructed from these fragments. The TL sentence is subsequently generated.

Automatic extraction of annotated translation examples provides a convenient way of constructing a rule base – a process which is error prone when manually created by linguists. The similarity with traditional rule-based or transfer systems has led to such approaches being described as 'example-based transfer' in the MT literature, since transfer is carried out on the basis of examples rather than rules *per se*. Algorithms traditionally employed by transfer systems i.e. generating sentences from syntactic/semantic representations, are sometimes found in such systems.

Matching against a set of tree structures is a more complex task than matching against a set of raw translation examples and involves a considerable computational cost (Noetzel & Selkow, 1983; Watanabe, 1992; Maruyama & Watanabe, 1992; Zhang & Shasha, 1997). This can be alleviated, to some extent, by the use of massively parallel processors (Kitano & Higuchi, 1991a, b; Sumita et al., 1993). EBMT based on the correspondence of tree structures also requires a significant amount of external linguistic knowledge in the form of parsers and perhaps bilingual lexicons. This detracts from portability. However, the more linguistic information that a system is given, in theory, the more accurate its translations. One advantage of including structural information in translation examples is the ability to represent explicitly alignments between languages that indicate a structural divergence, e.g. head-switching.

The MBT2 bi-directional English-Japanese EBMT system of Sato and Nagao (1990) is a typical 'example-based transfer' system. Translation examples are represented as pairs of SL and TL word-dependency trees with explicit links between the sub-trees (including the leaf nodes which correspond to lexical units). Sub-trees are linked so that

example fragments or sub-trees from more than one translation example or word-dependency tree pair can be used to retrieve the translations of matched portions of the SL input. These linked sub-trees are then termed translation rules and are represented in Prolog notation. Given the translation example in (1), the translation rule formed is represented diagrammatically as Figure 5 (Sato, 1995).

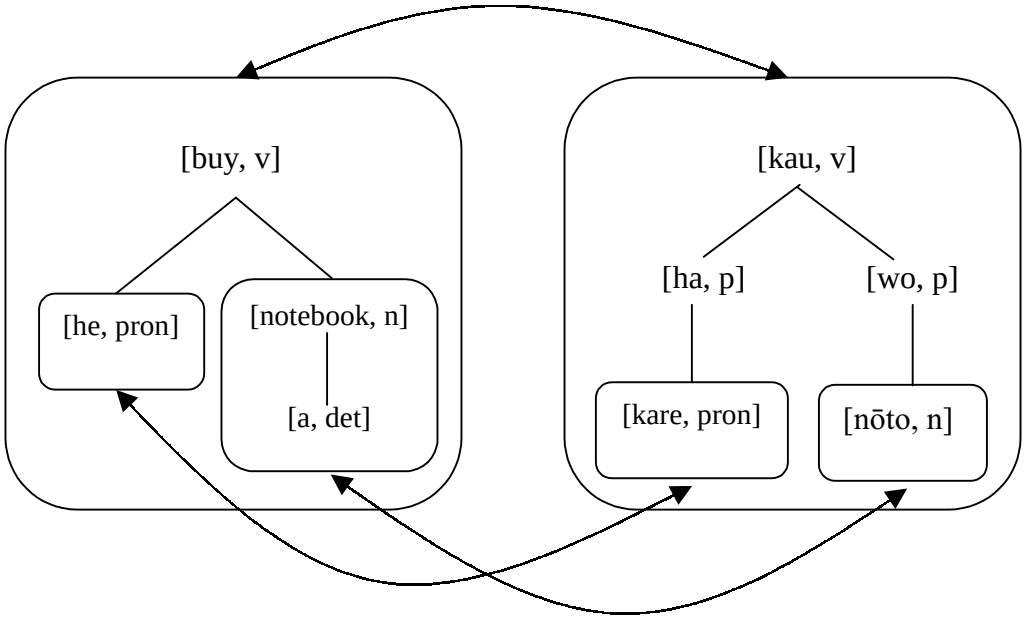

Figure 5: Example Entry in Sato (1995: 36)

(1) He buys a notebook

Kare ha nōto wo kau

Similar representations of linked annotated tree-structures are found in Sadler (1991), Matsumoto et al. (1993), Sobashima et al. (1994), Sato (1995), Matsumoto & Kitamura (1997), Meyers et al. (1998) and Watanabe (1992, 1993, 1994, 1995). The translation patterns in the transfer systems of Takeda (1996a, b) and Watanabe & Takeda (1998) are also similar. They are made up of pairs of SL and TL Context Free Grammars with zero or more syntactic head and link constraints for non-terminal symbols. Although

they are manually created, they suggest a method by which example-based processing can assist the production of new translation patterns.

There is a variety of differences in terms of the complexity of the data structures used to represent dependency structures and the correspondences between them. At one end of the scale, the dependency structures and the correspondences between them in Sato & Nagao (1990) are clear and easy to understand – an important point for error analysis. At the other end of the scale, the mappings between dependency structures in Watanabe (1992) are complex and opaque. In their scheme, a translation rule *r* is a triple (2) consisting of a matching graph G_m, a construction graph G_c, from which the translation is constructed, and a set of mappings between them *M*. The graphs are expressed as rooted labelled directed acyclic graphs. Two types of mappings between nodes in G_m and G_c are defined (upwards or downwards) (3) depending on whether the translation node governs or is governed by the SL node. (3) is updated in Watanabe (1993) to include lexical mappings M_L between the graphs as a distinct set of correspondences from the structural mappings. A further extension is made by Watanabe (1993) where incorrect translations produced by the normal structure-preserving technique are compared with the correct translation, so that a new translation rule is produced.

(2) $r = < G_m, M, G_c >$

(3) $M = < M\uparrow, M\downarrow >$

The dependency structures in the system of Al-Adhaileh & Kong (1998, 1999) are represented with links at the structural and lexical level expressed in terms of indices. They give examples between English and Malay. The nodes in the trees are indexed to show the lexical head and the span of the tree of which that item is the head. For example, given the sentence: *He picks the ball up,* the node labelled: "ball(1)[n](3-4/2-4)" indicates that the sub-tree headed by *ball*, which is the word spanning nodes 3 to 4 (the fourth word) is the head of the sub-tree spanning nodes 2 to 4 i.e. *the ball*. Links between structural and lexical units are also given in terms of indices.

The key part of the approach of Grishman (1994) and Meyers et al. (1996, 1998) is the fact that only dominance-preserving alignments are allowed when computing the correspondences between the sub-trees. For example, if node *a* dominates node *b* in the SL tree, then *f(a)* dominates *f(b)* where *f* is the 1:1 mapping function between source and target nodes. Alignments are computed by dynamic programming (DP) where each node in the SL tree is matched with each node in the TL tree. The authors report that eliminating dominance violations reduces the search space considerably. Once the sub-structures in the SL and TL trees have been aligned, they are termed transfer rules.

Planas & Furuse (1999) represent examples as a multi-level lattice where it is possible, depending on the amount of linguistic information available, to combine typographic, orthographic, lexical, syntactic and other information. Their proposal is designed to represent translation examples in a TM system and improve the matching mechanism to take place on any, if not all, of the levels specified. However, their approach is also amenable to EBMT. Multi-level information is also included in the approach of Zhao & Tsujii (1999). They propose a multi-dimensional feature graph with information about speech acts, semantic roles, syntactic categories and functions, etc.

The approach taken by Poutsma (1998) and Way (1999) is loosely contained within the category of structural EBMT. The source text is parsed using Bod's (1992) DOP (data-oriented parsing) technique, which is itself a kind of example-based approach. Matching sub-trees are then combined in a compositional manner.

2.1.3 Translation Pattern Extraction

It is possible to build a rule base by methods other than parsing and storing translation examples as fully annotated tree structures. The dependence on reliable parsers for both languages in the corpus reduces the portability of systems such as those outlined in section 2.1.2. Therefore, alternatives to extracting a rule base were investigated. This led to a new strain of EBMT systems that developed techniques to extract generalised or variablised sentence translations from translation examples. These were often termed 'translation patterns', 'templates' or even 'rules' and typically contained less structural information than example-based transfer systems. It is against this set of translation

patterns that SL inputs are matched. Such systems typically conform to the two-phase translation methodology depicted in Figure 4.

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. The generalisations are marked by variables. One way of classifying approaches is by the methods used to generalise translation examples and what constraints, if any, apply as a result. The following section shows how there is wide variation in the amount of external linguistic knowledge required when generalising sentence pairs. At one end of the scale significant resources, such as parsers and bilingual dictionaries are required. At the other, language-neutral principles, such as those borrowed from analogical reasoning (Skousen, 1989), are used.

An approach that makes use of significant linguistic resources to generalise sentence pairs is that of Kaji et al. (1992). Given a corpus of English sentences and their Japanese translations, they use parsers for both English and Japanese and an English-Japanese bilingual dictionary to find correspondences between sentences at the phrase-structure level (Figure 6). These phrases are replaced by variables to produce translation patterns with variables that contain syntactic and possibly semantic constraints.

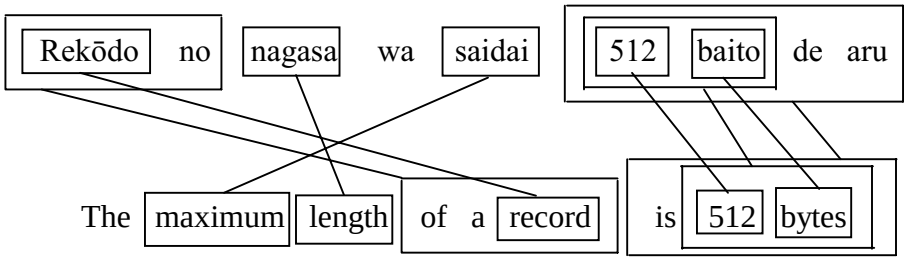

Figure 6: Translation Template (Kaji et al. 1992: 673)

Their method proceeds as follows: the sentences in the corpus are analysed and then individual words that are possible translations of each other are coupled by means of the bilingual dictionary. The coupling of words is done in order to obtain candidates for variables in the translation patterns; Coupling is restricted to content words. Next, coupling of phrases is carried out. Translation ambiguity between coupled words and

phrases needs to be resolved as one word and indeed one phrase may have more than one possible counterpart in the other language. Both ambiguities are resolved implicitly in the phrase coupling stage on the basis of word translations included in each potential phrasal alignment. The length of the phrases also influences alignment in that longer phrases are the preferred translations of longer phrases and similarly shorter phrases tend to be translated by shorter phrases.

The phrases that have been coupled are replaced by variables and include information as to the syntactic category of the replaced phrase. The resulting translation pattern may contain any subset of the set of coupled phrases. Translation patterns (4) and (5) are translation patterns formed from the set of coupled phrases between the English and Japanese sentences depicted in Figure 6. Further refinement of the translation patterns is carried out by semantic marking in the case of conflicting translation patterns, e.g. between 'play the piano' and 'play baseball': *play X[NP/sport]* and *play X[NP/instrument]*. The aim of the constraints in (4) and (5) is to ensure well-formed translations. However, the required knowledge sources detract from the portability of the system – portability being one of the advantages of EBMT.

(4) *X[NP]* no nagasa wa saidai 512 baito de aru ←→
The maximum length of *X[NP]* is 512 bytes

(5) *X[NP]* no nagasa wa saidai *Y[N]* baito de aru ←→
The maximum length of *X[NP]* is *Y[N]* bytes

The translation patterns, although involving structure-preserving relationships, primarily involve the direct transfer of strings. As a consequence, they are clear and easy to understand, unlike the more opaque examples that are tree-to-tree relationships (see section 2.1.2). A similar idea is presented in Kinoshita et al. (1994), where translation patterns consist of sequences of constants and variables e.g. *"use of $1 reduces $2"*. The variables are marked for syntactic category, e.g. verb or noun phrase, and number. The authors do not explain how the patterns are extracted. Interestingly, 1:0 and 0:1 relationships exist between the SL and TL variables enabling the translation of structures involving omission or insertion.

As Kitamura & Matsumoto (1995) demonstrate, it is possible to generalise sentence-pairs with semantic as well as grammatical information. The generation of translation rules centres on individual words. Words are chosen as sources for the generalisation of translation examples. Once a word is selected, it is found within the translation examples that contain it. These are then parsed with "LFG-like grammars" into dependency structures. From these, matching sub-graphs are extracted. If the graphs consist of just a single word, then a word-level rule is generated. Otherwise, it is regarded as a phrasal expression. The elements that differ are generalised according to similarities as defined by a thesaurus. The authors state that the quality of the translation rules is dependent on the quality of the thesaurus used and that their method is most effective when used with non-idiomatic text. They also state that their method is restricted to active-declarative sentences.

The approach of Nomiyama (1992) is similar in that semantic information is used. He describes how translation examples are generalised into rules using a thesaurus by replacing words by more general concepts in the thesaurus. As an example, when translating the sentence *the dog barks*, the word *dog* can be replaced by the concept <dog> or even the more general concept <animal> in order to match with the set of translation patterns which may not contain or have been generalised to include the concept <dog>. The frequencies of the words replaced and the distance between the replaced word and the concept in the hierarchical thesaurus (semantic similarity) are used to assign scores or weights to the translation patterns. Almuallim et al. (1994) and Akiba et al. (1995) report on a similar approach, but in using algorithms borrowed directly from the ML field, they are more formal in the description of their implementations. However, the translation examples from which they induce the transfer rules are largely hand-constructed. The 'pattern-based' approach to MT of Watanabe & Takeda (1998) is similar in a certain respect. Although their approach is essentially a variant of the traditional structural rule-based approach, they propose extending the rule set by incorporating more specific rules (reducing the number of variables) and by treating existing rules as if they were examples. In this way, a rule used to translate a phrase such as *take a bus* can be used to translate a phrase such as *take a taxi*, where *taxi* and *bus* are semantically similar enough to be interchangeable.

The extraction of translation templates in the approach of Carl (1999) is also dependent on the principle of generalisation of translation examples by the replacement of mutually interchangeable SL and TL items. For instance, given the translation examples in (6) and (7), the translation template (8) is derived. The approach makes use of a rich morphological analysis and a syntactic rule formalism to enable shallow parsing of the German and English translation examples. This means that the interchangeable items in the induction process (*m* and *n* in the example) must be equivalent morpho-syntactically. The syntactic rule formalism employed enables the percolation of feature bundles in the derivation trees so that the variables contain the morpho-syntactic constraints necessary to ensure well-formed translations. Like the approach of Kaji et al. (1992) more than one generalisation of a translation template can be induced from a pair of translation examples, depending on the number of mutually interchangeable items. The induction method is, however, limited to comparing structurally similar languages.

(6) $(m)_{\alpha} \leftrightarrow (n)_{\alpha}$

(7) $(a\ m)_{\beta} \leftrightarrow (b\ n)_{\beta}$

(8) $a\ X_{\alpha} \leftrightarrow b\ X_{\alpha}$

Furuse & Iida (1992a, b) describe a system where the differing degree of effort in the transfer process can be matched by either one of three types of generalised example pairs. They state that the translation of simple SL expressions need not involve syntactic and semantic analysis as with longer more complex sentences, but rather string transferral may suffice. The first type of examples are stored as string literals – with effectively no generalisation (9). The second consists of "pattern-examples" where words are replaced by variables (10). Thirdly, grammar-level transfer rules are expressed as context-sensitive rewrite rules, using sets of concrete instances of each category (11).

When there is more than one way to apply this kind of transfer knowledge in the translation process, i.e. when one SL expression of any of the three types corresponds to more than one TL expression, then the most similar expression is chosen on the basis of distance in a hierarchical thesaurus. For example, when translating the sentence: *jinjika o onegai shimasu* (*jinjika* = 'personnel section'), the TL expression selected is 'may I speak

to the personnel section', since *jinjika* and *jimukyoku* are more semantically similar than *jinjika* and *bangō*.

(9) *Sochira ni okeru* ←→ We will send it to you

Sochira wa jimukyoku desu ←→ This is the office

(10) *X o onegai shimasu* ←→ may I speak to the X

(X = jimukyoku / office, …)

X o onegai shimasu ←→ please give me the X

(X = bangō / number, …)

(11) N1 N2 N3 ←→ the N3 of the N1

(N1 = kaigi / meeting, N2 = kaisai / opening, N3 = kikan / time)

N1 N2 N3 ←→ N2 N3 for N1

(N1 = sanka / participation, N2 = mōshikomi / application, N3 = yōshi / form)

The scale of generalisation of the translation patterns in this approach and the level of linguistic knowledge necessary for it is of interest at this point. The first type do not contain any generalisations and thus require direct string matches with an SL expression. They are thus analogous to a corpus of raw translation examples. The third type involves grammatical knowledge about the SL and TL and are effectively transfer rules of the 'traditional' kind representing structural correspondences between SL and TL expressions. The second type are somewhere between the two, where a generalisation has been made on the basis of semantically similar interchangeable SL and TL items. The generalisation is represented by the variables. Of the three, this second type most closely resembles the translation patterns extracted in the approach taken in this thesis. However, the authors do not explain how such generalisations are carried out. A similar idea is found in Kitano & Higuchi (1991a, b) who distinguish between "specific" cases and "generalised cases" with a "unification-grammar" in place for anything not covered by either of these.

Generalisation of translation examples is also a technique employed by Brown (1999). The underlying system is one based on partial overlapping exact matches (Brown,

1996), where the translations of the partial fragments, which require a bilingual dictionary to compute, are made available as candidates for use in a multi-engine MT system (Frederking et al. 1994). However, the addition of generalisation techniques is shown to reduce significantly the amount of pre-translated text required, or conversely, with a certain amount of text, a higher rate of recall is achieved.

Brown adds a "modest" amount of linguistic information in the form of equivalence classes. Certain words can be used interchangeably, forming an equivalence class denoting numbers, weekdays, country names etc. Substituting any other member of the class produces a well formed sentence. Words or phrases in the translation examples are replaced by the name of the equivalence class, thereby making the examples more general. This makes the approach similar to that of Nomiyama (1992), except that no thesaurus is used. The words or phrases contained in an SL input are generalised in the same way so that matching can now take place with the generalised example set. The example given by Brown is reproduced as follows: the sentence (12) can be generalised to become (13) after an initial generalisation, and further generalised after a second pass to become (14). (14) can now match (15) among other possibilities.

(12) John Miller flew to Frankfurt on December 3rd.

(13) <first-name> <last-name> flew to <city> on <month> <ordinal>.

(14) <person-m> flew to <city> on <date>.

(15) Dr. Howard Johnson flew to Ithaca on 7th April 1997.

A further generalisation is made possible by the addition of more conventional linguistic knowledge, such as number and gender. Tokens are tagged with the appropriate number/gender information and are subsequently matched recursively against a set of grammatical rewrite rules in a shallow parsing process until no further combinations can take place. Since the tags contain linguistic information, they enforce linguistic constraints and enable the selection of the correct sense and translation of a word.

Brown reports that adding equivalence classes and recursive matching reduces the amount of translated text for Spanish-English and French-English EBMT by a factor of

six. This is important for low-density languages for which parallel text corpora are scarce, enabling high coverage with the minimum of text. Building bilingual resources is frequently an expensive and time-consuming task. However, lower translation quality is reported when using the grammar rules due to over-generalisation since the exceptions to the rules yield incorrect translations. The cost of creation of the equivalence classes and grammar rules is reported to be approximately 2 person weeks by a *non-linguist*. However, if the morphological information necessary for the linguistic annotation phase is not available electronically in pre-existing sources, it requires a considerable amount of investment time. Brown (2000) shows how it is possible to create automatically the set of equivalence classes by employing clustering techniques, thus reducing the amount of labour involved.

2.1.3.1 Language-Neutral Generalisation Techniques

Approaches that employ techniques to generalise translation examples requiring significant linguistic resources are subject to certain constraints. First, the addition of external linguistic tools reduces portability – a major disadvantage for EBMT. Second, for lesser studied languages, the resources may not exist and their creation may be time-consuming or expensive. Finally, the error rate incurred from hand-crafted or automatically produced linguistic tools is passed on the to the EBMT system itself. Therefore 'pure' EBMT systems that operate by using raw translation examples and language-neutral algorithms that eschew external knowledge sources, such as bilingual dictionaries, parsers, taggers, are advantageous in that they are highly portable to new text types, domains and language pairs (although they have limitations of their own). The following section outlines some of the existing approaches that (broadly) conform to these specifications. In some cases, minimal linguistic knowledge is involved, i.e. stop lists of closed-class words.

An approach to translation pattern extraction by the Department of Computer Engineering and Information Sciences at the University of Bilkent, Ankara, Turkey, is based on analogical reasoning between pairs of translation examples in a sentence-aligned bilingual corpus. They have published their method extensively (Güvenir & Cicekli, 1996a, b, 1998; Cicekli & Güvenir, 1996, 2001; Cicekli, 2000). Their previous

attempts at extracting translation patterns involved the correlation of syntactic structures between English and Turkish (Güvenir & Tunç, 1996) by a similar method. However, the authors reported that they could not find reliable parsers for both languages. This led to the development of language-neutral techniques for extracting correspondences, or translation templates as they term them, between languages by analogical methods.

Their method is based on the following observation: given two sentence-pairs in a bilingual corpus, the orthographically similar parts of the two SL sentences correspond to the orthographically similar parts of the two TL sentences. Similarly, the different parts of the two SL sentences correspond to the different parts of the two TL sentences. The differences are replaced by variables to produce generalised translation examples. As an example the two sentence pairs in (16) can be generalised to produce the translation pattern (17). The underlined text denotes the similar parts of the two sentence pairs.

(16) I gave the ticket to Mary ←→ Mary'e bileti verdim
I gave the pen to Mary ←→ Mary'e kalemi verdim

(17) I gave the X_s to Mary ←→ Mary'e X_t+i verdim

Note that due to the correspondence of the variables X_s and X_t in the translation pattern, the bilingual relationship between the items *ticket* and *pen* is obtained. In this way, subsentential bilingual knowledge is inferred (see section 2.2).

For highly inflected languages or worse, agglutinative languages such as Turkish, the ability to generalise sentence pairs correctly is reduced. Therefore, sentences are morphologically analysed so that words are represented at the lexical level, that is, surface forms are divided into their stems and morphemes. This permits the generalisation of sentence pairs such as (18), where no sensible generalisation is possible, by representing them in their lexical form (19). The + operator indicates a morpheme boundary while *H* is a morpho-phonemic representation capturing the Turkish system of vowel harmony.

(18) I am coming ←→ *geliyorum*
I am going ←→ *gidiyorum*

(19) I am come+ing ←→ *gel+Hyor+yHm*
I am go+ing ←→ *gid+Hyor+yHm*

A similar method is employed by Malavazos & Piperidis (2000) and Malavazos et al. (2000). They base their algorithm more firmly within the analogical reasoning framework (Skousen, 1989), although their method of generalising translation examples is analogous to that employed by Güvenir & Cicekli above. However, they arrive at their method by a slightly different observation: "given any SL and TL sentence pair, any alteration of the SL sentence will most likely result in one or more changes in the respective TL sentence, while it is also highly likely that constant and variable units of the SL sentence correspond to constant and variable units in the respective TL sentence." (Malavazos & Piperidis, 2000: 517) As a result, generalised sentence pairs or translation rules are abstracted which consist of sequences of string literals interspersed with variable elements.

In the analogical reasoning framework, the translation rules are depicted as the supracontexts within the set of translation examples and provide sentential patterns of translation, whereas the subcontexts comprise the translation relations between the words or phrasal elements that have been replaced by variables, forming a bilingual lexicon. The results of the analogical process by which sentence pairs are generalised and variable units linked is stored in an analogical network (Federici & Pirrelli, 1994).

The similarity of the algorithm to that of Güvenir & Cicekli is evident by the example the authors provide: the English-Greek sentence pairs in (20) are generalised to produce the translation template (21). However, no suggestion is made as to how the inflected nature of Greek is to be handled.

(20) **Customizing** application **settings** ←→ **Προσαρμογή ρυθμίσεων** εφαρμογής
Customizing network **settings** ←→ **Προσαρμογή ρυθμίσεων** δικτύου

(21) Customizing X_s settings ←→ Προσαρμογή ρυθμίσεων X_t

The approach of Echizen-ya et al. (1996, 2000) is also one that generalises sentence pairs by means of what they term “inductive learning with genetic algorithms”. Generalisations of sentence pairs are carried out in a manner analogous to the approach of Malavazos et al. and Güvenir & Cicekli: similar parts between two sentence pairs are equated, as are the differences. The differences are replaced by variables to generalise the sentence pair. This process is termed “induction”. As an example, the sentence pair in (22) is generalised to produce the translation pattern (23). The underlined text denotes the similarities between the two sentence pairs.

(22) He like tennis ←→ Kare wa tenisu ga suki desu
He likes tea ←→ Kare wa ocha ga suki desu

(23) He likes X_s ←→ Kare wa X_t ga suki desu

A distinction is made between the *sentence translation rules*, such as (23) and *part translation rules* which identify translation relations between words or phrases, such as tennis-*tenisu* and tea-*ocha* (X_s and X_t). Furthermore, more translation rules are generated from the translation rules extracted by recursively applying the same induction process to the translation rules extracted. This ability to create more refined translation rules adds to the flexibility of the system.

The above section shows how translation patterns or templates are formed by the generalisation of translation examples. However, this is not the only method by which they are created. A template-based EBMT system called *Gaijin* (Veale & Way, 1997) fits loosely into the above categorisation schema of EBMT systems that operate by extracting translation patterns to be used as a rule base. The translation templates extracted represent mappings between SL and TL chunks for each sentence pair in a bilingual corpus. A single translation template provides the sentential context for the translation of a given SL input, but the translation is performed by using aligned SL and TL chunks from other translation templates. At this point, the *Gaijin* system marks a departure from the approaches described above: when the aligned chunks within the templates do not match the SL input exactly, it is possible to *adapt* the example chunks to match the SL input exactly. Minor differences between an SL chunk in a translation template and a chunk of

the SL input can be rectified by adaptation. Any changes made to the SL example chunk are reflected in the TL chunk. Templates and chunks are retrieved in the matching process based on the level of ease of adaptability to the SL input. This places this approach within the adaptation-guided retrieval paradigm alongside approaches such as those of Collins & Cunningham (1995, 1997) and Collins et al. (1997).

In the *Gaijin* system, translation templates serve as mappings between the phrasal constituents of a given pair of SL and TL sentences. Therefore, a segmentation algorithm is required to chunk the sentences. The algorithm employed is based on the psycholinguistic marker hypothesis principle (Green, 1979). This principle states that all natural languages are marked for grammar at the string level by a closed set of specific morphemes and lexemes. This has previously been exploited for segmental purposes in MT by Juola (1994, 1997). An MT system can achieve a basic phrase-segmentation of an SL input by exploiting a closed list of known marker words that signal the beginning or end of each segment. The *Gaijin* system employs a marker set that includes among others closed-class words such as *in, out, on, with, the* for English. Each segment is identified with the category of its leading marker word, i.e. *Prep* for preposition, *Det* for determiner etc. Chunks identified in this manner are subsequently aligned to produce complete translation templates. The resulting templates are stored in a Prolog style format. The template extracted from translation example (24) is given in Figure 7.

(24) Displays controls for colouring the extruded surfaces
Durch Klicken auf dieses Symbol lassen sich Optionen zum Kolorieren der extrudierten Flaechen anzeigen

The translation templates in the *Gaijin* system do not contain variables as in the approaches above, but are made up of mappings between SL and TL chunks. SL input sentences are matched against these chunks and the corresponding TL chunks are retrieved and put in the order specified by alignments within the translation templates to produce an TL string. The amount of linguistic information required in this approach makes it portable to new language pairs with minimal effort as only a set of marker words is required.

```
template(example-14,English,German,

[s (A, _, a14), s (B, prep,b14), s (C, det, c14)],
[durch, klicken, auf, t(A, prep, a14), t(B, prep, b14), t(C, det, c14), anzeigen]).


chunk(english, german, a14, [displays, controls], [dieses, symbol, lassen, sich, optionen]). % A
chunk(english, german, b14, [for, colouring], [zum, kolorieren]). % B
chunk(english, german, c14, [the, extruded, surfaces], [der, extrudierten, flaechen]). % C
```

Figure 7: Translation Template: Veale & Way (1997)

A similar segmentation algorithm is given in Furuse & Iida (1994, 1996) where certain types of function words typically indicate the boundaries between major constituents. Where this is not possible, constituent boundaries are expressed in terms of POS bigrams. However, the translation patterns extracted represent generalised sentence translations in that they consist of constants and variables, e.g. *X at Y, X to Y, I would like to X*, etc.

2.2 Alignment Algorithms

Translation patterns, that is generalisations of sentences that are translations of each other, are typically composed of sequences of text fragments and variables and provide sentential translation paradigms. In order to provide the recombination phase with more flexibility, a more refined bilingual knowledge source containing word or phrasal translations is necessary. One method of producing such a bilingual lexicon or phrasicon is to align the text fragments and variables of which the translation patterns are composed. Another is to extract lexical, terminological, phrasal or even collocational correspondences automatically from the corpus itself (Kaji & Aizono, 1996; Brown, 1997; Turcato, 1998). Existing methods vary as to factors such as the nature and flexibility of the bilingual correspondences computed and the complexity and limitations of the algorithms. This section limits itself to the language-neutral alignment techniques of string-based systems, i.e. those that do not use bilingual dictionaries or other knowledge sources, as does the alignment method presented in this thesis.

2.2.1 Phrasal Alignment for EBMT

In structure-based EBMT systems, where the examples are stored as annotated tree structures, structural knowledge and additional sources such as bilingual dictionaries and thesauri facilitate the alignment of SL and TL nodes (Kaji et al., 1992; Utsuro et al., 1992; Grishman, 1994). This frequently involves considerable computational cost. Matsumoto et al. (1993) attempt to match each SL node with each TL node whereas Meyers et al. (1996) restrict the search to produce alignments that preserve the least common ancestor relationship. This is improved further by Meyers et al. (1998) where only dominance preserving alignments are computed. Alignments are typically limited to the bijective variety.

In the linear PanEBMT system of Brown (1994) and Frederking & Brown (1994), determining the TL equivalents of the SL chunks is carried out by a 'brute-force' mechanism. This is similar to the method of Nirenburg et al. (1994) where the maximum and minimum possible segments of the TL sentence which could possibly correspond to the SL chunk are identified. A score is assigned to each possible sub-string of the maximum segment containing at least the minimum segment. The sub-string with the best score is selected as the TL equivalent of the chunk. The alignment score or bilingual correlation metric is based on a number of simple tests, such as the number of SL words in the chunk with correspondences in the TL chunk. Correspondences are determined by a bilingual dictionary and root/synonym list to handle inflectional variations.

In contrast, the MEG system (Somers et al., 1994) makes use of pre-computed word-level alignments to correlate SL and TL word groups of varying sizes. Weighted *n*-grams (Jones & Alexa, 1994) are used where bilingual lexical distribution, computed by Dice's co-efficient (Dice, 1945), is used to calculate the probabilities of word alignments. This method proves useful for the language pair under study – English and German – where single German word compounds are aligned with multi-word sequences in English. TL equivalents of SL chunks are found by a method analogous to that used to match an SL input against the SL sentences in the corpus. Given the set of SL sentences in which an SL chunk is found, the corresponding TL chunk will be the sequence that is common to all of the corresponding TL sentences.

Systems that extract translation patterns using language-neutral methods typically attempt to provide finer bilingual knowledge by aligning the text fragments and/or variables of which they are composed, again by language-neutral methods. In some systems there is no alignment problem to be solved since translation patterns contain only one generalisation or variable on either side of the pattern and so the alignment is implicit. This is the case for the second type of translation rule (10) described in Furuse & Iida (1992a, b). In this example, the text fragments or constants are not aligned.

In many cases, only bijective alignments between text fragments are computed, where one SL fragment corresponds to one and only one TL fragment. In the translation patterns of Güvenir & Cicekli, where there may be multiple variables and text fragments to align, the correspondences must be inferred by checking previously learned translations patterns. As an example, consider the translation example (25) and the resulting match sequence (26). Without prior information, it cannot be determined whether *I* corresponds to *kitab* or *+m*. However, if it has already been learned in previous translation patterns that *I* corresponds to *+m* and *you* corresponds to *+n*, then the translation pattern (27) is inferred. If a translation pattern has *k* alignments to be made, *k–1* alignments must be resolved from existing translation patterns in order to complete the alignment process of the given translation pattern. This limits the number of match sequences that may be aligned to become translation patterns, since not all match sequences may be resolved. Malavazos & Piperidis (2000) and Malavazos et al. (2000) report a similar method.

(25) I give+p the book ←→ kitab+yH ver+DH+m
you give+p the pen ←→ kalem+yH ver+DH+n

(26) (I, you) give+p the (book, pen) ←→ (kitab, kalem) +yH ver+DH (+m, +n)

(27) X_1 give+p the X_2 ←→ X_2+yH ver+DHX_1

The variables in Echizen-ya et al. (2000) and Carl (1999) are also aligned on a one-to-one basis only, but the alignment method is unclear. The alignment methods appear to be based on how generalisations were extracted in the first place. In the case of

Echizen-ya et al. (2000) the induction mechanism is applied recursively to the translation patterns already extracted to produce further translation patterns with a greater number of generalisations. Carl (1999) reports a similar recursive step.

Not all approaches to aligning variables in translation patterns are limited to a one-to-one mapping. Some compute more flexible alignments that are non-bijective and form bilingual relationships that are *m:n* in nature. For example, the segmentation algorithm in the *Gaijin* system of Veale & Way (1997) results in translation patterns that may be viewed as sequences of SL and TL phrases. The problem is therefore viewed as sequence alignment. The method proposed is similar to that which they use to align the sentences within the corpus. For each translation pattern, each SL segment is compared with each TL segment, employing both segment length and word correspondence weights from a lexicon statistically extracted from the corpus. However, since the segmentation algorithm employed is relatively crude, it is possible that one SL segment will score more highly when aligned with two TL segments or vice versa. In such cases, the two TL segments are merged to form one segment, resulting in a 1:1 relationship after all. This process of segment merging, effectively the opposite of segmentation, is not described. However, in the approach of Kinoshita et al. (1994), bilingual relationships of a 1:0 and 0:1 nature are possible in order to describe omission and insertion in the translation process. However, their method is not described.

2.2.2 Statistical Machine Translation

Non-bijective alignment is possible in the EBMT paradigm of Statistical Machine Translation (SMT). SMT was first proposed by Brown et al. (1988, 1990, 1993) – the research group at IBM who developed the *Candide* system. Their aim was to forego all linguistic knowledge in favour of deriving probabilistic translation models directly from a corpus of sentence-aligned bilingual text. They developed five probabilistic models of translation with increasing sophistication. The later models are generally considered too complex to implement, even by the authors. In practice, only the first two are ever considered.

The first and second translation models of Brown et al. (1993) are word-based translation models where alignments are computed between individual words. An initial

estimate of the alignments is given, after which an iterative process, the Expectation-Maximisation (EM) algorithm of Baum (1972), refines the alignments. Iteration halts at the convergence point where the maximum likelihood solution is reached or the most probable alignments have been computed. All words in a sentence pair in the corpus are connected, if only to a null pointer. The authors carried out their experiments on samples of the English-French Canadian Hansard parliamentary proceedings.

The alignments are flexible in that a word in one language may be aligned with zero, one or more words in another language (fertility). They also account for a range of translation phenomena, such as structural divergences, by being able to compute long-distance dependencies. However, the models are unidirectional in nature as each TL word is connected to exactly one SL word or the null pointer. The number of alignments in a sentence pair is therefore equal to the number of TL words, whereas an SL word may be connected to more than one TL word or none at all. Figure 8 provides an example of word alignments in a French-English sentence pair.

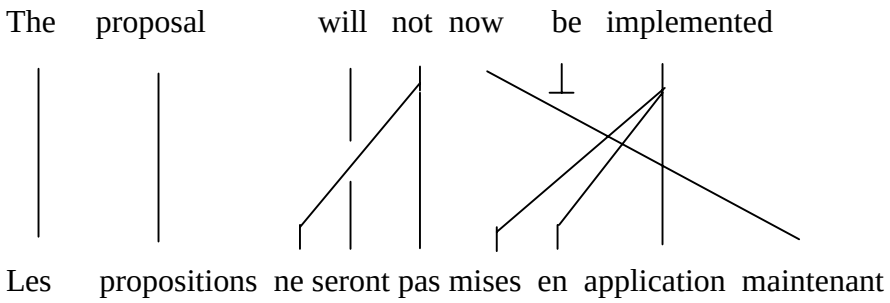

Figure 8: Example of Word Alignments: Brown et al. (1990)

Furthermore, the EM algorithm is computationally expensive and requires large amounts of training data, in the order of approximately 1 million words, to estimate and refine alignment probabilities. The greater number of parameters in the second model – where parallel alignments are favoured – increases the complexity. The complexity increases still further with the subsequent models, until they become unfeasible. It is only in these later models that the natural structuring of words into phrases is considered so that it is possible to compute alignments between phrases.

There have been modifications to the IBM translation models which, to some extent, address the problems outlined above. However, gains in one area are frequently offset by limitations in another. For example, Dagan et al. (1993) reduce the number of parameters in models 1 and 2 so that less training data is required. They also modify the parameter estimation method so that the input corpora need not be aligned and can be noisy. However, high and low frequency words are filtered out and not considered and connections to null are ignored as they are too difficult to model. Furthermore, DP is used to compute word alignments with the limitation that only adjacent alignments are possible. Sato & Nakanishi (1999) also attempt to suppress the processing overhead by rejecting the treatment of low frequency words. However, the resulting models are still word-based and directional.

The approach of Wang & Waibel (1998) extends the first two models of Brown et al. (1993). Phrases are aligned in an initial rough alignment phase, followed by a more accurate alignment of the words contained within those phrases. Phrasal alignment has the effect of speeding up the decoding algorithm. However, phrasal alignments are limited to a one-to-one mapping. Furthermore, the increased number of free parameters, as compared to model 2, increases the complexity of the model. Och & Weber (1998) also base their translation model on model 2. They extend it by modelling phrasal alignments on a one-to-one basis but have the limitation that long-distance dependencies are difficult to model.

2.2.3 Bilingual Vocabulary Alignment Algorithms

There is a wealth of literature on the problem of bilingual alignment, or more specifically, the automatic extraction of lexical correspondences from bilingual corpora. Summaries of word, phrase and term alignment algorithms are given in both Fung & McKeown (1997) and Somers (1998). Such methods can be useful when aligning text fragments in translation patterns since the problems involved are similar. For example, given an SL word or term, the problem involves finding its TL equivalent. However, the problem of aligning text fragments in translation patterns is different to word/term alignment since the sequences of SL and TL items to be aligned may have already been discovered and only their bilingual relationship needs to be computed. The problem of automatic word or

term extraction is more complex in that candidate SL and TL words or terms must be discovered first. The alignment of text fragments in translation patterns is therefore more akin to sequence alignment (Kruskal, 1983), since translation patterns may be viewed as finite sequences of SL and TL items which must be connected somehow.

Language-neutral sentence alignment techniques are, generally speaking, based on the relative lengths of SL and TL sentences (Gale & Church, 1991, 1993; Brown et al. 1991). Later, more robust methods included the use of lexical information (Chen, 1993; Wu, 1994). This information does not have to come in the form of manually provided lexicons, but may include the use of orthographically similar word pairs across languages (cognates) to provide anchor points in the alignment process (Simard et al., 1992). Alignment of text fragments in translation patterns is possible by using similar length-based methods (McTait & Trujillo, 1999), but since the text fragments are much shorter than sentences, there is less data and so the alignment process is more error-prone.

The literature on automatic extraction of bilingual vocabulary is extensive. One method of categorising the approaches is according to the type of lexical unit they extract and align. Some extract single words (Melamed, 1997), terms (van der Eijk, 1993; Daille et al., 1994), phrases (Wu, 1995), noun phrases (Kupiec, 1993), single words and terms (Ahrenberg et al., 1998; Gaussier 1998) and even collocations (Smadja et al., 1996). In these approaches, candidate SL and TL items are extracted and subsequently aligned by statistical scores of correlation. The underlying hypothesis of language-neutral vocabulary alignment techniques is that SL and TL items that are translations of each other will have significantly similar distributions in either text. The strength of association between SL and TL items is proportional to the similarity of their distributions. Correlation may be measured according to a number of information theoretic scores, frequently used in information retrieval (van Rijsbergen, 1979; Frakes & Baeza-Yates, 1992), such as Dice's co-efficient. The DK-vec algorithm of Fung & McKeown (1997) compares recency vectors, representing the distances between lexical items, using dynamic time warping – a technique originally used to compare speech signals.

In the absence of a bilingual lexicon, the advantage of using bilingual lexical distribution is clear. However, it also has many obvious weaknesses. First, a significant

amount of bilingual data is necessary in order to compute the distributions accurately. Second, similar distributions may identify translingual collocates. Third, one-to-many translation equivalence, due to one SL word being translated by two (or more) morphological variants of the TL word poses problems. For example, if an SL word has two different TL translations, the distributions of all three words will be quite different. Finally, the problem of fertility, where a single SL item may translate as a sequence of more than one TL items, is not generally considered apart from in statistical language models (see section 2.2.2). One exception is the alignment of German compound nouns with English word sequences (Jones & Alexa, 1994; Somers et al., 1994) where weighted *n*-grams are used.

A further major limitation of vocabulary alignment programs is their inability to compute non-bijective correspondences. One SL unit is typically connected to one and only one TL unit. This presumes a rather simplistic nature of bilingual phenomena. One exception is that of Smadja et al. (1996). They align continuous and non-continuous English and French collocations. English collocations are extracted from the SL text using a tool called Xtract (Smadja, 1993). Highly associated word pairs are found by identifying frequently occurring words in recurrent positions around a given word. This process is repeated iteratively by finding highly associated words with those word pairs and subsequent triples and *n*-tuples until no further associations can be made. Phrasal templates such as *The average finished the week <NUMBER> points up*, are also possible if the corpus is POS tagged. TL words that are highly correlated with the SL collocation are determined using Dice's co-efficient. The TL collocation is formed by iteratively identifying combinations of the TL words which are highly correlated with the SL collocation.

The key point of this method is the fact that the associated TL words may form a non-continuous equivalent, so in some sense, there is an *n:m* mapping between SL and TL items. For instance, Smadja et al. cite the example: *employment equity* which translates as *équité … matière … emploi*. However, this example serves more as a demonstration of the fact that very high and low frequency words are problematic.[1] In

[1] In the example, the ellipses denote the closed-class words: *en* and *d'*

fact, all approaches that make use of bilingual lexical distribution as a bilingual similarity metric suffer from this problem. Precision of the output is therefore dependent on setting thresholds for the Dice co-efficient. Furthermore, Smadja's approach is dependent on POS taggers to extract phrasal templates and parsers to filter semantically meaningless collocations. Not only does this reduce portability, but it also introduces errors.

2.2.4 Sequence Alignment

If translation patterns are composed of an SL sequence and an TL sequence of text fragments, determining the maximum likelihood alignment between the two sequences requires not just a bilingual correlation metric, but also a sequence comparison algorithm. One frequently used sequence comparison algorithm is the Dynamic Programming (DP) framework. As Kruskal (1983:23) has pointed out, it "has a history of multiple independent discovery and publication", starting with Levenshtein (1966) and Vintsyuk (1968). Its popularity is based on the fact that it is well-documented, efficient and easy to implement. Not only has it been used in familiar tasks such as sentence alignment (Gale & Church, 1991, 1993), word alignment (Dagan et al., 1993) and spelling correction (Wagner & Fischer, 1974), but it has also been used extensively in numerous scientific disciplines such as computer science, information science, pattern recognition, molecular biology and computational mathematics, reflecting similar problems such as the comparison of speech signals and other acoustic data, comparison of genetic sequences such as DNA, image recognition and file comparison.

When comparing sequences, the DP algorithm is able to compute local alignments other than simple one-to-one patterns. For example, in terms of the string-edit problem, Wagner & Fischer (1974) were able to compute substitutions (1:1), insertions (0:1) and deletions (1:0) between two sequences of characters. Lowrance & Wagner (1975) extended the algorithm to include transpositions, i.e. interchanging adjacent or contiguous characters. The algorithm has been extended still further to include compression and expansion, e.g. in the sentence alignment program of Gale & Church (1991, 1993), two SL sentences may be aligned with one TL sentence (compression) or vice versa (expansion). Figure 9 depicts these alignments types graphically between sequence x and sequence y. The alignments x_1:y_1 and x_3:y_2 depict substitution, x_2 is

omitted, y_3 is inserted, there is transposition between x_4, x_5 and y_4, y_5 and x_6 and x_7 are compressed to align with y_6 and conversely, x_8 is expanded to align with y_7 and y_8. Furthermore, the DP algorithm is advantageous in that it runs within an acceptable asymptotic running time. Its worst case time and space complexity is polynomial and of the order $\mathcal{O}(mn)$, where m and n represent the lengths of the two sequences.

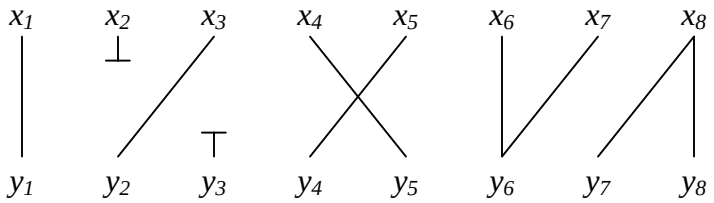

Figure 9: Local Alignment Patterns

One major disadvantage of the DP algorithm is its inherent inability to compute non-adjacent alignments. While DP is suitable for the task of sentence alignment, in that the order of sentences in two languages are highly similar, a sequence comparison algorithm for subsentential fragment alignment must take into consideration the fact that the order of words or fragments may not be similar. When translation patterns are extracted between two structurally divergent languages or languages with differing word orders, the DP algorithm may not successfully compute the correct alignments. A fairly thorough search of the literature has so far failed to find any description of the DP algorithm extended to cover this particular case. Kukich's (1992) survey suggests that Lowrance & Wagner (1975) "accounted for additional transformations (such as exchange of non-adjacent letters)" (p.394). Indeed, some of their illustrations would lead one to believe that this is the case. However, the text quite explicitly refers to "interchange [of] any two *adjacent* characters" (p.177, emphasis added), and an implementation of the algorithm confirmed the restriction. Oommen & Loke (1997) account for simultaneous transpositions and substitutions, but still only when the transposition is of *adjacent* characters. It would appear that the very nature of the DP equation (Figure 10) means that the algorithm could not easily be adapted to allow for non-adjacent alignments, since there is a very simple relationship between the terms of the two functions *D* and *d* in the equation which requires them to have neighbouring values (*i* and *i–1* and *i–2* etc). In

Figure 9, if x_1 and y_8 were translations, the DP equation would fail to capture that relationship. An exhaustive search algorithm is able to find the best set of alignments, both adjacent and non-adjacent, but is prone to a combinatorial explosion in the search space, making it impractical even for very short sequences.

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

Figure 10: DP Equation

Nevertheless, the DP equation has been applied to the subsentential alignment task, as in Boutsis & Piperidis (1998a, b). In this approach, an English-Greek sentence aligned corpus is POS-tagged. Surface syntactic analysis is applied on the basis of regular grammars. This shallow parsing results in the recognition of SL and TL clauses. Next a probabilistic model of alignment is applied which operates on the basis of previously computed word translation probabilities and character lengths of the clauses. Possible alignments are fed into a DP framework in order to compute the most probable alignment of clauses. Where the two sequences are very long, the optimal alignment is approximated by simulated annealing (SM) (Kirkpatrick et al., 1983). Although SM does not guarantee the optimal solution, it arrives at a good approximation in non-exponential time. Due to the fact that DP is used, alignments of the type 1:1, 1:0, 0:1, 1:2 and 2:1 are computed. In the absence of hand-aligned training data, clause alignment parameters (initial probabilities of a given alignment type) are estimated. These probabilities are refined using the EM algorithm. The disadvantage of such a method is again the large amounts of training data required by the EM algorithm. Moreover, the authors acknowledge the fact that, unlike sentence alignment, the order of clauses in the SL sentence is not necessarily the same as the order of clauses in the TL sentence.

The problem of long-distance dependencies is alleviated somewhat in a tree-based, as opposed to string-based, approach to alignment. Alshawi et al. (2000) learn dependency translation models as collections of finite state transducers in order to make long distance dependencies more local. This would enable their DP alignment phase to allow for, what are in actual fact, non-local alignments.

2.3 Recombination

Once an SL input has been segmented and the TL equivalents of those segments have been determined, the final task is to recombine the TL text fragments in an appropriate manner to form a translation string. Despite its obvious importance, recombination is an area that has received relatively little attention in the literature and is frequently limited to a small section or paragraph, if that, tacked onto the end of a research paper. Most attention is paid to the segmentation, alignment and matching phases. Furthermore, recombination techniques are often specific to individual approaches to EBMT.

2.3.1 Structure-based EBMT

In the case of structure-based EBMT, where the examples are stored as annotated tree-structures and the correspondences between the fragments are explicitly labelled, recombination is trivial. In fact, well established techniques in computational linguistics are applied. For example, in Sato (1995), the recombination stage is akin to tree unification. Once the TL tree has been constructed, the surface representation is produced by familiar generation algorithms that start with a tree structure and output a surface string. Watanabe (1992, 1995) adapts a process called “gluing” from Graph Grammars, which is a flexible kind of graph unification. Al-Adhaileh & Kong (1999) state that the process is “analogous to top-down parsing” (p.249).

2.3.2 Linear Recombination

On the other end of the scale, systems that produce a set of TL fragments with no knowledge about the ordering of items in the TL are more problematic. One frequently used method to solve this problem is to build a statistical model of the TL. Statistical systems, such as those of Brown et al. (1991, 1993) build bigram models of the TL so that the TL translation string that best fits this model, i.e. the one that has the highest

probability of being a well formed string in the TL, is selected as the translation solution. The PanEBMT (Brown, 1996) system works in a similar manner. TL equivalents of segments of the SL input are placed on a chart along with the partial translations of the SL input from the two remaining engines in the Pangloss system (Frederking et al., 1994). The final translation is produced by determining the best path through the chart according to a statistical model of the TL (Brown & Frederking, 1995).

An alternative method is given in Somers et al. (1994). Given the set of TL chunks, translations are produced by considering all possible combinations of those chunks. Translations are ranked according to a score assigned to each translation based on the fact that the chunks have come from real language data, i.e. the corpus, and that the environment in which they were found gives some clue as to their potential future environment. Each TL chunk is augmented with 'hooks', i.e. the *n* previous or subsequent sequence of POS tags of the chunk, weighted according to frequency of occurrence in the corpus. The hooks enable the chunks to be connected and the credibility of the resulting TL string assessed. The frequency scores are multiplied as the chunks are combined to produce the final score. An additional method of allowing the monolingual user to choose the most probable correct translation from the list of alternatives, is to perform a kind of back-translation, or *disalignment*, as the authors term it. Using the same techniques to compute an TL translation from an SL input sentence, an SL sentence is produced from the translation and the most well formed one indicates, in theory, the best TL translation.

2.3.3 Pattern-based Systems

Systems that make use of translation patterns (section 2.1.3) alleviate the problem of recombination in that correspondences between the text fragments and variables in translation patterns have been computed during the extraction or alignment phases so that they are explicitly labelled and there is direct information about the order of lexical items in the TL. Recombination typically operates by 'template matching' or retrieving the translation pattern that provides an 'optimal cover' of the SL input. Frequently, this is the pattern with the longest (contiguous) match sequence between the SL input and the SL side of the pattern. Unmatched parts of the SL input are bound to the SL variables of the

pattern. TL equivalents of the values of the SL variables are retrieved from the remaining translation patterns and bound to the appropriate variable position of the original or longest matching translation pattern. The ease of such a recombination phase is compensated for by the computational effort required by the translation pattern extraction and alignment phases. In the event of translation ambiguity, a score of translation confidence is usually attached to each translation solution. This is based on either or both of two criteria: a) how well the TL string is a translation of the SL string (perhaps by attaching frequency scores to the translation patterns) and b) to what extent the TL string is a valid word sequence of the TL.

This is the case for the approach of Güvenir & Cicekli. First, the most similar translation templates to the SL input are retrieved. Any unmatched parts of the SL input are bound to the SL variables of the template. Text fragments from the remaining templates matching the values of the variables are retrieved and their TL equivalents are bound to the appropriate variable in the original template. Retrieval of the TL equivalents of text fragments is trivial due to the fact that the fragments and variables in the templates have been aligned. Where the value of a bound SL variable cannot be matched by one single text fragment, it can be built up recursively by using more than one text fragment and combining the TL equivalents.

In the event of multiple translation solutions, strings are ranked according to a translation confidence score (Öz & Cicekli, 1998). This score is determined by attaching weights to the text fragments in the translation templates used to form the TL string. Although the templates are bi-directional, the weight varies depending on the direction of translation. The weight or confidence of a template is determined by how frequently a certain phenomenon is translated in the corpus. The score is therefore based on distributional information and the formula used (28) bears some similarity to Dice's co-efficient. As an example, to compute the confidence that text fragment *Y* is a translation of text fragment *X*, N_1 represents the number of sentence pairs in the corpus containing *X* in the SL sentence and *Y* in the TL sentence. N_2 denotes the number of sentence pairs in the corpus where *X* is found in the SL sentence, but *Y* is not found in the corresponding TL sentence. It is the calculation of the value N_2 which ensures the directional nature of the score. An additional score is attached to the rule combinations. The authors state that

"although some rules […] are assigned high confidence factors when they are considered as single rules […], they may have a very low confidence factor when they are used with other rules […]." (p.56). Each translation solution is compared with the TL sentences in the corpus. If a translation is identical to an TL sentence in the corpus, it is assigned a maximal confidence score of 1.0. Else, the Euclidean distance between a translation and (part of) an TL sentence from the corpus is calculated, based on the number of lexical items they share.

(28) $$\frac{N_1}{N_1 + N_2}$$

The system of Malavazos & Piperidis (2000) and Malavazos et al. (2000) is quite similar. The best matching, or 'optimally covering' translation templates for a given SL input are retrieved. This is based on a DP framework that assigns higher scores to matches with fewer and longer contiguous matching sequences of words and fewer variables. Unmatched parts of the SL input are bound to the SL variables of this pattern and their TL equivalents retrieved from the remaining set of patterns. In the event of multiple translation alternatives for the value of the SL variables, the context of each TL equivalent (±3 words) in the corpus is determined, rather like the 'hooks' proposed in Somers et al. (1994). Based on this idea of context, the TL chunk that provides the least amount of boundary friction is preferred. An additional matching and recombination technique is proposed where there are no applicable translation templates. Partially matching SL sentences in the corpus are retrieved (as in traditional TM) and the translations of unmatched sequences of the SL input are retrieved from the set of translations patterns as before.

Retrieval and recombination in the *Gaijin* system (Veale & Way, 1997) is also similar in that a single spanning translation template is used to match the SL input, but the SL text fragments that complete the match may be retrieved from other templates. Since the phrases are aligned and indexed, retrieving the appropriate TL chunk and inserting it into the appropriate slot on the TL side of the original template is trivial. Where multiple spanning translation templates are possible, they are prioritised according

to the number of identical phrasal segments they share with the SL input. However, the *Gaijin* system, like other adaptation-guided retrieval and case-based reasoning approaches, allows the adaptation or modification of the templates and phrasal segments they contain. Template matching is also the technique used in Kaji et al. (1992) where unmatched parts of the SL input are bound to the variables. However, the translations of those words or phrases is carried out by conventional MT, as is the case in Kinoshita et al. (1994).

On account of the amount of knowledge sources in the system of Carl (1999), matching an SL input against the templates and recombining is a more complex process. An SL input sentence is decomposed and generalised according to the translation templates. The sequences that match any of the templates are reduced recursively into one node, annotated with (external) morpho-syntactic constraints and (internal) constraints that determine the internal structure of the TL string. “Reduction” is analogous to matching an SL input against a set of templates. “Refinement” is the process where the equivalent TL chunks are retrieved according to the internal constraints. Production of the TL string is then a familiar process of generation from a morpho-syntactic derivation tree (“specification”).

The system of Echizen-ya (2000) layers multiple translation rules, both sentential and partial, when constructing TL translations. Multiple translation rules are layered by automatically producing a Translation Transition Network constructed from multiple translation rules that represent the basic structure of the whole sentence. The translation rules are layered from the general translation rules representing the basic structure of the sentence up to the concrete translation rules representing strings which bind to the variables. The system also employs a heuristic to distinguish between correct and erroneous translation rules determined by a process of feedback from the user. The frequency of correct translations produced when using certain translation rule combinations increases a *correct frequency count* attached to a rule combination. The converse heuristic is applied for erroneous translations, i.e. an *incorrect frequency count* is attached to rule combinations that frequently produce erroneous translations.

2.4 Translation Memory

TM systems also form part of the memory-based or corpus-based approaches to translation, since they also make use of a corpus of translated examples, typically sentences, as the principal bilingual knowledge source. However, in its traditional form, TM cannot be included under the MT framework since it does not create new translations. It is merely a database retrieval mechanism whereby the most similar SL sentences to a given SL input are retrieved from a database of SL and TL sentence pairs. The corresponding TL sentences are ranked according to the level of similarity between the SL input and the SL sentence in the database, then output to the user as a draft translation. The user selects the highest scoring match and edits the TL sentence as appropriate. It is therefore an aid to translators, placing it in the Machine Aided Human Translation category, as pointed out by Kay (1980).

Typically, TM can only handle a single sentence pair at a time and consequently cannot combine fragments from more than one sentence pair to match or cover an SL input. Therefore, the effectiveness of a TM system relies on a fairly close match between the SL input and an SL sentence in the database. Newer systems, such as *Translator's Workbench* from Trados[2] have the ability to translate certain fixed expressions such as prices and dates, but this additional functionality is of limited use.

However, there is debate as to whether TM forms part of the MT paradigm or not. What we have pointed out above is the fact that TM and EBMT may be distinguished on the basis of whether new translations are formed or not. Traditional TM systems contain no TL knowledge whatsoever and are thus fully portable to new TLs. It is debateable whether they contain SL knowledge – as they sometimes claim; if they do not, they are fully portable to new SLs as well.

From a slightly different angle, TM and EBMT may be viewed as two techniques that lie at opposite ends of a spectrum in memory-based translation (Figure 11). EBMT traditionally requires varying amounts of external linguistic resources to compute the subsentential relationships between SL and TL translation examples necessary to form TL translations. On the other hand, TM does not require any external linguistic

[2] http://www.trados.com

knowledge: consequently, it cannot carry out any linguistic operations apart from computing a score of similarity between two SL sentences. However, there are EBMT techniques (as outlined in section 2.1.3.1) that can compute relationships between subsentential fragments and recombine them to form translations without the need for external linguistic resources. In addition, there have been recent attempts to produce a 'new generation' of TM systems that employ techniques from EBMT in order to improve flexibility, i.e. by being able to produce translation suggestions from information contained in more than one translation example (Langé et al., 1997; Planas & Furuse, 1999; McTait et al., 1999; McTait, 2000; Carl & Hansen, 2000).

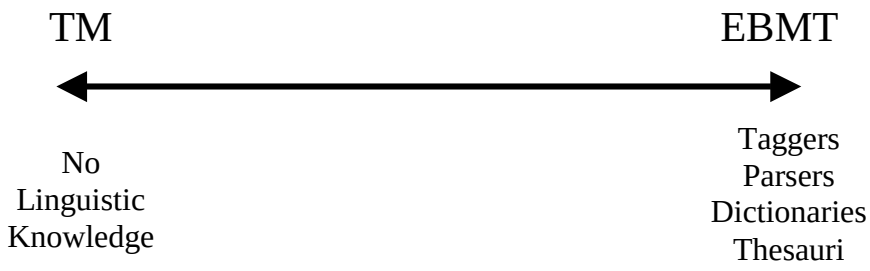

Figure 11: Spectrum of Memory-Based Translation

Producers of commercial systems have also begun to realise the necessity of incorporating information from more than one source and combining it in some way to produce TL translations that did not previously exist in the TM database of translation examples. The *Déjà Vu* TM system[3] claims to use techniques from EBMT in order to assemble an TL sentence from information outside a single translation example. In actual fact, *Déjà Vu* attempts to translate smaller portions of the SL sentence by matching parts of it against entries in the TM database and the (manually created) terminological lexicon and then 'reassembling' those translations to produce an TL string. The exact process is not revealed in the user manual or other documentation. The distinction between TM and EBMT is therefore changing and not clearly defined. Moreover, it may be said that TM is taking on more and more characteristics of EBMT.

[3] http://www.atril.com

2.5 Conclusions of Chapter

The approaches to EBMT that are relevant to this thesis have been presented and analysed critically with respect to their advantages and disadvantages. The most similar approaches have been presented in particular detail. Techniques outside EBMT, such as sequence alignment algorithms, have also been discussed as they are central to the approach to EBMT proposed. More specifically, the shortcomings of previous systems have been highlighted. The following is a summary of the findings of this chapter in that algorithms or approaches that require extension or improvement are highlighted. This thesis therefore attempts to address these issues.

Structure-based EBMT systems are knowledge intensive. They require significant knowledge sources, such as taggers, parsers and bilingual lexicons, in order to store translation examples as annotated tree structures. Knowledge intensive techniques decrease the portability of a system to new language pairs since reliable linguistic resources must be available for both languages in the corpus. Such resources may be difficult to acquire and expensive to produce. Moreover, given the state of the art, in parsing technology for example, such resources are not guaranteed to produce correct results either. Keeping external linguistic information to a minimum is preferable to maintaining portability – a major advantage of EBMT.

On one hand, structure-based EBMT systems provide detailed knowledge about the structure of the SL and TL. In addition, constraints are specified by the linguistic resources used. Translations therefore have a greater chance of being well formed. On the other hand, linear systems cannot accurately ensure that translations are well-formed, but the fact that they do away with significant knowledge sources makes them highly attractive. The hypothesis that an increase in linguistic knowledge in a system improves performance is tested in this thesis. Furthermore, data-driven approaches to inferring bilingual knowledge on linguistic levels deeper than that of the word forms in a corpus is an area that has received surprisingly little attention.

Learning translation patterns from sentence aligned bilingual corpora by purely analogical methods offers a convenient approach to constructing a language-neutral EBMT system. However, existing techniques, such as those presented in section 2.1.3.1, only form translation patterns from lexical items from two sentence pairs. One sentence

pair is compared to one and only one other sentence pair. If more sentence pairs were compared, i.e. if more textual evidence from the corpora were used, the resulting translation patterns would be more accurate (McTait & Trujillo, 1999).

Many EBMT systems restrict themselves to computing bijective or one-to-one correspondences between text fragments. This is also the case for the pattern-based systems described in section 2.1.3.1. Systems that *are* able to compute non-bijective alignments, such as SMT systems, are unidirectional in nature, require large amounts of training data and are often restricted to word-level models of translation. Most word, term or phrase alignment algorithms are similarly restricted. This reduces the flexibility of the systems when describing translation phenomena such as the translation of English phrasal verbs, made up of a verb and a particle, into languages where the equivalent is a single token, for example, when *give … up* is translated into French as *abandonner* (Figure 1).

While 'structure-based' EBMT systems make use of structural knowledge as a heuristic when computing alignments between SL and TL tree structures, existing string-based alignment algorithms for aligning text fragments in translation patterns are insufficient. They are unable to align text fragments successfully between languages that are structurally divergent. While the DP algorithm has proved useful for sequence alignment problems such as spelling correction and sentence alignment, the alignment of subsentential text fragments is more problematic. The DP algorithm is capable of computing only adjacent alignments and since the order of text fragments between two languages may not be similar, an extension of the DP framework is required in order to compute long-distance dependencies. Exhaustive search algorithms are able to compute adjacent and non-adjacent alignments, but are prone to combinatorial explosion making them impractical even for very short sequences. SMT systems are able to compute long-distance dependencies, but suffer from the restrictions outlined above.

3 Methodology

This chapter describes in detail the EBMT methodology introduced in Chapter 1. This includes details of the translation pattern extraction algorithm, the algorithm to align the text fragments and variables of which the translation patterns are composed and the recombination phase where TL translations are produced. Initial results are presented.

3.1 Translation Patterns

In order to make clear what is extracted from the corpus, translation patterns are defined formally. As described in Chapter 1, a translation pattern is composed of a sequence of SL subsentential text fragments and a sequence of TL subsentential text fragments. The bilingual relationship (alignments) between the SL and TL text fragments in the two sequences is made explicit. A translation pattern also contains a sequence of SL variables and a sequence of TL variables. The variables represent the text that appears between the text fragments. The bilingual relationship between the variables is also made explicit. The sequences of text fragments and variables are viewed as separate sequences and are aligned in separate procedures.

A translation pattern is defined formally as a 4-tuple: {*S*, *T*, A_f, A_v}. *S* (*T*) represents a sequence of SL (TL) text fragments, separated by SL (TL) variables representing text fragments. A subsentential text fragment is defined as a continuous series of one or more lexical items or tokens. Text fragments form the fundamental units of which the translation patterns are composed. In *S*, there can be any number *p* (*p>0*) of SL text fragments F_p^S with *p*, *p+1* or *p-1* SL variables V_p^S. In *T*, there can be any number *q* (*q>0*) of TL text fragments F_q^T with *q*, *q+1* or *q-1* TL variables V_q^T. One possible configuration of the text fragments and variables in *S* and *T* is given in Figure 12, where there are as many SL (TL) variables as there are SL (TL) text fragments.

$$F_1^S, V_1^S, F_2^S, V_2^S \ldots F_p^S, V_p^S \leftrightarrow F_1^T, V_1^T, F_2^T, V_2^T \ldots F_q^T, V_q^T$$

Figure 12: Possible Configuration of S and T

A_f represents the global alignment of text fragments between *S* and *T*. A_v represents the global alignment of the variables between *S* and *T*. A_f is defined as a set of local alignments $\{<A,B>_1, <A,B>_2 \ldots <A,B>_k\}$ where each local alignment is represented as a pair <*A*,*B*>. A (B) represents a pointer to zero or more SL (TL) text fragments, according to the local alignment patterns stipulated by the sequence comparison algorithm (section 3.2.3.2). The global alignment of variables A_f is represented analogously.

In order to store the translation patterns in an efficient and space-saving manner, the text fragments *F* of which they are composed are represented as pointers to the positions in the corpus from which they were extracted. In more detail, each text fragment *F* is represented by a data-structure with the following fields:

- Pointer to the start-position of *F* in the sentence from which it was retrieved.
- Pointer to the end-position of *F* in the sentence from which it was retrieved.
- Pointer to the index of the translation example (sentence-pair) containing *F*.
- Pointer to the side of the corpus - SL or TL - containing *F*.

As an example, consider the data structure of text fragment *F*, made up of the following values such that *F* is equivalent to {4,8,43,0}. This entails that the text fragment *F* spans 5 words in length, starts at position 4 and terminates at position 8 in the SL sentence of translation example 43 in the corpus. The SL side of a corpus is denoted by 0 and the TL side by 1. The concept is depicted in Figure 13.

3.2 Translation Pattern Extraction

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 sentence-level alignments in the corpus are presumed to be correct. In the absence of sentence-aligned corpora, any of the statistical corpus alignment algorithms (Brown et al., 1991; Gale & Church, 1991, 1993; Church, 1993; Kay & Röscheisen, 1993) or any of the lexical hybrids thereof (Chen, 1993; Simard et al. 1992; Simard & Plamondon, 1998) may be used to align the sentences in parallel corpora. Hand-checking is required to ensure

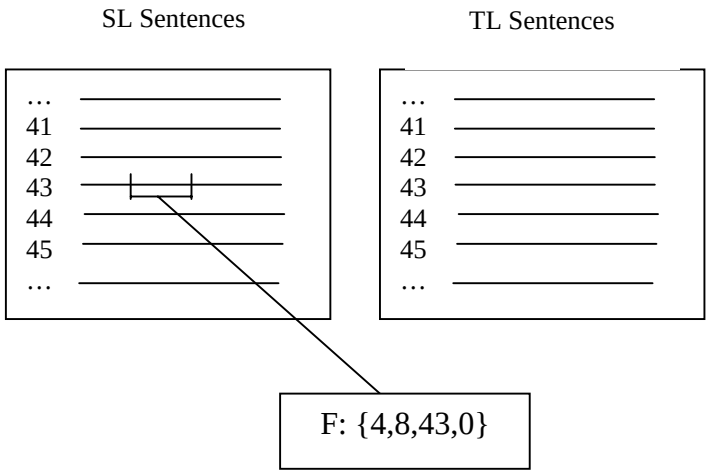

Figure 13: Depiction of F Stored as a Set of Pointers to Positions in the Corpus

that the resulting corpus is aligned with 100% accuracy. Only 1:1 sentence alignments are considered to be useful. Other local sentence alignment patterns, such as 1:2, are discarded. In the experiments undertaken in this thesis, the corpus has been checked so that each SL sentence has been aligned with an equivalent TL sentence.

Translation patterns are extracted by means of a language-neutral recursive machine-learning algorithm. It operates on the simple principles of string co-occurrence and frequency thresholds (similar distributions of strings): possibly discontinuous pairs of SL and TL strings that co-occur in a minimum of two translation examples (sentence pairs) in a bilingual corpus are likely to be translations of each other. No bilingual dictionaries or other language or language-pair specific resources are required.

Since strings are only required to co-occur in a minimum of 2 sentence pairs (where 2 is the frequency or co-occurrence threshold), the algorithm is useful in instances of sparse data. This makes it amenable to 'less-studied' languages where there are limited supplies of bilingual corpora. If the frequency threshold is increased so that SL and TL strings are required co-occur in a greater number of translation examples, the accuracy of the patterns is increased (McTait & Trujillo, 1999).

Translation patterns provide sentential translation contexts. However, in the absence of a bilingual lexicon, the text fragments and variables of which the translation patterns are composed are aligned in order to provide the recombination phase (section

3.3), where TL translations are produced, with a more refined bilingual knowledge source. Language-neutral techniques, such as bilingual lexical distribution and cognate matching, are used to align the text fragments and variables.

The resulting translation patterns resemble transfer rules in traditional RBMT, but are simpler, shallower and have fewer linguistic constraints. The text fragments and variables are neither marked for number, gender, person, case nor syntactic category. Translation patterns are, strictly speaking, generalisations of sentences that are translations of each other. Sentences are generalised by identifying recurring word groups (sequences of text fragments), aligning them and then aligning any discontinuities (sequences of variables) that may occur between these word groups.

Translation pattern extraction is divided into 3 clear stages: The monolingual phase (3.2.1), the bilingual phase (3.2.2) and the alignment phase (3.2.3) where the text fragments and variables are aligned. The monolingual phase is applied independently to the SL and TL sentences of the corpus and is divided further into 3 stages: tokenisation, word list construction and collocation formation. In the monolingual phase, lexical items that occur in 2 or more sentences are collected. From these, all recurrent contiguous and non-contiguous strings (collocations) are computed, noting where the discontinuities arise. The bilingual phase involves equating the SL and TL collocations formed in the monolingual phase on the basis of simple co-occurrence criteria. Finally, in the alignment phase, the bilingual relationship between the SL and TL strings (text fragments) of which the collocations are composed is made explicit. The discontinuities, which become variables in the translation patterns, are aligned analogously.

Although the algorithms presented in this thesis are essentially language-neutral in nature – at least for Indo-European languages – in the experiments undertaken, English and French have been used.

3.2.1 Monolingual Phase

Tokenisation is a simple, yet language-specific task. The tokenisation process for English is largely simplified so that tokens are identified by white space characters and other delimiters such as punctuation. The apostrophe is not included in the list of delimiters as this would incorrectly separate clitic expressions such as *can't* (*cannot*) and possessive

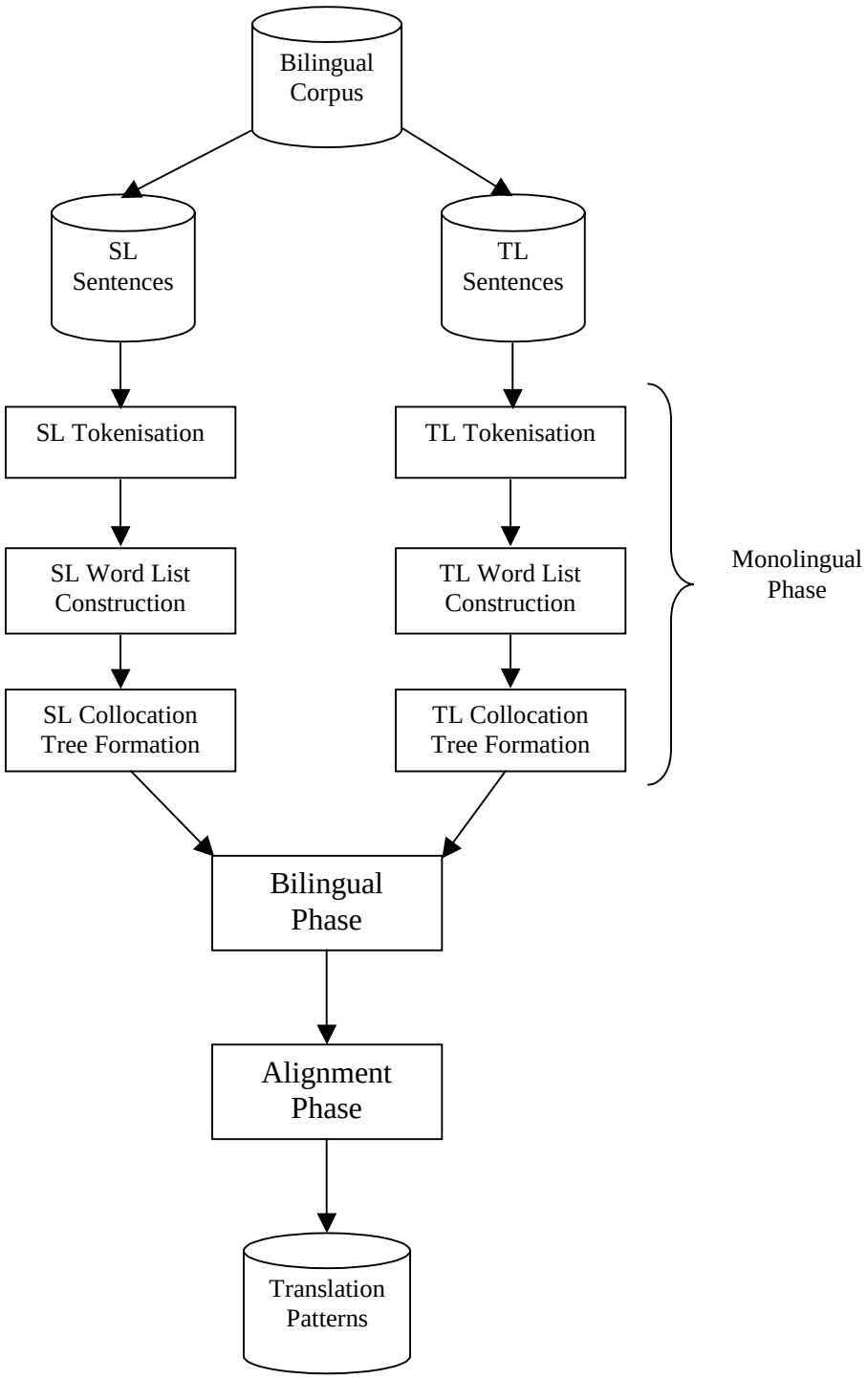

Figure 14: Translation Pattern Extraction

expressions containing an apostrophe such as *president's*. Tokenisation for French is slightly more complex since the apostrophe frequently joins articles, pronouns and prepositions to nouns, verbs and adjectives etc. In such cases, the list of articles, prepositions and pronouns that change their orthography (usually by losing the final *e*) when joined to a vowel-initial word by an apostrophe is included in the tokenisation module. For example, *l'ordinateur* is separated to become *le* + *ordinateur*. Similarly, *d'initiative* becomes *de* + *initiative* etc. Since the translation pattern extraction algorithm is based on string co-occurrence, it is desirable for there to be the greatest possible amount of string normalisation so that string co-occurrence is maximised.

After tokenisation, lists of lexical items (tokens) that occur in a minimum of two sentences are retrieved, together with a record of the sentences in which they were found. The frequency threshold specifies the number of sentences in which lexical items are to co-occur. Given the corpus sample in (29)[4], the lexical items occurring in two or more SL sentences (30) and the lexical items occurring in two or more TL sentences (31) are collected. The integers denote the sentences from which they were retrieved.

(29) 1. The commission **gave** the plan **up** ←→ La commission **abandonna** le plan
2. Our government **gave** all laws **up** ←→ Notre gouvernement **abandonna** toutes les lois

(30) (gave)[1, 2], (up)[1,2]

(31) (abandonna)[1, 2]

The lexical items collected are allowed to combine to form longer word combinations or *collocations*. They are termed collocations as they form "some sort of arbitrary and recurrent word combination" (Benson, 1990). In this instance, a collocation is defined as a data structure representing (possibly discontinuous) strings that co-occur in 2 or more sentences. The lexical items combine recursively to form a tree-like data structure of collocations. Each lexical item is tested to see whether it can combine with any daughters of the root node, and if so, recursively with each subsequent daughter (and

[4] The term, 'corpus sample' denotes the sentence pairs of a corpus which constitute all the relevant knowledge to form the translation pattern (as if the sample were the entire corpus).

the daughters of that node and so on), as long as there is an intersection of at least two sentence IDs. The combination process is constrained only by the integer IDs of the sentences from which the lexical items were retrieved and the frequency threshold (a minimum of 2). The intersection constraint, which matches the frequency threshold, enforces string co-occurrence in 2 or more sentences. If the node to be added cannot combine with any daughter of the root (or parent) node, it is added as a new daughter of the root (or parent) node.

The result is a tree of collocations of increasing length, but decreasing frequency as the tree is descended from the root node to the leaves. Therefore, the leaves become the most informative parts of the tree and are collected at the end of this phase. The leaf-collocations are filtered so that only the longest are selected. Collocations that are subsumed by other collocations with the same sentence IDs are considered to be spurious and are removed. The longest collocations provide more context, hence the potential for ambiguity is reduced.

As an example, the first SL lexical item *(gave)* is added to the root node of the SL collocation tree as a new daughter, as it currently has no daughters (Figure 15). The integers in the nodes again represent the sentence IDs from which the lexical items were retrieved. The lexical item *(up)* is subsequently tested for combination with any daughter of the root node and any of its subsequent daughters and so on. It is therefore tested for combination with the daughter of the root node *(gave)*. Since there is an intersection of two sentence IDs between *(gave)* and *(up)*, they are allowed to combine to form the longer word combination or collocation *(gave)(up)*. This new collocation is added as a daughter node of the node (*gave*). The sentence IDs of the new daughter node are formed from the intersection of the IDs of the two nodes. In this case the intersection of the sentence IDs of *(gave)* and *(up)* is *[1,2]*.

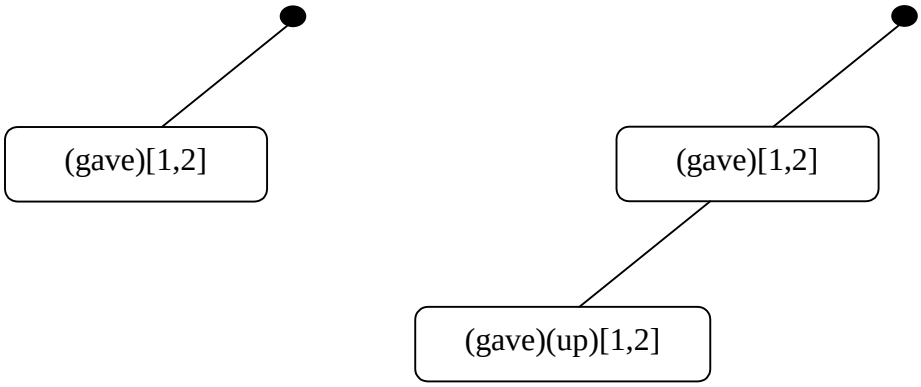

Figure 15: SL Collocation Tree

In the case of the TL collocation tree, the first (and only) lexical item, *(abandonna)* is added as a daughter of the root node, as there are no other daughters with which to test for combination. Since there are no further lexical items that occur in two or more sentences in the corpus, the TL collocation tree remains as in Figure 16. Collecting the leaves from the SL collocation tree produces the leaf-node list *{(gave)(up)[1,2]}* and collecting the leaves of the TL collocation tree returns the list *{(abandonner)[1,2]}*. Since the lists are one item in length, no filtering is necessary.

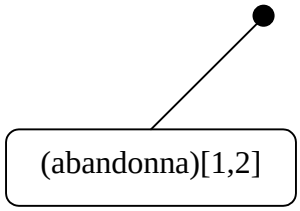

Figure 16: TL Collocation Tree

The process of collocation tree construction is described in more detail: The top-level collocation tree construction algorithm is given in Figure 17. The first line of this (and each subsequent) algorithm indicates the name of the function, the input parameters and the return. The algorithm `CollectCollocs()` takes as input a list of lexical items containing a record of the sentences from which they were retrieved. (30) is such a list. A

tree of collocations of increasing length and decreasing frequency is constructed, as in Figure 15 and Figure 16. The output is a list of filtered leaf-node collocations retrieved from the tree. First, the Collocation Tree `CollocTree` is initialised. This provides the root node of the tree to which each member of the list of lexical items is tested for addition. Subsequently, these lexical items are referred to as 'collocations' as they fulfil the criteria defined above, although they are only one word in length. Each collocation is tested for combination with any of the daughters of the root node by means of the function `AddToChildrenColloc()`. If combination is possible, it is further tested, recursively and in the same manner, for combination with any of the subsequent daughters and so on. The test for combination is the intersection of at least two sentence IDs. If the collocation cannot be added to any daughters of the root node (or indeed any other parent node), it is added as a new daughter of that parent node by `AddChildColloc()`. Once all the collocations have been tested for combination and added to the tree, the leaf-nodes are collected. They are filtered so that any collocation subsumed by another collocation with the same sentence IDs is removed. The lexical items of which the filtered collocations are composed are ordered so that they appear in the same sequence as they do in the sentences of the corpus. The result is a list of the longest, possibly discontinuous, collocations or recurrent word sequences in the SL or TL sentences of a corpus.

```
CollectCollocs(LexicalItems) -> LeafNodeList
   CollocTree = new Colloc();
   For each item L in LexicalItems
      Colloc = new Colloc(L)
      If AddToChildrenColloc(Colloc,CollocTree)
         Do Nothing
      Else AddChildColloc(Colloc,CollocTree)
   LeafNodeList = CollectLeaves(CollocTree)
   Filter(LeafNodeList)
   OrderLexicalItems(LeafNodeList)
   Return LeafNodeList
```

Figure 17: Top-Level Collocation Tree Construction Algorithm

The function `AddToChildrenColloc()` is detailed in Figure 18. The collocation to be added to the tree `Colloc` is tested for combination with each child of the current

node of the tree `CollocNode`. The function `Collocate()`, detailed in Figure 19, carries out the test for combination between the two collocations. If combination is possible, `Colloc` is tested for combination with any subsequent daughters by calling `AddToChildrenColloc()` or it is added as a new daughter node by `AddChildColloc()`. This has the effect that each collocation to be added to the tree is recursively tested for combination with the nodes of the tree, filtering down from the root node to the branches. Since the order of the lexical items in the list to be added to the tree – `LexicalItems` in Figure 17 - is not specified, it is frequently necessary to split the tree. This is explained in more detail with reference to the procedure outlined in Figure 21. Finally, if any collocation was added to the tree, the Boolean value `true` is returned.

```
AddToChildrenColloc(Colloc,CollocNode) -> True | False
   For each child C of CollocNode
      Collocate(Colloc,C)
   TestForSplitTree(Colloc,CollocNode)
   If any Colloc added to Children of CollocNode
      Return True
   Else Return False
```

Figure 18: AddToChildrenColloc() Function

The `Collocate()` function in Figure 19 attempts the combination of the collocation to be added `Col1` with an existing node `Col2`. If there is an intersection of at least two sentence IDs, then the collocation to be added `Col1` is tested for combination recursively with each subsequent daughter of `Col2` and so on, until it is added as a new daughter by means of the function `AddChildColloc()`.

```
Collocate(Col1,Col2) -> True | False
If Intersection(IDs(Col1),IDs(Col2)) ≥ 2
   If AddToChildrenColloc(Col1,Col2)
      Do Nothing
   Else AddChildColloc(CombineCollocs(Col1,Col2),Col2)
Else Return F
```

Figure 19: The Collocate() Function

The function `CombineCollocs()` creates and returns a new node by combining the lexical items from both the collocation to be added `Col1` and the node to be added to `Col2`. The intersection of the integer IDs between `Col1` and `Col2` becomes the new list of integer IDs (Figure 20).

```
CombineCollocs(Col1,Col2) -> NewColloc
   NewColloc = new Colloc()
   NewColloc.SetIds(Intersection(IDs(Col1),IDs(Col2))
   NewColloc.SetLexicalItems(LexItems(Col1), LexItems(Col2))
   Return NewColloc
```

Figure 20: Combining Collocations

The function `AddChildColloc()` adds collocations as new daughters of the current parent node. This indicates that no further combinations are possible with subsequent daughters of the current node. On account of the recursive nature of the algorithm and that node combination is based on the intersection of integer IDs, a collocation may be added as a daughter of more than one node.

Figure 21 details the function used to split the tree. This is necessary when a collocation to be added `Col1` subsumes one or more children of the node being added to `Col2`. In such an event, all the subsumed children must become children of the node being added. This ensures that the longest possible collocations are extracted.

```
TestForSplitTree(Col1,Col2) -> void
   TempColloc = CombineCollocs(Col1,Col2)
   For each Child C of Col2
      If StrictSubsetP(IDs(C),IDs(TempColloc))
         SplitP = True
         AppendDaughters(TempColloc,C)
         DeleteChild(Col2,C)
   If SplitP == True
      AppendDaughters(Col2,TempColloc)
```

Figure 21: Function for Splitting the Collocation Tree

As an example, consider the following list of collocations to be added: *{(A)[1,2,3,4], (B)[3,4], (C)[2,3,4]}*. Figure 22 depicts how each collocation from the list is added to the tree with no tree-splitting procedure. Adding the collocations *(A)[1,2,3,4]*

and subsequently *(B)[3,4]* is unproblematic as indicated in the first schematic. The second schematic depicts how *(C)[2,3,4]* is added to the tree. The result is undesirable since *(B)[3,4]* is subsumed by *(C)[2,3,4]* and could combine to form the longer collocation *(A)(B)(C)*.

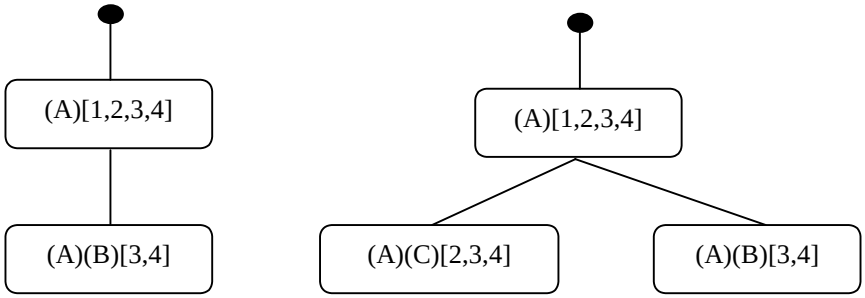

Figure 22: Collocation Tree Construction without Tree-Splitting

Figure 23 depicts the tree-splitting procedure, with the desired result, when the collocation *(C)[2,3,4]* or `Col1` in Figure 21 is added to the node *(A)[1,2,3,4]* or `Col2`. First, each child of the node `Col2` is combined with `Col1`. As a result, the collocation *(A)(C)[2,3,4]* or `TempColloc` is formed. Subsequently, each daughter of the original parent node - *(A)[1,2,3,4]* or `Col2` - is tested for addition as a daughter of `TempColloc`. In the event where the set of sentence IDs of a daughter node of `Col2` is a strict subset of the set of sentence IDs of `TempColloc`, it is added as a daughter of `TempColloc` then deleted from its original parent node `Col2`. In this example, the set of sentence IDs of the daughter of `Col2` – *(A)(B)[3,4]* – is a strict subset of the set of the sentence IDs of `TempColloc` - *(A)(C)[2,3,4]*. Therefore, the daughter of `Col2` is combined with it and added as a daughter of `TempColloc`. Finally, the daughter of `Col2` is deleted from `Col2`. The test for a strict subset between sets of sentence IDs is used so that spurious nodes are not added. At the end of the loop, if the tree is required to be split i.e. the set of IDs of one of the daughter nodes of Col2 was a strict subset of those of TempColloc, the node `TempColloc` is added as a daughter of the original parent node `Col2`.

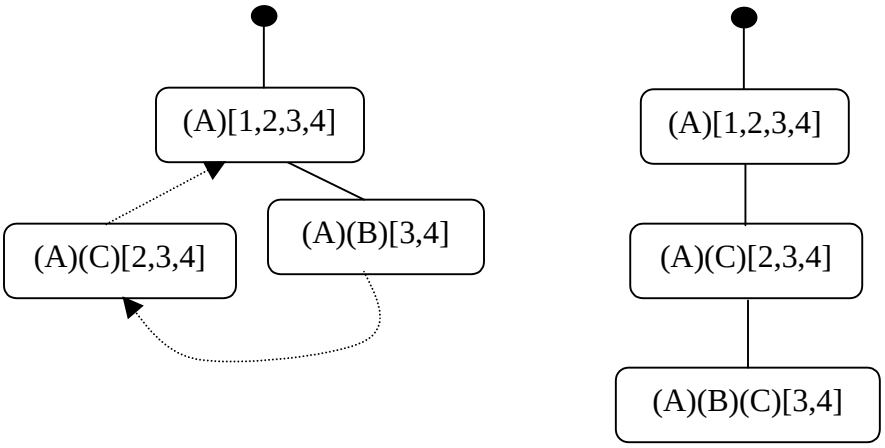

Figure 23: Splitting the Collocation Tree

The collocation trees shown thus far are highly simplified examples. They can grow to become much larger in size and more complex. A more substantial example is given in Figure 24, given the following list of collocations to be added: *{(government)[1,2,3,4], (implemented)[2,3,4], (plan)[3,4], (policy)[2,3], (decided)[1,2], (committee)[5,6,7]}*. The dotted lines indicate potential extensions to the tree. In fact, new nodes can be added to any node in the tree, as long as there is an intersection of at least 2 sentence IDs.

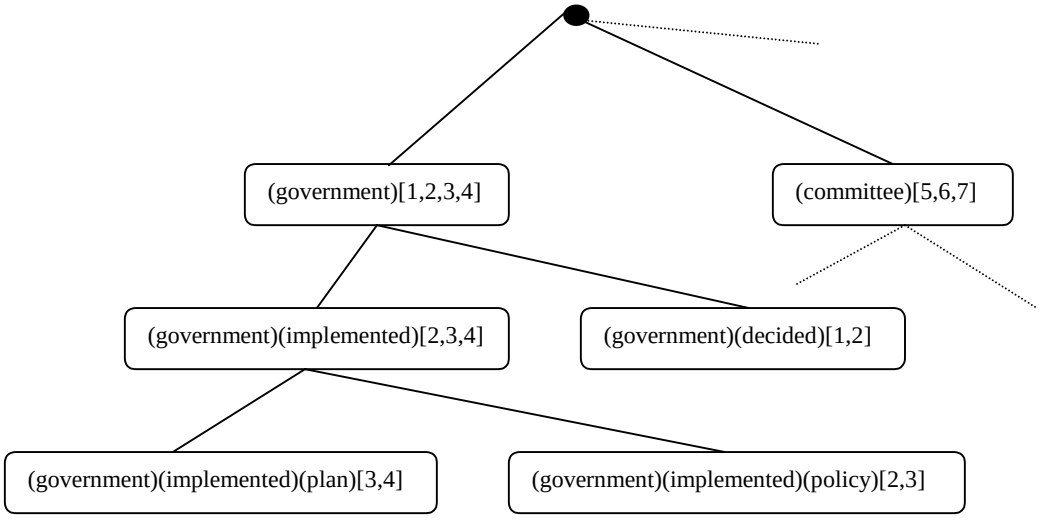

Figure 24: More Substantial Collocation Tree

The function `CollectLeaves()` in Figure 25, collects the leaf-nodes of the tree of collocations. Starting from the list of collocations that are the daughters of the root node `CollocList`, the tree is searched recursively until the terminal nodes are found and collected. Terminal nodes are defined as those that do not have any daughters.

```
CollectLeaves( CollocList ) -> LeafNodeList
   LeafNodeList = new List()
   For each Colloc C in CollocList
      If HasDaughtersP(C)
         CollectLeaves(GetDaughters(C));
      Else
         LeafNodeList.addElement(C);
   return LeafNodeList;
```

Figure 25: Leaf-Node Collection Algorithm

The function `Filter()` in Figure 26, filters the list of leaf-node collocations so that only the longest remain. Each leaf-node collocation is tested against each subsequent leaf-node collocation. Where a pair of collocations share the same set of sentence IDs – the collocations occur in the same events – the longest collocation is retained and the other removed.

```
Filter(LeafNodeList) -> FilteredLeafNodeList
   For each Colloc Ci in LeafNodeList
      For each Colloc Ci+1 in LeafNodeList
         If IDs(Ci) = IDs(Ci+1) AND Length(Ci) > Length(Ci+1)
            Remove(Ci+1,LeafNodeList)
   Return LeafNodeList
```

Figure 26: Leaf-Node Filtering Function

The output of the monolingual phase is therefore a list of sorted, filtered leaf-node collocations representing the longest possible continuous or discontinuous recurrent word combinations in a given set of SL or TL sentences.

3.2.2 Bilingual Phase

In the bilingual phase, the filtered SL and TL leaf-node collocations extracted in the monolingual phase are equated on the basis of simple co-occurrence criteria, as the bulk of the processing has been carried out during the monolingual phase. SL collocations that occur in the same events as TL collocations are considered to be translations of each other. In more detail, SL and TL collocations that share exactly the same sentence IDs are equated. This enforces the principle where SL and TL strings that co-occur in two or more sentence pairs are considered to be translations of each other.

The leaf nodes of the SL and TL collocation trees in Figure 15 and Figure 16 respectively are equated as they both have exactly the same sentence IDs (32). The correct orthography or order of the lexical items and discontinuities that make up a translation pattern is computed by scanning the sentences from which the lexical items were retrieved (33). The discontinuities are represented in the examples as ellipses. They subsequently act as variables. Since, by definition, the lexical items of which the collocations are composed are found in 2 or more sentences, the orthography is chosen from either (or any) of those two sentences. In practice the simplest, i.e. the one that provides the least number of discontinuities, is selected.

(32) (gave)(up) ←→ (abandonna)

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

The test for equating SL and TL collocations is more accurately described in terms of the `Set-Exclusive-Or()` of the set of SL and TL sentence IDs of the collocations (34). The IDs that appear in one set, but not both, is returned. Therefore, an empty list must be returned for the `Set-Exclusive-Or()` of the sentence IDs of an SL and TL collocation if they are to be equated.

(34) $\texttt{Size(Set-Exclusive-Or(IDs(Colloc}_{SL}\texttt{),IDs(Colloc}_{TL}\texttt{)))=0}$

Translation patterns are not formed from the inner leaves of the collocation trees. Only terminal nodes are considered as they represent the longest possible recurrent word sequences. The effectiveness of EBMT lies in the retrieval of the longest possible matching segments. This reduces translation ambiguity, passage and boundary friction (as explained in Nirenburg et al. 1993:48). If translation patterns were formed from non-terminal nodes, they would be subsets of translation patterns formed from the terminal nodes and would therefore be spurious. They also risk being incomplete and inaccurate. Consider the collocation trees in Figure 15 and Figure 16. Equating the non-terminal SL collocation (gave)[1,2] with the TL node (abandonna)[1,2] would produce an erroneous bilingual relationship, despite the fact that they both have exactly the same sentence IDs.

It is intuitive that if the text fragments of which the translation patterns are composed are translations of each other, then the discontinuities or variables that occur between them must also be translations of each other. This is so, since the variables represent the text appearing between the text fragments in the translation pattern. Since translation patterns are composed of lexical items that occur in two or more sentences in a corpus, then at least two *complement* translation patterns are formed. They are created simply as the inverse of a regular translation pattern. The variables become text fragments and the text fragments become variables. Furthermore, complement translation patterns may contain lexical items that occur only once in the corpus (*hapax legomena*). From the translation pattern (33) and the corpus sample in (29), the complement translation patterns formed are given in (35).

(35) The commission (…) the plan (…) ←→ La commission (…) le plan

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

3.2.3 Alignment of Text Fragments and Variables

The text fragments of which the translation patterns are composed are aligned. The sequences of discontinuities or variables that occur between the text fragments are aligned analogously, but as separate sequences. Aligning the text fragments and variables produces translation patterns that are flexible enough for the recombination phase, where

TL translations are produced (section 3.3). Alignment of the text fragments creates a more refined bilingual knowledge source, in effect, producing a bilingual lexicon of phrasal translations.

Given that a translation pattern contains a sequence of SL text fragments and a sequence of TL text fragments, the problem is viewed as *sequence alignment* or computing the optimal or most probable global alignment between two sequences of text fragments in two languages. The two sequences of variables are aligned analogously by considering the sequences of text fragments that the variables represent - as defined by the corpus - as separate sequences of text fragments.

The alignment of text fragments in translation patterns is analogous to the process of conventional bilingual alignment such as the alignment of words, terms, phrases and even sentences. Consequently, it involves similar problems and requirements. Hence, the solution involves both a sequence-comparison algorithm and a bilingual similarity metric to determine the maximum likelihood alignment between the sequences of text fragments.

The bilingual similarity metric operates without the need for external linguistic resources such as bilingual lexicons. Instead, it is based on language-neutral techniques such as bilingual lexical distribution as determined by the frequency data in the corpus and comparing similar word forms across languages (cognates). For each local alignment, a score representing a real value between zero and 1 is returned, indicating the probability that the texts fragments are translations of each other.

Like familiar alignment algorithms such as corpus alignment, the sequence comparison algorithm must handle substitutions, insertions, deletions, compression and expansion. This enables the computation of bijective (1:1) and non-bijective bilingual relationships of the type *m:n* where $m \neq n$. Unlike the order of sentences between two parallel texts, the problem of sequence alignment in translation patterns is compounded by the fact that the order of words or subsentential text fragments between two languages is often dissimilar. Therefore, an algorithm for aligning text fragments must also handle non-adjacent alignments in order to compute long-distance dependencies. Figure 27 shows an example.

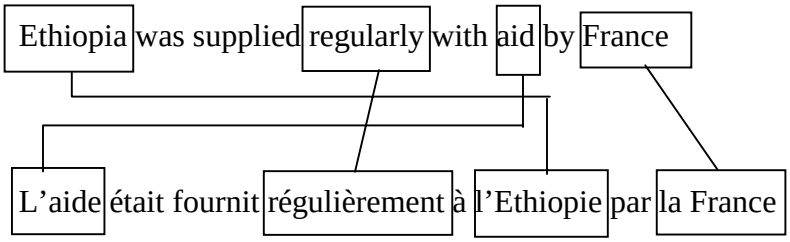

Figure 27: Example of Non-Adjacent Alignments

Furthermore, the sequence comparison algorithm must execute in a practical asymptotic running time. An exhaustive search algorithm is capable of computing alignments between adjacent and non-adjacent text fragments, but is prone to a combinatorial explosion in the search space, making it impractical even for very short sequences. The well-known Dynamic Programming (DP) algorithm provides a partial solution in that it runs within an acceptable upper bound and can compute substitutions, insertions, deletions, compression and expansion, but it is inherently incapable of computing non-adjacent alignments.

An algorithm is proposed that performs the alignment in two passes. The first pass involves using DP to compute initial alignments assuming that all local alignments are adjacent. Subsequently, the (possibly empty) set of non-adjacent alignments are computed by a second-pass algorithm. If any non-adjacent alignments are computed, they are recorded and removed from the two sequences, which are then realigned with DP. The same bilingual similarity metric is used in both passes. The final global alignment is a concatenation of the non-adjacent alignments and the second application of the DP algorithm. If no non-adjacent alignments were computed in the second-pass algorithm, the DP algorithm is not reapplied, in which case the final global alignment is the output of the initial application of DP.

3.2.3.1 Bilingual Similarity Metric

The bilingual similarity metric, *BS* in (36), is a combined score based on the number of cognates the text fragments share and the similarity of the distributions of the text fragments *BLD* (bilingual lexical distribution). The score weights cognates heavily and is

consequently useful in the context of experiments with English and French which share a substantial number of cognates. It may be appropriate to change the weightings with different language pairs where cognates play less of a role.

$$BS = \frac{BLD + |Cognates|}{1 + |Cognates|} \quad (36)$$

Bilingual lexical distribution (*BLD*) is based on the following principle: Given a bilingual corpus aligned at the level of the sentence, if an SL text fragment and an TL fragment appear in the same sentence pairs in virtually all cases – and conversely, if an SL and TL fragment do not appear in the same sentence pairs in virtually all cases – this is a good clue for determining translation equivalents. Bilingual lexical distribution *BLD* is computed using Dice's co-efficient (37), returning a real value such that $0 \leq BLD \leq 1$. *S* (*T*) represents the set of SL (TL) sentences containing the SL (TL) text fragment in question.

$$BLD = \frac{2(|S \cap T|)}{|S| + |T|} \quad (37)$$

In order to refine the scores produced by (37), closed class words, as defined by manually created stop-lists, are removed from the text fragments. Closed class words are of such high frequency that they are unable to be aligned by a distributional score such as (37), therefore they distort the values returned for the remainder of the text fragment. Removing closed class words from the fragments returns higher and truer probability scores that one fragment is a translation of the other. However, the addition of manually created stop-lists of closed-class words compromises to some extent the language-neutral nature of the similarity metric, although they are extremely quick and easy to produce. Alternatively, stop lists can be compiled automatically in a language-neutral manner simply on the basis of frequency of co-occurrence.

The distributions of text fragments – which are frequently greater than one word in length – is sought rather than the distributions of single lexical items that the text fragments contain. The longer the fragment, the more context and the less potential for

ambiguity, resulting in more accurate distributional scores. However, when a distributional score for text fragments is unavailable using (37), (for example, when the text fragment occurs only once in the corpus), an alternative distributional score is returned based on the average of the distributional scores between the SL and TL lexical items of which the text fragments are composed. The algorithm for computing this alternative distributional score given an SL phrase and an TL phrase as input proceeds as follows: the distributional score between each SL word contained within an SL fragment and each TL word contained within an TL fragment is computed. The score is calculated using (37), but is modified so that *S* and *T* represent single lexical items. For each SL word, the highest distributional score for each TL word is recorded. Finally, a score indicating the probability that an SL fragment and TL fragment are translation equivalents is returned by dividing twice the sum of all the highest distributional scores divided by the sum of the lengths (in words) of the SL and TL fragments. The algorithm is detailed in Figure 28.

```
For each SL Word W_S ∈ SLFragment
   For each TL Word W_T ∈ TLFragment
      Add Dice(W_S,W_T) to DistributionScores list
   Add Maximum(DistributionScores) to MaxScores list
Return 2*(∑ MaxScores/(Length(SLPhrase) + Length(TLPhrase)))
```

Figure 28: Algorithm for the Calculation of Alternative Distribution Score

Determining translation equivalents by comparing distributions is not a perfect measure and suffers from certain well-known problems. One such problem is data-sparseness. Results are excluded where text fragments appear only once in the corpus. Furthermore, distributional scores frequently identify *translingual collocates* as opposed to true translation equivalents. For example, the results of the K-Vec vocabulary extraction experiments between English and French (Fung & Church, 1994: 1099) produced alignments between *nuclear* and both *croisière* (*cruise*) and *essais* (*tests*), rather than with *nucléaire*. A SL lexical item may also be translated by more than one morphological variant of an TL lexical item. Unless lexical items are grouped under the same lemma, the distributions between SL and TL morphological variants may be too

different to determine bilingual relationships. The inclusion of cognates into the bilingual similarity metric provides an alternative score that not only provides a 'back-up' score if the distribution score fails or is not available, but also reinforces or 'smoothes' the score if it provides a good or non-representative result respectively. With two different types of score, there is a greater chance of returning a more accurate or representative score of translation equivalence.

A score based on cognates is particularly useful in instances of sparse-data so that text fragments containing words of low frequency are easily aligned. However, the cognate score may not be appropriate when aligning fragments between distant language pairs that either do not share the same writing system (in which case a transliteration scheme may be necessary) or do not share many cognates (of a historical or loan-word nature). Furthermore, there is the chance that orthographically similar words between languages are unrelated (*faux amis*). In such cases, a score including bilingual lexical distribution is useful.

The cognate score is based on how many cognates the SL and TL text fragments share. Each word in the SL text fragment w_S is compared with each word in the TL fragment w_T to test whether they are cognates. The score used to determine whether two words w_S and w_T are cognates is Levenshtein Distance (LD). LD is defined as the minimum number of substitutions, deletions and insertions necessary to convert one string into another, as computed by DP. To return a similarity score, that is a real value between zero and one indicating the probability that w_S and w_T are cognates, the LD score is normalised over the maximum possible distance between the two strings (38). The maximum distance is defined as the sum of the cost of the maximum number of substitutions plus the sum of the cost of any remaining deletions or insertions (39), where $1 \leq i \leq |w_S|$ and $1 \leq j \leq |w_T|$. If the cost of substitutions, deletions and insertions are all equal to 1 (40), the maximum distance is simply the length of the longest of the two strings, *Max(i,j)*. If the similarity score (38) is above an experimentally determined threshold, the two strings w_S and w_T are considered to be cognates. Given an SL and TL text fragment, the number of cognates that the SL and TL text fragments share is returned.

(38) $$SimScore = 1 - \frac{LD(w_S, w_T)}{MaxDist(w_S, w_T)}$$

(39) $$MaxDist(i, j) = Max\begin{cases} D(i-1, j) + d(x_i, 0) \\ D(i, j-1) + d(0, y_j) \\ D(i-1, j-1) + d(x_i, y_j) \end{cases} = Max(i, j)$$

(40) $$d(x_i, 0) = 1$$
$$d(0, y_j) = 1$$
$$d(x_i, y_j) = \begin{cases} 1 \text{ if } x_i \neq y_j \\ 0 \text{ if } x_i = y_j \end{cases}$$

Strings below five characters in length and function words – as determined by a manually created stop-list – are not considered for cognate matching. Also, for efficiency reasons, strings that are identical are automatically considered to be cognates. In an effort to increase the similarity score between words that are obviously related, accented characters are normalised to their unaccented equivalents. For example, when comparing *television* with *télévision*, normalising *é* to *e* returns a much higher similarity score. This enables the threshold to be set higher, thus weeding out relatively high-scoring non-cognate word pairs. In the English-French experiments undertaken this worked well, but it may not be appropriate for other language pairs.

The above description of the bilingual similarity metric has pointed out that where text fragments contain one or more instances of closed class words, they are effectively removed from the text fragment in the alignment process. Alignment of text fragments is therefore dependent only on the open-class words that they contain. The alignment of closed class words is problematic. It is neither possible to align them by means of bilingual lexical distribution nor by cognate matching. Providing a manually created list of bilingual word pairings for function words is also problematic since it is difficult to

equate them with any degree of accuracy. They also have different functions depending on the context in which they are found.

In the event that a text fragment is composed uniquely of closed class words, its alignment requires an alternative solution to cognate matching or a distributional score. Two methods are proposed. First, text fragments containing only function words are not aligned. They are effectively ignored as being part of the sequences of text fragments. Second, in order to remove the possibility that text fragments composed uniquely of closed-class words exist, the variables representing the text fragments preceding and subsequent to such fragments are concatenated to form one larger text fragment. Both solutions are implemented and both have disadvantages.

(41) depicts a translation pattern with an SL and an TL text fragment composed uniquely of closed class words (*and the* and *et aux*). The first solution excludes them from the alignment process. Therefore, in (41), only the SL fragment *Emergency aid to the* is aligned with the TL fragment *Aide d'urgence à la*. No further fragment alignments are made. The consequence of this method is that the complement translation pattern of (41) will have variable positions that are unaligned, reducing its flexibility in the recombination phase (section 3.3).

(41) Emergency aid to the (…) and the (…) ←→
Aide d'urgence à la (…) et aux (…)

(42) Emergency aid to the Russian Federation and the former republics ←→
Aide d'urgence à la Fédération Russe et aux anciennes républiques

The second solution 'fills in' the variable positions appearing before and after the fragments composed uniquely of function words. The translation pattern (41) therefore becomes (42). The consequence of this second method is that the text fragments become much longer, reducing the flexibility of the translation patterns in recombination. In the example in (42), the text fragments span the entire length of the sentence pair from which the translation pattern was extracted.

3.2.3.2 Sequence Comparison Algorithm

Given two sequences *x* and *y* of lengths *m* and *n* respectively, the alignment of *x* and *y* is initially carried out by DP. The DP equation is given in (43). Let x_i be one of the SL text fragments and y_j be one of the equivalent TL fragments. Let the function *D(i,j)* be the minimum edit distance between the text fragments $x_1 \ldots x_i$ and their translations $y_1 \ldots y_j$ where $1 \leq i \leq m$ and $1 \leq j \leq n$. The minimum edit distance of the sequences *x* and *y* is therefore *D(m,n)*. Let *d* represent the distance function, which is in fact the inverse of the bilingual similarity metric detailed in section 3.2.3.1. In the order listed in (43), the terms in the equation correspond to insertion (1:0 relationship), deletion (1:0), substitution (1:1), expansion (1:2 and 1:3), compression (2:1 and 3:1) and finally substitution of *adjacent* characters or *swaps* (Lowrance & Wagner 1975).

$$D(i,j) = \min \begin{cases} D(i, j-1) + d(0, y_j) \\ D(i-1, j) + d(x_i, 0) \\ D(i-1, j-1) + d(x_i, y_j) \\ D(i-1, j-2) + d\left(x_i, \{y_{j-1}, y_j\}\right) \\ D(i-1, j-3) + d\left(x_i, \{y_{j-2}, y_{j-1}, y_j\}\right) \\ D(i-2, j-1) + d\left(\{x_{i-1}, x_i\}, y_j\right) \\ D(i-3, j-1) + d\left(\{x_{i-2}, x_{i-1}, x_i\}, y_j\right) \\ D(i-2, j-2) + d\left(x_i, y_{j-1}\right) + d\left(x_{i-1}, y_j\right) \end{cases} \tag{43}$$

The values of *D(i,j)* at the *m* x *n* array boundaries are set up as follows: For the 0^{th} column (row) of the array, the value of *D(i,0)* (*D(0,j)*) is made equal to the sum of the costs of deleting (inserting) the first *i* (*j*) symbols of *x* (*y*). This is formalised in (44).

$$\begin{aligned} & D(0,0) = 0 \\ & D(i,0) = \sum_{k=1}^{i} d(x_k, 0) \quad \text{for} \quad 1 \leq i \leq m \\ & D(0, j) = \sum_{k=1}^{j} d(0, y_k) \quad \text{for} \quad 1 \leq j \leq n \end{aligned} \tag{44}$$

The DP algorithm consists of two embedded loops and is clearly of a time and space complexity in the order of $\mathcal{O}(mn)$, where *m* and *n* are the lengths of the sequences to be aligned. This makes it practical in terms of time and space but it is not fast. In fact, it is the part of the translation pattern extraction phase that incurs the largest time processing overhead, as the number of patterns extracted can be large (3.4). The overhead is kept at a minimum by the fact that the values of *m* and *n* are rarely large (cf. DP for file comparison) and by ensuring a minimum of terms in the DP equation. By keeping a record of the application of the DP algorithm, a 'trace' of the alignment is returned i.e. the pairings of the two sequences of SL and TL text fragments that make up the global alignment. The DP algorithm assumes that all alignments are adjacent

The grain-size of the alignment types in the DP equation (43) has consequence for complexity and the recombination phase. More alignment types, such as 2:2, 2:3 and 3:2 etc. could easily have been included as terms in the DP equation. Although increasing the number of terms in the equation increases the processing overhead, increasing the grain size of the alignment types reduces the likelihood of long distance dependencies occurring. A further issue when considering the grain size of the alignments is the classic trade-off between accuracy and flexibility in recombination. When matching an SL input against translation patterns, coarse-grained alignment types, such as 2:3 etc, provide longer matches. Longer matches are preferred in that they provide more context and reduce ambiguity. Therefore, the likelihood of a correct TL translation string increases. However, there is less chance of matching an SL input against longer fragments, therefore flexibility suffers. Moreover, matches against smaller fragments involve a greater amount of translation ambiguity, passage and boundary friction. This is summarised by the observations of Nirenburg et al. (1993):

> The longer the matched passages, the lower the probability of a complete match (…). The shorter the passages, the greater the probability of ambiguity (one and the same *S′* can correspond to more than one passage *T′*) and the greater the danger that the resulting translation will be of low quality, due to passage and boundary friction and incorrect chunking. (Nirenburg et al. 1993: 48).

Fine (1:2) as opposed to coarse (2:3) grained local alignment types are favoured in this approach in order to maintain flexibility and recall.

The DP equation is only a partial solution in that it is only capable of computing alignments between adjacent text fragments. In fact, the very nature of the DP algorithm precludes the computation of non-adjacent alignments. There is a very simple relationship between the terms of the two functions *D* and *d* in the equation which requires them to have neighbouring values (*i* and *i-1*, *i-1* and *i-2* etc). The ability to compute non-adjacent alignments, which highlight long-distance dependencies, is crucial when aligning text fragments between structurally divergent languages and languages with differing word orders. It is even necessary, although to a lesser extent, when aligning text fragments between two closely related languages.

The second pass algorithm, which computes the (possible empty) set of non-adjacent alignments, is applied subsequent to the DP algorithm. Its input is the same two sequences of text fragments *x* and *y* of lengths *m* and *n* respectively. A further input parameter is the set of adjacent alignments as computed by the DP algorithm. The second-pass algorithm is summarised as follows: each SL fragment x_i for $1 \leq i \leq m$ is tested for alignment with each TL fragment y_j for $1 \leq j \leq n$. For each potential alignment pair $<x_i, y_j>$, the bilingual similarity score $Score(x_i, y_j)$ is computed. If two conditions are satisfied, the text fragments x_i and y_j along with $Score(x_i, y_j)$ are recorded as a triple and added to a list of candidate non-adjacent alignments (Figure 29). The first condition states that the alignment must be high-scoring. This is defined as $Score(x_i, y_j)$ being above a predefined threshold. The second condition states that the alignment must be non-adjacent. This is enforced by the condition where the absolute difference (*abs*) between *i* and *j* must be greater than 1.

Once a list of triples representing candidate non-adjacent alignments is 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 non-adjacent alignment. Second, if the proposed alignment $<\alpha,\beta>$, is a subset of an *adjacent* alignment <*A,B*> that has previously been computed by the initial application of the DP algorithm (where *A* and *B* represent pointers

to SL and TL text fragments respectively and where $\alpha \in A$ and $\beta \in B$), they are not considered further. The reason for this is simply that the DP algorithm has either found the same alignment as the second pass algorithm or that an alignment other than a 1:1 type is more probable, thus the proposed alignment is not likely to improve the final global alignment.

```
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 29: Algorithm to Compute Candidate Non-Adjacent Alignments

The final condition stipulates that the score of the proposed non-adjacent alignment $<\alpha,\beta>$ must be greater than both the two individual scores of the two adjacent alignments computed by DP of which α and β are a member. In more detail, each adjacent alignment computed by the initial application of the DP algorithm is represented as a pair $<A,B>$ where A and B represent pointers to zero or more SL and TL text fragments respectively according to the local alignment patterns stipulated by the DP equation (43). The initial global alignment computed solely by DP is a set containing k ($k>0$) local alignments $\{<A,B>_1, <A,B>_2,\ldots<A,B>_k\}$. The proposed non-adjacent alignment is also represented by a pair $<\alpha,\beta>$, but α and β represent one and only one SL and TL text fragment respectively. This final condition is summarised in (45)[5] which states that the bilingual similarity score of the proposed non-adjacent alignment $Score(\alpha,\beta)$ must be greater than the scores of the two adjacent alignments, $Score(<A,B>_p)$ and $Score(<A,B>_q)$, of which α and β are a member respectively, such that $\alpha \in A_p$ and $\beta \in B_q$. The two adjacent alignments $<A,B>_p$ and $<A,B>_q$ must also be two distinct elements that form part of the list of local alignments that make up the global alignment as computed by DP such that $p \neq q$. The intended effect, if the condition is

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

satisfied, is that the addition of the non-adjacent alignment *improves* the likelihood that the final global alignment provides the optimal alignment of the two sequences.

$$
\begin{aligned}
& Score\,(\alpha,\beta) > Score\,(\langle A,B\rangle_p)\,\& \\
& Score\,(\alpha,\beta) > Score\,(\langle A,B\rangle_q), \\
& \alpha \in A_p,\ \beta \in B_q, 1 \le p \le k, 1 \le q \le k,\ p \ne q
\end{aligned}
\qquad (45)
$$

In order to ensure that the alignment process executes within a practical asymptotic running time, only non-adjacent alignments of a bijective (1:1) nature are considered in the second pass. The two-embedded loops within the algorithm (Figure 29) ensures that it runs within a worst case complexity of the order $\mathcal{O}(mn)$ and is therefore no worse than DP. The obvious disadvantage is that the algorithm 'misses' non-bijective non-adjacent alignments. However, the most common alignments are bijective in nature (Table 2).

To illustrate the effect of the second-pass algorithm, the translation pattern in Figure 30 is first aligned using just the DP algorithm. The alignment of the text fragments (boxed text) is incorrect. The result of the application of the second-pass algorithm is the computation of the non-adjacent alignment: *project-projet*. Subsequent application of the DP algorithm to the remainder of the text fragments excluding *project* and *projet*, produces the correspondences *maternal-maternelle* and *neonatal-néonatals*. The alignments computed by the second-pass algorithm and the second application of DP are concatenated to return the final global alignment.

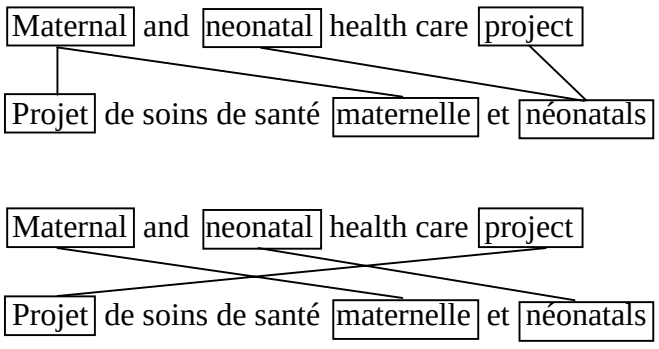

Figure 30: Before and After Application of Second-Pass Algorithm

3.3 Recombination

The input to the recombination phase is an SL input sentence and the set of translation patterns extracted from the corpus. The output is one or more TL translation strings, the best of which is selected according to a score indicating translation confidence (section 3.3.5).

Translation patterns that share lexical items with the SL input and partially (or fully) *cover* it are retrieved in a pattern matching procedure. 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 provided by the base pattern, the more context and 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 segments are bound to the variables on the SL side of the base pattern. The translations of the SL segments bound to the SL variables of the base pattern are retrieved from the remaining set of translation patterns. As the text fragments and variables are aligned, the TL equivalents are retrieved trivially and bound to the relevant variables on the TL side of the base pattern to from translation strings.

The following is a simple example. Given the SL input (46), suppose the longest covering base pattern is (47). To complete the match with the SL side of (47), a translation pattern containing the text fragment *Ethiopia* is required (48). The TL translation (49) is generated by recombining the text fragments: *Ethiopia* and *Etiopía* are aligned in (48) as are the variables in the base pattern (47). Since *Ethiopia* and *Etiopía* are aligned on a 1:1 basis and so are the variables in the base pattern (47), the TL text fragment *Etiopía* is bound to the variable on the TL side of (47) to produce (49).

(46) AIDS control programme for Ethiopia

(47) AIDS control programme for (…) ←→ Programa contra el SIDA para (…)

(48) (…) Ethiopia ←→ (…) Etiopía

(49) Programa contra el SIDA para Etiopía

As pointed out in section 3.2.3, the local alignments may be of a type other than a simple 1:1 relationship. Therefore, it is clearer to think of the algorithm proceeding in terms of matching successive alignments of the variables in the base pattern rather than matching successive SL variables individually. Matches are based not only on string equality but also on matching the alignment type.

3.3.1 Pattern Retrieval

The first stage in the recombination process is the retrieval of translation patterns that serve as base patterns. Candidate translation patterns that share lexical items with the SL input are retrieved and successively filtered until they meet the criteria that define base patterns. A base pattern is defined as a translation pattern that either fully or partially covers the SL input. In more detail:

- The SL side of a base pattern must be less than or equal in length (measured in words) to the SL input
- The SL side of a base pattern consists solely of lexical items of which the SL input is composed
- The lexical items in the SL side of the base pattern appear with the same frequencies as they do in the SL input
- When the variables of the SL side of the base pattern have been instantiated with the string values of the unmatched segments of the SL input, that sequence of lexical items must match the SL input exactly.

The following illustrates the concept of a base pattern. Assuming that the SL input is (50), translation pattern (51) is a valid base pattern. It fulfils the above criteria so that when the variable on the SL side of the pattern is instantiated with the string value *C D*, the SL side of the pattern matches the SL input exactly. (52) is not a valid base pattern as it is longer than the SL input and contains lexical items that do not appear in the SL input. (53) cannot be a base pattern as it contains an instance of a lexical item with a greater frequency than that found in the SL input, therefore it cannot contribute to match the SL input (50).

(50) A B C D E

(51) A B (…) E ←→ A′ B′ (…) E′

(52) A B (…) E F G H ←→ A′ B′ (…) E′ F′ G′ H′

(53) A B (…) B ←→ A′ B′ (…) B′

The first stage of the pattern retrieval process is relatively simple and consists of retrieving *candidate* translation patterns which have the potential to be base patterns. For each translation pattern *P*, if its SL side P_{SL} shares one or more lexical items with the SL input, it is added to a list of candidate patterns. This procedure is detailed in Figure 31.

```
RetrieveCandidatePatterns() -> List of Translation Patterns
For each Word w in SL Input
   For each Translation Pattern P
      If P_SL Contains w
         Add P to list of CandidateTranslationPatterns
Return CandidateTranslationPatterns
```

Figure 31: Procedure to Retrieve Candidate Translation Patterns

From an implementation point of view, this iterative procedure is made more efficient by the use of indexing each SL word in the corpus to the list of one or more translation patterns, or their indexes, that contain it. The resulting inverted file (Frakes & Baeza-Yates, 1992) is depicted in Figure 32 as a hash table. The retrieval algorithm in Figure 31 is therefore modified so that for each word in the SL input, the translation patterns whose SL side contains any of its lexical items are retrieved efficiently without the need for iterating through a potentially large list of translation patterns.

Poliomyelitis	→	$\{P_1, P_2, P_6, P_{126}, \ldots\}$
Epidemic	→	$\{P_4, P_5, P_{56}, P_{678}, \ldots\}$
Programme	→	$\{P_{45}, P_{567}, P_{789}, \ldots\}$
Control	→	$\{P_{23}, P_{46}, P_{345}, \ldots\}$
…		…

Figure 32: Inverted file Mapping SL Words to Translation Patterns or their Indexes

The list of candidate translation patterns (that share one or more lexical items with the SL input) is further filtered according to the criteria defined in the list above. For each candidate translation pattern returned by the algorithm in Figure 31, three tests are applied before it is added to a list of filtered patterns and returned. The three tests are, in the order shown in Figure 33, that the length of the SL side of the base pattern P_{SL} is less than or equal to the length of the SL input (measured in words), that the SL side of the base pattern only contains lexical items that appear in the SL input and that the words in the SL side of the base pattern occur with identical frequencies as they do in the SL input.

```
FilterPatterns(Patterns, SLInput) -> List of Patterns
For each Translation Pattern P
   If Length(P_SL) ≤ Length(SLInput)
      If P_SL ContainsOnlyLexicalItemsP(SLInput)
         If WordFrequencyEqualP(P_SL, SLInput)
            Add P to list of FilteredPatterns
Return FilteredPatterns
```

Figure 33: Method to Filter the Candidate Translation Patterns

The list of filtered patterns undergoes a test before it is considered to be a valid base pattern. As outlined above, when the variables on the SL side of the base pattern are instantiated with the string values of the unmatched segments of the SL input, that sequence of SL lexical items must match the SL input exactly. The SL input is matched, word by word, successively against the SL side of the base pattern. Where the two sequences fail to match, and until they start to match once more, the sequence of unmatched lexical items is bound to the current variable on the SL side of the base pattern. This continues until the end of the two sequences is reached. As an example, consider the SL input (50) and the SL side of the base pattern (51). Each successive lexical item of (50) is matched against each successive lexical item in the SL side of (51). *A* is matched against *A* and *B* is matched against *B* until *C* is reached. Neither *C* nor *D* can be matched against the next SL lexical item in (51) which is *E*. Therefore *C* and *D* are recorded as the value bound to the first variable on the SL side of the base pattern. Finally, the lexical element *E* in the SL input is matched against *E* on the SL side of (51). As the end of the SL input has been reached, as has the end of the SL side of the base pattern, the process terminates. The variable on the SL side of the base pattern (51) is

instantiated with the value *C D* to make the sequence *A B C D E*. This matches the sequence of lexical items in the SL input exactly. Therefore, the translation pattern (51) is considered as a valid base pattern and the recombination process proceeds.

3.3.2 Core Recombination Method

Where more than one base pattern has been identified, they are ranked according to the extent to which they cover the SL input. The ratio of cover is defined simply as the length of the SL side of the base pattern over the length of the SL input measured in words (54). The translation process subsequently proceeds by considering the base patterns with the highest ratio of cover first. Where TL translations are produced, the remaining base patterns are not considered further – unless they share the same ratio of cover – since the longest patterns are preferred. Longer patterns provide more context, reduce ambiguity and are more likely to produce correct translations.

(54) $$Cover = \frac{Length(Pattern_{SL})}{Length(SLInput)}$$

Given an SL input and a base pattern, the core recombination algorithm proceeds by instantiating the variables on the SL side of the base pattern with the string values of the unmatched segments of the SL input. This is carried out using the algorithm outlined in 3.3.1. Once the variables are instantiated, the TL equivalents of those string values are retrieved from the remaining set of translation patterns and inserted into the appropriate variables on the TL side of the base pattern forming an TL string. On account of translation ambiguity, there may be more than one TL equivalent for each SL fragment - or rather series of fragments as they may be aligned say 2:1, as in Figure 1, rather than on a simple bijective (1:1) basis. The recombination algorithm must take into account all possible translations of the SL fragments and consequently may form multiple translation solutions. Selection of the best translation solution is made only after all solutions have been produced (3.3.5).

As an example, consider the SL input in (50) and assume that the longest covering base pattern is (55). The variables on the SL side of the base pattern are instantiated with the values *B* and *D* to produce the sequence *A B C D E* matching the SL input. The

remaining set of translation patterns are subsequently searched for instances of the SL text fragments *B* and *D*. Supposing that the variables in (55) are aligned on a 1:1 basis and are aligned in succession i.e. the first variable on the SL side is aligned with the first variable on the TL side etc, then the instances of *B* and *D* in the remaining translation patterns must also be aligned with their TL equivalents on a 1:1 basis. (56) is an example of a translation pattern where *B* is aligned to an TL text fragment *B′* on a 1:1 basis.

(55) A (…) C (…) E ←→ A′ (…) C′ (…) E′

(56) (…) B (…) ←→ (…) B′ (…)

All remaining translation patterns containing the SL text fragments *B* and *D* that are aligned on a 1:1 basis are retrieved. On account of the fact that the text fragments in the translation patterns are aligned, the TL equivalents of *B* and *D* are retrieved trivially. From the set of translation patterns that contain *B* or *D*, the list of all TL equivalents is constructed. Suppose that all the TL equivalents retrieved for *B* was the list *{B′, X′}* and that the list of all TL equivalents for *D* was the list *{D′, Y′, Z′}*, each of these TL fragments must be considered with equal importance when recombining, as no information is present to favour one TL equivalent over the other. Taking into account each TL equivalent of *B* and *D*, six translation solutions are produced (57). Each solution is assigned a score indicating the confidence of the translation (3.3.5), the best of which is chosen as the final solution.

(57) *A′ B′ C′ D′ E′*
A′ X′ C′ D′ E′
A′ B′ C′ Y′ E′
A′ X′ C′ Y′ E′
A′ B′ C′ Z′ E′
A′ X′ C′ Z′ E′

On account of the possibility of non-bijective alignments (2:1, 1:2 etc), the algorithm for retrieving the TL equivalents of SL text fragments bound to the variables on the SL side of the base pattern proceeds in terms of matching successive *alignments* of

the variables of the base pattern, rather than matching successive variables. For example, given the SL input in (58), suppose the base pattern were (59) where *A* is aligned with *A′* and *B* with *B′* on a 1:1 basis respectively. The two SL variables in (59) are aligned on a 2:1 basis with the single TL variable. Therefore, in order to complete the match with the SL input, the translation patterns are searched for instances containing the text fragments *X* and *Y* aligned with a single TL text fragment on a 2:1 basis. The translation pattern (60) is one example. This allows recombination to proceed to produce the TL string *A′ Z′ B′*.

(58) A X B Y

(59) A (…) B (…) ←→ A′ (…) B′

(60) (…) X (…) Y ←→ (…) Z′ (…)

The algorithm for matching SL text fragments in the set of translation patterns and retrieving their TL equivalents is shown in Figure 34. For each local alignment between the base pattern variables, the SL text fragments associated with the SL variable(s) are retrieved. Then for each translation pattern *P*, if *P* contains an instance of the SL text fragment(s) and *crucially* if it is aligned in exactly the same manner as it is in the base pattern, then the TL fragments aligned with the SL fragment(s) are returned and added to a list of equivalents. This is carried out for all variable alignments of the base pattern so that a list of all equivalents for all SL fragments bound to the SL base pattern variables is retrieved. The SL text fragments bound to the SL side of a base pattern are matched not only in terms of string value but also alignment type.

```
RetrieveAllTLEquivalents( BasePattern ) → List of TL Equivalents
   For each Alignment of SL and TL Base Pattern Variables A
      Retrieve SL Fragments F_SL bound to SL variables given by A
      For each Translation Pattern P
         If MatchP(F_SL, A, P)
            TL text fragments F_TL = RetrieveTLFragments(F_SL, A, P )
            Add F_TL to list of TL equivalents, Equivs_TL
      Add Equivs_TL to list of All TL Equivalents, AllEquivs_TL
   Return AllEquivs_TL
```

Figure 34: Algorithm to Retrieve all TL Text Fragments

To increase the time efficiency of the algorithm to retrieve TL translation equivalents from the translation patterns, an inverted file is set up indexing SL fragments from the set of translation patterns to their TL equivalents, according to the alignments in the translation patterns. Hashing enables the rapid retrieval of TL text fragments, rather than iterating through a potentially large list of translation patterns. A depiction of the inverted file is given in Figure 35. The inverted file distinguishes the alignment types along with the bilingual relationships between the string values. For example, the SL fragment *B* is mapped onto the equivalent TL fragments *B′*, *X′* etc on a 1:1 basis. Similarly, the SL fragment *D* is mapped onto any of *D′*, *Y′* and *Z′* etc on a 1:1 basis. However, the SL fragments *(X,Y)* are mapped onto the single TL fragment *Z′* on a 2:1 basis. The final line in the diagram depicts how a single SL fragment can be mapped onto a series of TL fragments on both a 1:2 and 1:1 basis (and more alignment types if required).

```
(B)    →    {(B′), (X′), …}
(D)    →    {(D′), (Y′), (Z′), …}
(X,Y)  →    {(Z′), …}
(V)    →    {(V′,W′), (U′,V′), (U′)…}
…           …
```

Figure 35: Inverted File Mapping SL Text Fragments to their TL Equivalents

Once all TL equivalents for all SL variable positions in the base pattern have been retrieved, a recursive algorithm is applied to consider each possible TL equivalent – or set of equivalents in the case of non-bijective alignments – when forming TL translation strings (Figure 36). It is on account of this translation ambiguity that multiple translation solutions are produced. The algorithm `MakeTranslation()` takes as input parameters, the base pattern, the alignments of the variables in the base pattern, the set of all TL equivalents given the SL text fragments bound to the SL base pattern variables (`ALLTLEquivs`) and an initially empty list in which to store all translation solutions (`Translations`). The algorithm proceeds by iterating through each local alignment – taking the head and tail of the list – and retrieving the equivalent TL fragments for that alignment (`TLEquivalents`). Subsequently, each TL fragment within that set (or set of

fragments if the current alignment represents a non-bijective alignment) is considered (`TLEquiv`) and placed in the appropriate position in an array representing the TL variables of the base pattern (`TLBaseVariables`). `MakeTranslation()` is called recursively in order to iterate through the list of alignments. When the end of the list of alignments is reached (terminating condition), each TL variable position in the base pattern is effectively filled, so that a translation string is produced. The function `BuildTLString()`, as its name suggest, builds an TL string from the TL side of the base pattern and the array of instantiated TL base pattern variables and stores it in the global variable `Translations`.

```
MakeTranslation(BasePattern,Alignments,AllTLEquivs,
                TLBaseVariables)       → void
   If Size( Alignments ) < 1
      BuildTLString( BasePattern, TLBaseVariables, Translations )
      Return
   Else
      TLEquivalents = GetTLEquivalents(Head(Alignments), AllTLEquivs)
      For Each TLEquivalent TLEquiv
         Instantiate( TLBaseVariables, TLEquiv )
         MakeTranslation(BasePattern,Tail(Alignments),AllTLEquivs,
                         TLBaseVariables)
```

Figure 36: Recursive Algorithm to Build All Translation Solutions

In order to finalise the TL translation strings produced, the inverse of the tokenisation procedure, *reassembling*, is applied. Instead of separating tokens from a string of words and white-space characters, the sequence of tokens that make up the translations are separated by white-space characters. Additionally, in the case of the reassembly of French tokens, the same information included in tokenisation regarding pronouns and articles that join subsequent vowel-initial words by replacing an *e* with an apostrophe is included. For example, when an article such as *le* is followed by a vowel-initial word such as *ordinateur*, they are condensed into one token *l'ordinateur*.

3.3.3 Recursive Matching

In the core recombination phase, *direct* matches are sought between SL fragments bound to the SL base pattern variables and SL fragments in the remaining set of translation patterns. It is possible that there are no SL text fragments in the remaining set of

translation patterns that directly match SL text fragments bound to the SL variables of the base pattern. In such cases, recombination cannot proceed to produce full translations as TL variables on the TL side of the base pattern will remain uninstantiated. One solution to this problem is the provision of more training data in the translation pattern extraction phase (section 3.2). This would provide a more wide-coverage lexicon of phrasal translations. An alternative solution involves searching for 'indirect' matches among the set of SL text fragments in the translation patterns using recursive matching. This method attempts to match an SL text fragment compositionally against *multiple* SL fragments from the set of translation patterns. A recursive algorithm is invoked that attempts to match successively shorter portions of an SL fragment, from left to right, against the SL fragments in the translation patterns. The TL equivalents are concatenated naively, according to the order of the matches with the portions of the SL fragment, to provide an TL equivalent for the whole SL fragment. Only SL text fragments aligned on a bijective or 1:1 basis are considered.

As an example, given the SL input *A B C D E*, assume that the base pattern is (61) and the remaining translation patterns are (62) and (63). In order to complete the match with the base pattern, the SL variable is instantiated with the value *A B C*. As there are no translation patterns containing the text fragment A B C, the recursive matching algorithm is invoked. The longest match sequence possible, from left to right, is sought at each iteration. First, a match is sought for *A B C*. As there are no translation patterns containing *A B C*, the next longest sequence *A B* is sought.[6] Again, as no match is found, the next longest sequence *A* is sought. As there is a match with *A* in (62), its TL equivalent is retrieved and placed on a list [A′] indicating the TL equivalent so far for the SL fragment *A B C*. As *A* has been matched, a match with the remaining sequence of SL items *B C* is sought using the same left-to-right recursive algorithm. A match is found in (63) and its TL equivalent is added to the result vector to produce [A′,B′,C′]. As the entire sequence of SL items has been matched, recursion terminates returning *A′ B′ C′* as the TL equivalent of *A B C*.

[6] As the algorithm only proceeds from left to right, no match is sought for *B C* at this point.

(61) (…) D E ←→ (…) D′ E′

(62) A (…) ←→ A′ (…)

(63) (…) B C (…) ←→ (…) B′ C′ (…)

The recursive matching algorithm is outlined in Figure 37. The arguments to the procedure include the SL fragment to be matched F_{SL}, the original string value of the SL fragment to be matched OF_{SL},[7] the set of remaining translation patterns `Patterns`, and the variable F_{TL}, to be instantiated with the string value of the final TL equivalent of F_{SL}. The final argument MF_{SL} is a variable containing the portion of F_{SL} that has been matched at each iteration. It is recorded in order to ascertain, when `MatchRecur()` terminates and returns to its calling environment, whether F_{SL} has been entirely matched by SL fragments from the translation patterns. This avoids partial matches being considered.

The terminating condition of the algorithm states that if the end of F_{SL} has been reached, there is no more iteration, whether all or part of F_{SL} has been matched. When the terminating condition does not apply, the main body of the function is considered. A match with F_{SL} is sought among the SL fragments in the translation patterns and, if successful, stored in `TLEquiv`. Where no match is found `MatchRecur()` is called recursively on the next longest sequence of F_{SL} where its final item is removed. On the other hand, where a match was successful, the TL equivalent is added to the list F_{TL} which stores the eventual entire TL equivalent of F_{SL}. The portion of F_{SL} that has been successfully matched is stored in MF_{SL} as explained above. Finally, `MatchRecur()` is called recursively by considering the remaining sequence of OF_{SL} that has yet to be matched. This is defined as the difference between the matched sequence F_{SL} and the original value of the SL text fragment to be matched OF_{SL}.

In actual fact, the function `GetMatch()` retrieves all translation equivalents of F_{SL}, and stores them in `TLEquiv` and eventually F_{TL}, as there may be more than one match in the set of translation patterns. The algorithm in Figure 37 is simplified for clarity. Given multiple matches for F_{SL}, each combination of TL equivalents must be considered as a solution. This concept is analogous to that used in recombination where all TL

[7] Initially, F_{SL} and OF_{SL} are identical in value

equivalents for a given alignment are considered as translation solutions with equal status. Consequently, the same recursive algorithm (Figure 36) is used.

```
MatchRecur(F_SL,OF_SL,Patterns,F_TL,MF_SL)→ void
If Size(F_SL) < 1
   Return
Else
   TLEquiv = GetMatch(F_SL, Patterns)
   If Size(TLEquiv) < 1
      MatchRecur(RemoveFinalItem(F_SL), OF_SL, Patterns, F_TL,MF_SL)
   Else
      Add(TLEquiv, F_TL)
      Add(F_SL,MF_SL)
      MatchRecur(Difference(F_SL,OF_SL), OF_SL, Patterns, F_TL, MF_SL)
```

Figure 37: Recursive Matching Algorithm

3.3.4 Partial Translations

Where an SL input cannot be fully matched against the set of translation patterns, it is possible to compute partial translations. In more detail, partial translations occur when one or more SL fragments bound to the variables on the SL side of the base pattern cannot be matched by SL text fragments in the remaining set of translation patterns, whether by direct (section 3.3.2) or recursive (section 3.3.3) matching. This results in uninstantiated variables on the TL side of the base pattern.

As an example, consider the SL input *A B C D E* and assume that the longest covering base pattern is (55). This result is that *B* is bound to the first SL base pattern variable and *D* to the final variable. Assuming that the only other relevant translation pattern is (56), the TL equivalent of *B* is retrieved and bound to the appropriate variable on the TL side of the base pattern, whereas no match is found for *D*. The result is that the final TL base pattern variable in (55) is left uninstantiated. If partial matches are considered acceptable, the translation solution becomes *A′ B′ C′ (...) E′*, where the ellipses denote a series of one or more missing TL words.

3.3.5 Translation Confidence Score

On account of translation ambiguity i.e. the fact that one SL fragment may have more than one TL equivalent, recombination frequently returns more than one translation

solution for each SL input. In such cases, the list of competing translation solutions are ranked best-first according to a score assigned to each translation solution based on parameters to measure the confidence of each translation (64). The score is based on three components. The first is a bigram model of the TL sentences in the corpus (*BG*). This model measures the probability that the word sequence of the translation is a valid word sequence of the TL. The second component (*BF*) measures the amount of boundary friction between each TL fragment bound to an TL variable in the base pattern and the previous and subsequent fragments in the base pattern. The third component (*BS*) returns the bilingual similarity or probability that the TL fragments instantiated to the TL base pattern variables are translations of the SL fragments bound to the SL base pattern variables. Finally, a penalty *p* is applied if any of the TL variables of the TL base pattern remain uninstantiated. All three components, including *p* return a real value between zero and 1. Weights – ω_1, ω_2, ω_3 – are applied to each of the three components of the score.

$$Confidence = \left(\omega_1\left(BG\right)+\omega_2\left(BF\right)+\omega_3\left(BS\right)\right)\left(1-p\right) \tag{64}$$

Measuring the probability that an TL translation string is a valid sequence of TL words according to a bigram model of the TL is a technique originally borrowed from the Speech Recognition field (Katz, 1987). Given a set of analyses for a given utterance, an *n*-gram model is invoked to determine the most likely word form or word sequence. The same technique has been used in SMT (Brown et al. 1993) where the most likely TL word sequence among a set of competing TL translations is selected. It has also been used for the same purpose in the multi-engine Pangloss Mark III MT experiments of Brown & Frederking (1995). The translations of fragments of the SL input from each of the three MT engines are placed on a chart. The best path through the chart is determined by a trigram model (with back-off to bigram and unigram models) to produce the most likely translation. The same principle is applied when computing the term *BG* in the translation confidence score (64). A bigram model is used to measure the probability that each translation solution is a valid sequence of words in the TL. The sequence assigned the highest probability is, at this stage, the most likely translation for the given SL input.

The bigram model is computed from the TL sentences of the bilingual corpus used in the translation pattern extraction phase (section 3.2). The strength of association between the two words that make up each bigram is computed using Dice's co-efficient. Given a translation solution, the probability that it is a valid word sequence of the TL - *BG* in equation (64) - is determined using equation (65). If the translation solution is *n* words in length, the probability that it is a valid word sequence is calculated by taking the average score of each of its constituent bigrams. Each bigram in the translation is represented as $<w_i, w_{i+1}>$ where $1 \leq i \leq n-1$. The score returned for each bigram is added to a running total then divided by the number of bigrams in the translation or $n-1$.

(65)
$$BG = \frac{\sum_{i=1}^{n-1} 2\left(\frac{\left| w_i, w_{i+1} \right|}{\left| w_i \right| + \left| w_{i+1} \right|} \right)}{n-1}$$

Two important issues arise when using bigram models. Bigram models are tuned to the text type from which they were constructed. In these experiments, the bigram model is constructed from the TL sentences of the corpus from which the translation patterns are extracted. Therefore, the bigram model is perfectly tuned to the translations produced. The second issue is data sparseness. Bigram models are probabilistic models and become more reliable as more data is supplied. As bigram models are constructed from monolingual text, it is relatively easy to supply large amounts of data to the bigram model to improve its performance in terms of coverage and reliability.

The second component of the translation confidence score *BF* measures the amount of boundary friction incurred when TL fragments are bound to the TL variables of the base pattern. It makes use of the fact that the TL text fragments have been extracted from real texts and so there is information about the contexts in which the text fragments are known to have occurred. Therefore, a score is desirable that measures how well the inserted TL text fragment fits into the context of the preceding and subsequent TL text fragments of the base pattern (Figure 38).

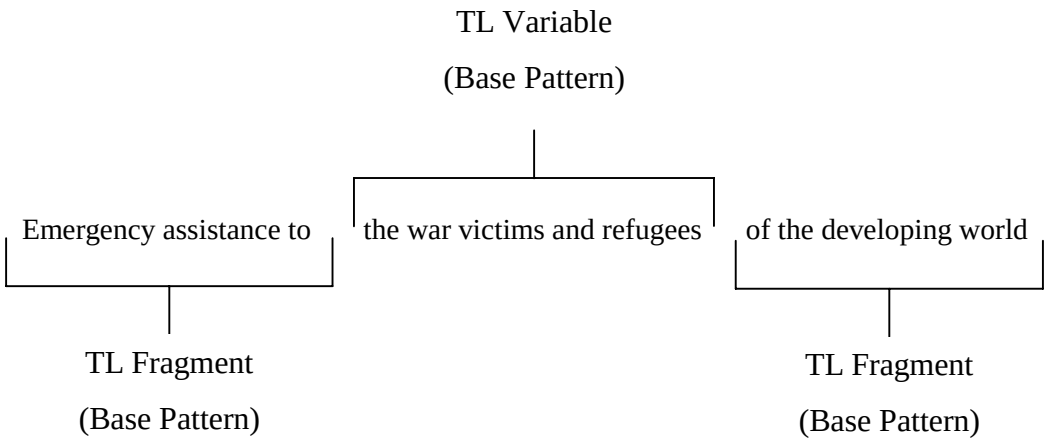

Figure 38: Binding an TL Text Fragment to an TL Base Pattern Variable

Boundary friction is measured by considering the two sequences of *N* words of the TL fragments of the base pattern preceding and subsequent to the inserted TL fragment. The corpus is scanned for instances of the initial and final word of the inserted TL fragment. For each instance of the initial word of the inserted TL fragment found in the corpus, the sequence of words prior to it is compared with the sequence of *N* words preceding the inserted TL text fragment in the base pattern. Similarly, for each instance of the final word of the inserted TL fragment in the corpus, the sequence of words subsequent to it is compared with the sequence of *N* words subsequent to the inserted TL fragment in the base pattern. Where there are multiple instances of the initial or final word of an TL fragment in the corpus, the sequence comparison that returns the greatest overlap is returned. The concept is borrowed from Somers et al. (1993) who use the image of hooks being attached before and after the TL fragment inserted indicating the words (and POS tags) that can occur before and after it.

To illustrate this idea, consider the example in Figure 38 where the TL text fragment *the war victims and refugees* is bound to the TL variable of a base pattern between the TL fragments *emergency assistance to* and *of the developing world*. First, the corpus is scanned for instances of *the*. For each occurrence of *the*, the sequence of words prior to it is recorded (66). Each of these sequences is compared with the original sequence of *N* words *{emergency, assistance, to, the}* as depicted in Figure 38. Starting

from the right hand side of the sequences, the maximum overlap between them is returned. In this case, the maximum overlap takes place with the sequence *{financial, assistance, to, the}*, returning an overlap value of 2 words (excluding the overlap with *the*). Similarly, the corpus is scanned for instances of *refugees*. For each occurrence of *refugees*, the sequence of words subsequent to it is recorded (67). Each of these sequences is compared with the original sequence of *N* words *{refugees, of, the, developing, world}*. Starting from the left hand side of the sequences, the value of the maximum overlap is returned. In this case, the maximum overlap takes place with the sequence *{refugees, of, the, war, in, the, Balkans}*, returning an overlap value of 2 words (excluding the overlap with *refugees*).

(66) [financial, assistance, to, the]
[financial, support, to, the]
[relief, aid, to, the]

(67) [refugees, of, the, war, in, the, Balkans]
[refugees, of, Angola, and, Namibia]
[refugees, of, war, torn, Africa]

An overlap score is returned for each TL fragment bound to an TL variable. In more detail, it is calculated as the average of the overlap score between the inserted TL fragment and the preceding TL base pattern fragment and the overlap score between the inserted TL fragment and the subsequent TL base pattern fragment. Where an TL fragment is inserted in sentence-initial or sentence-final position, only the overlap score with the previous or subsequent base pattern fragment is returned. The final score for each translation solution, returned as *BF* in (64), is calculated as the average overlap score for each TL base pattern variable. *BF* is calculated using (68), where F_i is the inserted TL text fragment for $1 \leq i \leq n$ where there are *n* TL variable positions in the base pattern and where $n>0$. F_{prev} represents the text fragment of the base pattern preceding F_i. F_{subsqt} represents the text fragment of the base pattern subsequent to F_i. *N*, which depicts

the size of the overlap window, is currently set at 5. Given the example translation solution depicted in Figure 38 where $n = 1$, BF is calculated as (2+2)/(2*5) = 4/10 = 0.4.

$$BF = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Overlap(F_i, F_{prev}) + Overlap(F_i, F_{subsqt})}{2N}\right) \quad (68)$$

The third component of the translation confidence score *BS* represents a measure of the probability that the TL text fragments retrieved during recombination are translations of the SL text fragments bound to the SL variables of the base pattern. This is carried out using the bilingual similarity metric (36) used when aligning the text fragments of which the translation patterns are composed. For each local alignment between the SL and TL variables in the base pattern, the bilingual similarity score is calculated between the SL text fragments bound to the SL variables and their TL equivalents bound to the variables on the TL side of the base pattern. Each score is added to a running total, which is eventually divided by the number of variable alignments.

As an example, given the SL input *P Q R S T*, suppose the base pattern retrieved is (69). The first SL variable is aligned with the first TL variable and the final SL variable is aligned with the final two TL variables. The text fragment *Q* is bound to the initial SL variable and *S* to the remaining variable. If *Q′* is retrieved as the TL equivalent of *Q* and *{S′,X′}* is retrieved as the translation of *S*, the translation solution is given by the TL sequence of text fragments *P′ Q′ R′ S′ T′ X′*. The score *BS* in (64) is computed by taking the average of the bilingual similarity between the aligned fragments *Q*↔*Q′* and *S*↔*{S′,X′}*. If the bilingual similarity of the alignment *Q*↔*Q′* is calculated as 0.9 and the bilingual similarity of the alignment *S*↔(*S′*, *X′*) is calculated as 0.6, then *BS* is calculated as (0.9 + 0.6)/2 = 0.75.

(69) P (…) R (…) T ←→ P′ (…) R′ (…) T′ (…)

The final element of (64) is the penalty *p* which as applied if there are any TL variables in the base pattern that remain uninstantiated. This phenomenon occurs in two cases. The first case occurs where partial matches between the SL input and the SL side of the base pattern are acceptable. Partial matches occur when one or more variables on the SL side of the base pattern cannot be matched by SL text fragments in the remaining set of translation patterns. This results in uninstantiated variables on the TL side of the base pattern (see section 3.3.4). The second occurs when an SL fragment bound to an SL base pattern variable is composed uniquely of closed class words. Since such fragments are not considered in the alignment process, they have no TL equivalent. The result is that the equivalent TL base pattern variable remains uninstantiated.

The penalty *p* is calculated as the ratio of uninstantiated TL base pattern variables to instantiated TL base pattern variables. Assuming a translation solution was formed where 1 out of the 3 TL base pattern variables was left uninstantiated, *p* becomes 1/3. The term *1-p* then becomes (1-1/3) = 2/3. In actual fact, *p* is scaled by 0.9 so that if all TL base pattern variables remain uninstantiated (so that *p* becomes 1) *1-p* never becomes zero.

Each of the three terms of equation (64) – *BG*, *BF*, *BS* – returns a real value between zero and 1. To ensure that the sum of the three terms remains between zero and 1, $\omega_1 + \omega_2, + \omega_3 = 1.0$. In the experiments undertaken $\omega_1, = \omega_2, = \omega_3 = 1/3$, thus providing equal weighting to each term in the equation. Due to the scaling of *p*, the translation confidence score (64) lies between the range 0 and 0.9.

3.4 Initial Results

This section presents initial results for the approach to EBMT described in this chapter. The aim is to give the reader some appreciation of the baseline performance of the approach, before the hypothesis is tested that adding external linguistic knowledge improves performance in terms of both accuracy and coverage. The hypothesis is made since it is known that approaches based (almost) purely on processing raw translations examples without external linguistic knowledge are prone to certain sources of error, such as false collocations and false translations. This is described in further detail Chapter 4.

The most comprehensive and useful method of testing an example-based approach to machine translation such as this, is to extract translation patterns from an appropriate bilingual text and translate unseen SL input of a similar text type, using the translation patterns extracted. In order to make a consistent evaluation and enable fair comparisons between variants of this EBMT methodology (see Chapter 4 and Chapter 5), an automatic evaluation methodology is used.

Using the algorithms described in section 3.2, translation patterns were extracted from a 3,000-sentence-pair sample of the World Health Organisation AFI corpus of 4,858 English and French titles[8]. In total, 10,767 translation patterns were extracted and passed to the recombination stage. The most frequent values for *p* and *q* (Figure 12) were 2 and 3. From the remaining set of 1,858 sentence pairs in the corpus, 1,000 were randomly selected and used as a test set of unseen SL input. For each SL input translated using the translation patterns extracted and the recombination algorithms described in section 3.3, one or more translation solutions were produced. The best translation string among all translation solutions was selected according to the translation confidence score given in section 3.3.5, so that only one solution was returned for each SL input. No direct matches between the SL input and the SL sentences in the corpus were sought. Only the EBMT mechanism is evaluated.

The correctness of each translation produced *TR* was determined automatically by comparing it with the reference translation *RT* as given in the corpus. The Levenshtein Distance (LD) between *TR* and *RT* was calculated, then normalised over the maximum distance between the two strings (maximum string-length), subtracted from 1 and multiplied by 100 to return a "percentage of similarity" score (70). Our implementation of LD used words as the fundamental unit as opposed to characters in applications such as spell checkers.

(70) $$SimScore = \left(1 - \frac{LD(TR, RT)}{Longest(TR, RT)}\right)100$$

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

The distribution of results for the translations of the test set of 1,000 unseen SL input sentences is given in Figure 39. Only full, as opposed to partial translations (section 3.3.4), were considered. 340 of the 1000 SL input sentences were successfully translated. Recall, which is calculated as the number of SL input sentences successfully translated divided by the total number of SL input sentences in the test set, therefore stands at 340/1000 = 34%. The rate of recall improves if only the SL input sentences in the test set that are uniquely composed of lexical items found in the training set of 3,000 sentence pairs are considered. This provides a more lenient calculation of recall as no SL input sentences are considered for which a full translation would be impossible due to unknown lexical items. The rate of recall maximally achievable is therefore calculated as the number of SL input sentences successfully translated divided by the number of SL input sentences in the test set that are uniquely composed of lexical items found in the training set. 728 of the SL input sentences in the test set were uniquely composed of lexical items in the training set, therefore maximal recall is calculated as 340/728 = 47%.

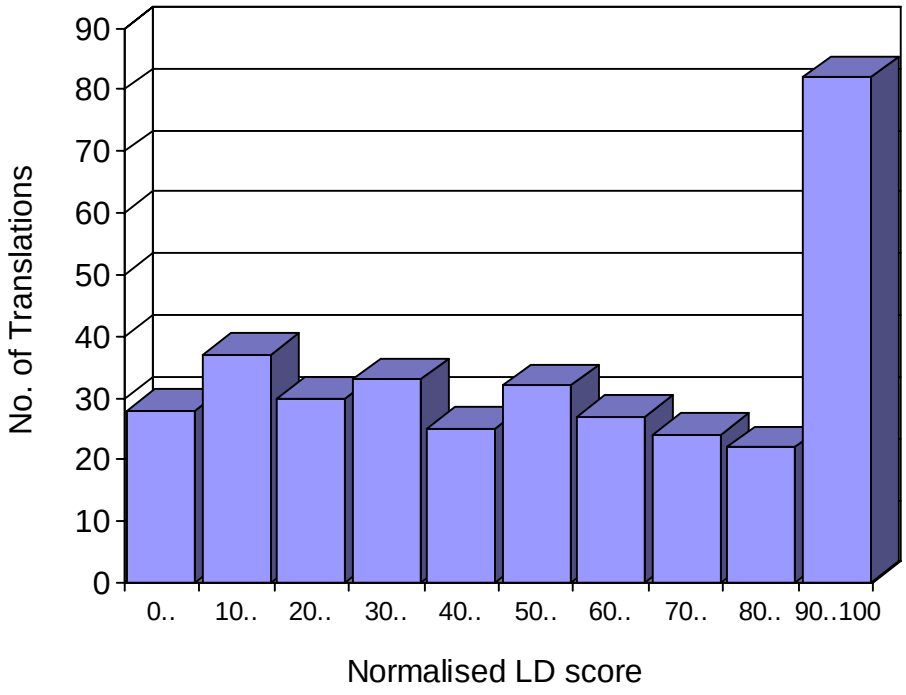

Figure 39: Distribution of Results

Of the 340 SL input sentences translated, 82 translations - 24% of the total number of translations - were scored as 90-100% accurate. Moreover, 73 of those translations - 21.5% of the total number of translations produced - were 100% correct. Accurate translations were largely produced on account of translation patterns whose SL side substantially covered the SL input. This reduced the amount of translation ambiguity, boundary and passage friction considerably. Moreover, such patterns, of which (47) is typical in the training set used, represent extremely frequently occurring phenomena.

In order to improve upon the baseline performance depicted by Figure 39, improvements to the basic approach are required to return a distribution curve where a greater number of high-scoring and fewer low-scoring translations are produced. Methods to improve performance are discussed in the next chapter where the weaknesses of the basic approach described in this chapter are discussed and the addition of external linguistic knowledge sources to alleviate these shortcomings is investigated.

4 Incorporating Linguistic Knowledge

The previous chapter introduced the basic methodology behind the approach to EBMT presented in this thesis. However, the principle where similar distributions of surface forms in a sentence-aligned corpus are considered to be translations of each other is error prone. This leads to the production of translation patterns that are *false translations*. Reducing the likelihood of false translations requires identification of the contexts in which they occur and the implementation of effective solutions. This chapter highlights the main sources of error and provides solutions to them based on the addition of external linguistic knowledge. The extra computational complexity incurred by the addition of external knowledge sources is also investigated. Finally, initial results are presented to test the hypothesis that adding linguistic knowledge improves the performance of the baseline methodology, producing a greater number of accurate translations and increasing coverage.

The principle source of false translations is the phenomenon where, due to the existence of homographs, SL lexical items that are orthographically identical have orthographically different TL equivalents. This general phenomenon encompasses the concepts of homography, polysemy, category ambiguity and translation ambiguity. As the extraction of translation patterns is reliant on the principle of string co-occurrence, distinguishing words on criteria other than just their orthography is the key to resolving false translations. The basic methodology outlined in Chapter 3 operates solely on the orthographical level. Therefore, introducing external linguistic knowledge sources provides a method of distinguishing words on the basis of deeper linguistic criteria.

External linguistic knowledge, in the form of morphological analysis and POS tagging, is added in order to reduce the likelihood of false translations and thus increase the accuracy of the translation patterns extracted. In addition, the aim of incorporating these resources is to increase coverage. Linguistic knowledge is added incrementally resulting in three variants of this approach to EBMT. The first variant, outlined in the previous chapter, is based on the distributions of surface forms in a sentence-aligned corpus. It requires no significant external linguistic resources. However, as outlined in the previous chapter, a very small amount of (optional) linguistic information is added in

order to avoid obvious errors. This takes the form of monolingual lists of closed class or function words along with information for tokenisation. The addition of this information compromises, to some extent, the language-neutral nature of the methodology and its portability to new language pairs. However, this information is obtainable in just a few person-hours, at least for European languages.

The second variant uses the methodology outlined in the previous chapter but is augmented to include morphological analysis. The corpus is lemmatised so that translation patterns are composed of lemmas and translation patterns on the morphological level are extracted. Furthermore, clitic expressions are expanded in order to increase coverage and reduce boundary friction. The recombination phase is updated accordingly. The third variant is augmented further to include POS annotation.

As more linguistic knowledge is added to the system, portability to new text types and language pairs decreases. In light of this trend, the linguistic information 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 when they are not fully available, they are created rapidly by well established methods. Furthermore, POS taggers, such as Brill's (1992) tagger are easily trained for new languages and text types.

4.1 Morphological Analysis

The following section outlines the variant of this approach to EBMT that is updated to include morphological analysis. This includes details of the updated methods of extracting translation patterns, the recombination phase and any issues concerning tokenisation and reassembly.

4.1.1 Tokenisation

The tokenisation process is updated to include information enabling the expansion of clitic expressions. For example, the English clitic *can't* is expanded to *can* + *not*. Similarly, French clitics such as *du* and *des* are expanded to *de* + *le* and *de* + *les* respectively. The addition of such minimal linguistic information is optional and is intended to increase coverage and reduce boundary friction.

4.1.2 Translation Pattern Extraction

The incorporation of morphological analysis into the translation pattern extraction algorithm is intended to enable morphological variants of the same lemma to be grouped together. This prevents false translations occurring where different SL morphological variants of the same lemma with identical orthographies are translated by orthographically different morphological variants of the same TL lemma. Consider the corpus sample in (71). The two instances of the English verb *give* are orthographically identical as English does not mark verbs in the past for person or number. Verb forms in English correspond to verb forms in French which do differentiate as to singular and plural etc. The two instances of *gave* are translated by morphological variants of the French verb *abandonner* with different orthographies. The SL items occurring in both sentences in the sample (shown in bold) combine to form an intuitively correct collocation, whereas the TL items occurring twice combine to form a 'false collocation'. Equating the SL and TL collocations produces the false translation or incorrect translation pattern (72). The two complement translation patterns formed would also become false translations.

(71) He **gave the** job **up** ←→ Il abandonna **le** poste

They **gave the** plan **up** ←→ Ils abandonnèrent **le** plan

(72) (…) gave the (…) up ←→ (…) le (…)

If the corpus sample in (71) is lemmatised (73), all morphological variants are normalised and grouped according to their lemma (lexical form).[9] This time, both lemmas *give* and *abandonner* occur twice in the SL and TL sentences of the corpus respectively, enabling more intuitively correct collocations to be produced. Articles are also normalised to the singular masculine form. As a result, an accurate, if general, translation pattern composed of lemmas is produced (74). Furthermore, abstracting morphological variants in a corpus to their lemma increases the likelihood of string co-occurrence.

[9] The term *lexical form* is introduced to clarify the concept of *lemma*. For English and French, the lexical or dictionary form of a word is the infinitive for verbs, (masculine) singular for nouns and adjectives etc.

Hence, more collocations and translation patterns are likely to be produced. A greater number of translation patterns is conducive to increasing coverage.

(73) He **give the** job **up** ←→ il **abandonner le** poste

They **give the** plan **up** ←→ Ils **abandonner le** plan

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

The linguistic information necessary for morphological analysis is provided in the form of a list of root forms, a list of morphological suffixes and a set of two-level morphological spelling rules (Koskenniemi, 1983).[10] The spelling rules are compiled into a single Finite State Transducer (FST) using the PC Kimmo freeware package (Antworth, 1990). The bi-directional nature of FSTs is advantageous in that they not only enable the transduction of surface forms into root forms and endings, but also the transduction of a root form-plus-ending into a surface form. The latter task is necessary in recombination (section 4.1.3).

The list of root forms and suffixes is created manually. The list of suffixes records morphological change of the inflectional type only. Examples of English inflectional suffixes include *+s*, *+ed*, *+ing*, where + denotes a morpheme boundary. The set of two-level spelling rules are also created manually and are subsequently compiled into finite state tables. In the case of English, the *Englex* two-level description of English from the Consortium for Lexical Research at New Mexico State University was used[11]. Two-level rules for French were developed by the author of this thesis, inspired largely by Bescherelle (1986), and compiled into finite-state tables using the `KGEN` compiler, which is part of the PC-Kimmo package.

As an example of a composite two-level spelling rule for English, the rule given in (75) represents the phenomenon where *y* is transduced as *i* in certain restricted

[10] The term *root form* is used in the literature on two-level morphology. It is used here, in the context of English and French, to represent the same concept as that of *lemma* or uninflected dictionary form. However, to facilitate French analysis, the root form of a verb is represented as both the lexical form and a *stem* with the infinitival endings (eg. -er, -re, -ir) removed (Appendix C), hence the new term *root form*.

[11] `ftp://crl.nmsu.edu/CLR/tools/ling-analysis/morphology/englex_pc-kimmo/`

environments. The rule states that where *y* is found subsequent to *r*, and where *y* appears at the morpheme boundary (indicated by +) and where *s* is the initial letter of an added suffix, *y* is transduced as *i* and the morpheme boundary symbol + transduces as *e*. As an example, the rule accepts `try+s` ↔ `tries`. The colon indicates the transductions. The symbol on the left of the colon is transduced into the symbol on the right or vice versa. Where both symbols are identical, the symbol remains unchanged. This rule is made more applicable to wider lexical contexts by grouping classes of letters, such as vowels or consonants. By grouping all consonants under the symbol `C`, the rule becomes (76) and accepts a wider range of phenomena including `fly+s` ↔ `flies`, `prodigy+s` ↔ `prodigies` etc. Similar rules were developed for French morphological phenomena. The rule in (77) describes the phenomenon where a morpheme boundary symbol + transduces as *e* when preceded by *g* and when a suffix beginning in *a* or *o* is added. One example of this phenomenon is when the 3rd person plural imperfective verbal ending *+aient* is added to the stem *mang* of the verb *manger* to produce *mangeaient*, rather than the incorrect form *mangaient.

(75) `y:i <=> r:r ___ +:e s:s`

(76) `y:i <=> C:C ___ +:e s:s`

(77) `+:e <=> g:g ___ [o:o|a:a]`

Each surface form found in the corpus is analysed according to the set of root forms, suffixes and two-level rules provided and split into a root form (lemma) and an inflectional suffix. This operation is represented diagrammatically in Figure 40. As there are no morpho-syntactic rules – as this would represent a further knowledge source – there exists an amount of ambiguity during analysis. During analysis, a surface form may be grouped under more than one lemma. For example, the word form *setting* may be analysed as *set+ing* or *setting+[]*. In the first instance, the word form is analysed as the present participle of the verb *set*. In the latter case, it is analysed as the singular of the nominal form *setting*. The empty parentheses denote the case where there is no morphological change from the root form. As there is no information present to

distinguish one analysis from another, both analyses are recorded and represented as an ambiguous analysis, or *dual word*, in this case *set+ing/setting+[]*.

The two-level model of morphological analysis is convenient in that it is relatively easy to construct a list of root forms, affixes and a set of two-level spelling rules, at least for European languages. It took the author approximately one person-month to construct a set of rules, root forms and suffixes for French. However, the two-level model is not universally applicable. For English and many western European languages, which do not exhibit a large degree of infixation, the two-level model is capable of describing a large portion of the spelling changes of a language. However, Semitic languages and languages which exhibit morphological reduplication, such as Malay, are not conveniently described by the two-level model. Moreover, for any language, irregular patterns of morphological inflection must be handled separately.

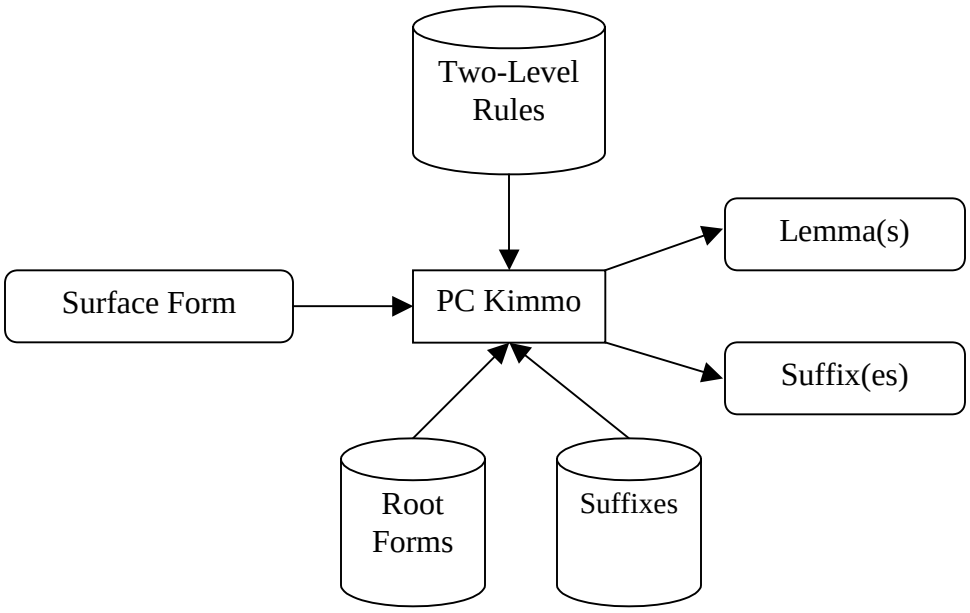

Figure 40: Resources Required for Morphological Analysis

The addition of morphological information to the translation pattern extraction algorithm not only produces translation patterns composed of lemmas but also translation patterns on the morphological level. These are termed *morphological patterns*. They are analogous to regular translation patterns except that the ‘text fragments’ are made up of

series of one or more morphological suffixes, such as *+ed*, *+ing* for English, instead of lexical items. The fragments and variables of which morphological patterns are composed are aligned analogously as in regular translation patterns. The function of morphological patterns is to represent bilingual patterns of morphological change. An example of a morphological pattern between English and French is given in (78). The SL fragment *[] [] +s* is aligned with the fragment *+s +s +s* and the two variables are aligned. The pattern could represent sentences such as *the big dogs barked* ←→ *les grands chiens aboyaient*, where *barked* and *aboyaient* are possible candidates for the SL and TL variable positions respectively.

(78) [] [] +s (…) ←→ +s +s +s (…)

The extraction of morphological patterns is a trivial process which takes place at the same time as regular translation pattern extraction. In fact, for each translation pattern, at least one morphological pattern exists as they are in fact equivalent. Morphological patterns are formed by replacing the lemmas of which regular translation patterns are composed with the corresponding morphological suffixes computed during analysis of the corpus. From an implementation point of view, where translation patterns are composed of text fragments pointing to positions in a corpus (Figure 13), morphological patterns are composed of the same fragments but contain pointers to the positions of the corresponding suffixes. As the two types of translation patterns are equivalent, the alignments of a regular translation pattern are directly transferable and become the alignments of the equivalent morphological pattern.

The following example illustrates the formation of morphological patterns. Consider the corpus sample in (79). Lemmatisation produces (80). The analyses are indicated by the + character representing the morpheme boundaries between lemmas and suffixes. Where no morpheme boundaries are indicated, no morphological change has taken place. From the lemmatised corpus sample in (80), the translation pattern (81), among others, is extracted in the usual manner. The morphological patterns in (82) are formed by replacing the lemmas of translation pattern (81) with the corresponding suffixes computed during morphological analysis of the corpus. The fragments and

variables of which the morphological patterns in (82) are composed are aligned in the same way as the fragments and variables of which the regular translation pattern in (81) are aligned. This is possible as both regular translation patterns and morphological patterns represent the same bilingual patterns of translation. Regular translation patterns and morphological patterns are equivalent, except that the former operates on the lexical level, while the latter operates on the morphological level.

(79) The telephones worked ←→ Les téléphones fonctionnaient

The telephone failed ←→ Le téléphone échouait

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

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

(81) The telephone (…) ←→ Le téléphone (…)

(82) [] +s (…) ←→ +s +s (…)

[] [] (…) ←→ [] [] (…)

To summarise, the input to the translation pattern extraction phase is updated to include a two-level morphological analyser. The output is not only a set of translation patterns composed of lemmas, but also a set of analogous translation patterns – morphological patterns – made up of the corresponding morphological suffixes computed during analysis. The recombination phase is therefore updated to include processing on the lexical and morphological levels.

4.1.3 Recombination

The first step in the updated recombination phase is the analysis of the SL input. This is carried out in exactly the same manner as the analysis of the corpus (Figure 40). The SL input is analysed and a record is kept of the sequence of lemmas and the sequence of morphological suffixes produced. As an example, given the SL input in (83),

morphological analysis returns the corresponding sequence of lemmas (84) and morphological suffixes (85)[12].

(83)	Vaccines	and	essential	medicines
(84)	Vaccine	and	essential	medicine
(85)	+s	[]	[]	+s

Recombination then takes place exactly as described in the previous chapter but on both the lexical and morphological levels. First, all translation patterns that combine to cover the sequence of lemmas corresponding to the SL input are retrieved in a process analogous to that in Chapter 3. Translation patterns composed of lemmas that share lexical items with the lemmatised SL input (84) are retrieved and filtered, returning a set of one or more base patterns (86). Any unmatched portions of the lemmatised SL input, *essential* in this example, are bound to the SL variables of the base pattern. The TL equivalents of these SL fragments are retrieved from the remaining set of translation patterns, as shown in bold in (87), and bound to the appropriate TL variables in the base patterns to produce a set of one or more 'translation strings' (88). These TL strings are in fact sequences of TL lemmas, as the translation patterns are themselves composed of lemmas in order to cover the lemmatised SL input.

(86) Vaccine and (…) medicine ←→ Le vaccin et le médicament (…)

(87) **essential** (…) for (…) ←→ (…) pour (…) **essentiel**

(88) Le vaccin et le médicament essentiel

In order to produce translations made up of surface forms, recombination on the morphological level is also carried out. Morphological patterns that combine to cover the sequence of SL morphological suffixes corresponding to the lemmatised SL input (85)

[12] An example without ambiguous analyses was chosen for reasons of clarity

are retrieved and recombined. Recombination on the morphological level uses largely the same methodology as outlined in the previous chapter, but with subtle differences. For example, as the set of all unique morphological suffixes is far smaller than the set of all unique lexical items for any given morphologically analysed corpus, retrieving base morphological patterns by the same pattern-matching method (3.3.1) incurs an impractical processing overhead due to substantial ambiguity and complexity. Instead, an alternative strategy is employed. Candidate morphological patterns that share one or more morphological suffixes with the analysed SL input are retrieved. Inverted files are used as before to speed up the retrieval process. Base morphological patterns are retrieved from this set by considering those that are analogous to the base pattern composed of lemmas retrieved previously.

For a given SL input, a base morphological pattern must be equivalent to the given regular base pattern composed of lemmas. The two patterns must be equivalent in terms of the number, size and position of the SL and TL fragments and the number and position of the SL and TL variables. The alignments between the fragments and variables must also be identical. As an example, consider the base pattern composed of lemmas in (89). The morphological pattern (90) represents an equivalent base morphological pattern, assuming that the fragments and variables of (89) are aligned in exactly the same manner as in (90). In surface terms, the two patterns are the same shape. In more detail, the SL and TL fragments are of the same length, occur in the same positions and contain the same number of SL and TL variables in the same positions.

(89) A B C (…) F G (…) ←→ A B C (…) F G (…)

(90) [] [] +s (…) [] +ed (…) ←→ +s +s +s (…) +s +aient (…)

Recombination on the morphological level then proceeds in the same manner as recombination on the lexical level. Given a base morphological pattern and a sequence of SL morphological suffixes corresponding to the lemmatised SL input, any unmatched portions of the sequence of suffixes are bound to the SL variables of the base morphological pattern. The TL equivalents of these unmatched portions of suffixes are

subsequently retrieved from the remaining set of morphological patterns and bound to the appropriate TL variables in the base morphological pattern. The result is a set of one or more sequences of TL morphological suffixes.

True translation strings composed of surface forms are produced by layering each sequence of TL lemmas produced by recombination on the lexical level with each sequence of suffixes produced by recombination on the morphological level. Each pair of sequences is combined and the formation of a translation string made up of surface forms is attempted. Surface forms are generated by combining the lemma at each position in the sequence of lemmas with the morphological suffix at the corresponding position in the sequence of suffixes. However, one condition must be satisfied. The lengths of the TL fragments of lemmas retrieved and bound to the TL base pattern variables must be equal in length to the corresponding TL fragments of morphological suffixes bound to the TL variables of the base morphological pattern. This conditions encourages the generation of meaningful translation strings and ensures that the sequences of lemmas and suffixes are equal in length. Assuming this condition is satisfied, if the sequence of lemmas and suffixes are both n items in length, then $\texttt{lemma}_i$ is combined with $\texttt{suffix}_i$ for $1 \leq i \leq n$ to produce a surface form at position i in a new sequence of surface forms. As an example, suppose recombination on the lexical level produces the sequence of lemmas in (91) and recombination on the morphological level produces the sequence of TL suffixes in (92), the corresponding sequence of surface forms produced is given by (93). Combining a lemma with the `[]` symbol means that no generation is required and that the lemma remains unchanged.

(91)	{Le,	vaccin,	et,	le,	médicament,	essentiel}
(92)	{+s,	+s,	[],	+s,	+s,	+s}
(93)	Les	vaccins	et	les	medicaments	essentiels

The direction of the FST used in morphological analysis is reversed so that it accepts a lemma (root form) and a morphological suffix as input and generates a surface form as output. As neither a word grammar nor a set of morpho-syntactic rules are

present to validate the word forms generated, an alternative strategy is required. The solution involves the use of a list of TL surface forms, against which generated word forms are checked. This prevents the generation of erroneous word forms such as *set+ed* → **setted*. The list of surface forms was provided by a French spelling dictionary. The updated architecture depicting the knowledge sources required for generation is shown in Figure 41.

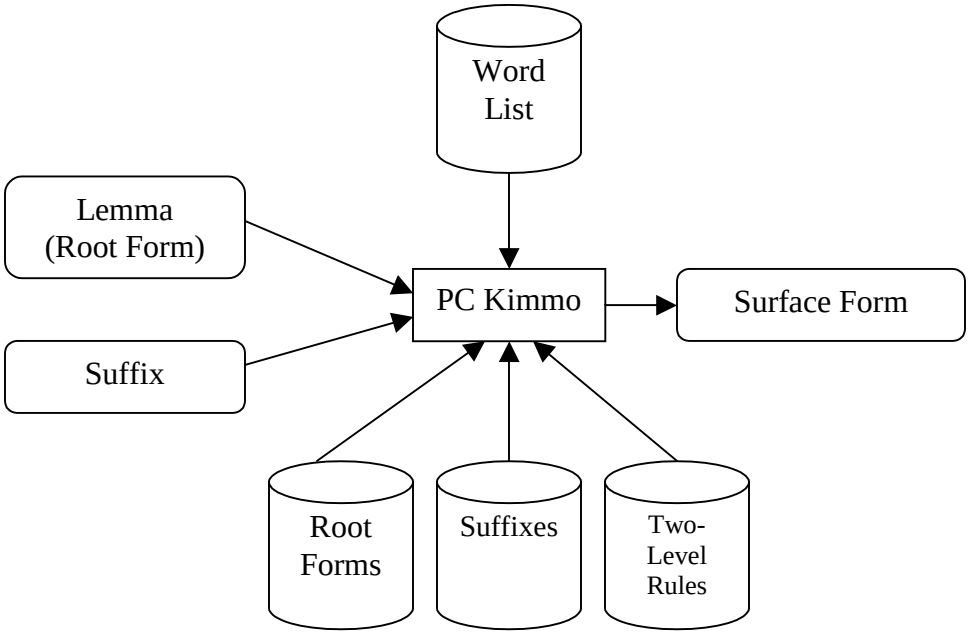

Figure 41: Resources Required for Generation of Surface Forms

In the case where (91) and (92) combine to form (93), each lemma and suffix combination generated a valid word form in the TL. Where a lemma+suffix combination fails to generate a valid TL word form, the combination process for that particular sequence of lemmas and suffixes is terminated as no complete string of TL surface forms is possible.

As mentioned in section 4.1.2, there is ambiguity during morphological analysis without any means of disambiguation. As a consequence, translation patterns, both regular and morphological, may contain 'dual words' where each analysis is recorded. This does not present a problem for the core recombination phase, since analysis of the SL input returns the same ambiguous analyses. Therefore, matching a lemmatised SL

input, containing ambiguous analyses, against a set of translation patterns composed of lemmas containing ambiguous analyses, such as *set/setting* in previous examples, is unproblematic. Ditto for morphological patterns, as 'dual suffixes' are possible when there are ambiguous analyses. However, in the absence of disambiguation, the layering phase in recombination requires the specification of each possible ambiguous analysis. Therefore, for each sequence of TL lemmas produced by recombination where there are ambiguous analyses, each possible sequence of lemmas, where only one analysis of each ambiguous cases is specified, is considered for layering with each possible sequence of suffixes, where only one analysis is specified, of each sequence of TL morphological suffixes produced by recombination. For example, if one of the sequences of TL lemmas produced by recombination were: s*et/setting the standard for staff train/training*, where the noun/verb ambiguity of *setting* and *training* is represented as a dual word in each case, each possible sequence of possible analyses is considered in the layering algorithm. In this case, four sequences of lemmas are considered (94). Similarly, for each sequence of TL suffixes containing ambiguous analyses, each possible sequence of suffixes is considered.

(94) set the standard for staff train
set the standard for staff training
setting the standard for staff train
setting the standard for staff training

The `Layer()` algorithm, shown in Figure 42, takes two arguments as input: the list of sequences of lemmas `LemmaSeqList` and the list of sequences of suffixes `SuffixSeqList` produced by recombination on both the lexical and morphological levels. Both lists may contain sequences of lemmas or suffixes containing ambiguous analyses. Therefore, for each sequence of lemmas (suffixes), each possible sequence is produced where each analysis of any ambiguous cases is specified, as in (94) above. Given the input list `LemmaSeqList` (`SuffixSeqList`), for each sequence of lemmas (suffixes), one or more sequences of lemmas (suffixes) is produced where each analysis of any ambiguous cases is specified. Each sequence produced is added to the list of

sequences `UnambigLemmaSeqList` (`UnambigSuffixSeqList`) This is carried out by the function `ReturnAllPossibleSequences()` which resembles the core recombination algorithm in Figure 36, which, for each base pattern, recursively considers all possible (translation) solutions.

```
Layer(LemmaSeqList, SuffixSeqList) → List of Translations
   UnambigLemmaSeqList = ReturnAllPossibleSequences(LemmaSeqList)
   UnambigSuffixSeqList = ReturnAllPossibleSequences(SuffixSeqList)
   For each lemma sequence, LemmaSeq, in UnambigLemmaSeqList
      For each suffix sequence, SuffixSeq, in UnambigSuffixSeqList
         SurfaceSeq = Combine(LemmaSeq, SuffixSeq)
         Translations.add(SurfaceSeq)
   Return Translations
```

Figure 42: Algorithm for Layering the Sequences of Lemmas and Suffixes

Once all sequences of lemmas and suffixes with ambiguous cases specified are produced, each sequence of lemmas `LemmaSeq` is combined with each sequence of suffixes `SuffixSeq` in an attempt to produce a sequence of surface forms `SurfaceSeq` representing a translation solution, just as (91) is combined or layered with (92) to produce (93). Each `SurfaceSeq` is added to a list of translation solutions `Translations` which is returned after all iterations.

Each translation solution produced by the `Layer()` algorithm is reassembled to produce true TL translation strings. This process is now updated to include reversing the rules used to expand clitic expressions such as *des* and *du* in French. For example, the sequence of French surface forms: *{Services, de, le, centre, de, épidémiologie}* is reassembled to produce the French string: *Services du centre d'épidémiologie*. Given the set of all reassembled translation strings, each is assigned a score of translation confidence in exactly the same manner as for the baseline methodology (3.3.5).

4.2 Part-of-Speech Annotation

Incorporating POS information into the translation pattern extraction process, in addition to the morphological information incorporated in section 4.1, aims to eradicate a further source of error leading to the production of false translations. By adding more linguistic

knowledge, lexical items are distinguished by linguistic criteria deeper than just their orthography or morphological variations. Adding a POS tagger as a further knowledge source enables SL homographs that are of different parts-of-speech to be distinguished, thus resolving category ambiguity. The hypothesis is that words with identical orthographies but different parts-of-speech are very likely to be different 'words' relating to different concepts. Where SL homographs are of different parts-of-speech, their TL equivalents are likely to have different orthographies, resulting in the production of false translations. The result of adding a POS tagger is that SL or TL homographs of different parts of speech are not permitted to co-occur to form collocations. In effect, the part-of-speech label becomes part of a feature-structure which must be matched along with the string value of the lexical items. Lemmatisation is not always sufficient as homographs exist that are of different parts-of-speech that abstract to the same lemma.

As an example, consider the corpus sample in (95), where *chairs* appears both as a verb and a noun. The strings that co-occur to form the erroneous translation pattern or false translation in (96) are shown in bold. Lemmatisation does not provide sufficient information to solve a case such as this, as both instances of *chairs* abstract to the orthographically same lemma *chair*. The solution lies in preventing the two instances of *chairs* co-occurring to form a collocation and a false translation. The collocation extraction algorithm is applied, as in 3.2.1, but updated so that the part-of-speech label is included as a feature-structure which much be matched along with the string value of the lexical items. In this case, *chairs:VB* and *chairs:NN* do not match, despite their identical orthographies, and are not permitted to form a collocation together. Adding POS information therefore provides a greater amount of linguistic constraints in the translation pattern extraction process.

(95) He **chairs** a committee on **the** judiciary ←→ Il préside un comité sur **le** judiciaire

My **chairs** are in **the** cellar ←→ Mes chaises sont dans **le** placard

(96) (…) chair (…) the (…) ←→ (…) le (…)

Although this example aims to describe the resolution of category ambiguity, the two instances of *chair* more accurately describes a case of polysemy, as there is a metaphorical link between the two senses of *chairs*. However, polysemes and true homographs with no metaphorical transfer of meaning – such as *bank* (river) and *bank* (finance) - are treated analogously in this and many other approaches to MT.

A further advantage of adding POS information is that ambiguous analyses are resolved. The tagger used in this implementation is the TreeTagger of Schmid (1994). One advantage of this tagger is the fact that it is freely available for English, French, Italian and German. Portability to any combination of language pairs is therefore less problematic. A further advantage is that, unlike other freely available taggers such as Brill's (1992) tagger, TreeTagger incorporates a lemmatiser. This facilitates the resolution of ambiguous analyses. Therefore, dual words and morphological suffixes are effectively ruled out when translation patterns are extracted (section 4.1.2) so that translation patterns are only ever composed of single lemmas. Ditto for morphological patterns. As translation patterns do not contain ambiguous items, recombination produces sequences of single lemmas and morphological suffixes. As all sequences do not contain unambiguous cases, the `Layer()` algorithm in Figure 42 is modified so that the Function `ReturnAllPossibleSequences()` is unnecessary. Each sequence of lemmas in `LemmaSeqList` is combined with each sequence of suffixes in `SuffixSeqList`. This has the effect of substantially reducing complexity (see section 4.5.2).

4.3 Semantic Disambiguation

The solution to the false translation problem lies in effectively distinguishing word forms - strings of characters separated by white space or punctuation - on criteria other than just their orthographies. Distinguishing different words that relate to different concepts is not entirely possible by morphological analysis and POS tagging alone. SL homographs that share the same orthography, abstract to the same lemma, share the same POS tag but are semantically different are consequently very likely to have different TL equivalents. As an example, consider the corpus sample in (97). The two instances of the word form *marked* are identical in orthography and part-of-speech as both are verbs. The first instance of *marked* refers to the meaning 'correct' while the latter refers to the meaning

‘stain’. As a consequence of the polysemy of the verb *mark*, both instances have different TL translations, resulting in a false translation (98).

(97) She **marked the** paper ←→ Elle corrigea **la** copie

He **marked the** table ←→ Il tacha **la** table

(98) (…) mark the (…) ←→ (…) le (…)

Preventing the formation of false translations, such as (98), containing instances of conceptual ambiguity, involves employing techniques from the literature on word sense disambiguation. In so doing, identical word forms with different semantics would not be permitted to combine to form collocations. One possibility is to annotate the corpus semantically so that the semantics of a word is included as part of a feature-structure. Collocation extraction would then take place as before but updated to include the constraint that not only must the string value of the lexical items be equal, but also the values of the feature-structures. Distinguishing words based on semantic criteria is a more complex task than distinguishing words according to their orthographies or part-of-speech label. The coverage of semantic taggers is still not wide enough for general purpose NLP tasks and remains, to some extent, experimental. However, semantics is used in some MT systems as selection restrictions for verb-argument relationships. At the present stage, it is unclear how the corpus is to be annotated semantically. Consequently, it is not included in this approach, but remains a possible line of further enquiry.

4.4 Further Sources of Error

There are further sources of translation ambiguity to those highlighted in the above subsections where the addition of morphological analysis and POS annotation is insufficient. These further sources of error are unable to be resolved by tools for linguistic annotation. In this approach, translation ambiguity is a source of error that leads to the production of false translations. Translation ambiguity is typically problematic for many, if not all, approaches to MT.

As an example, an SL lexical item may be translated by multiple TL equivalents depending on the register or genre of the text type. In the case of French and English, the French word *domicile* may be translated as either *home* or *domicile* in English. A further source of translation ambiguity is the case where the grammar of the TL plays a role. For example, the English verb *know* may be translated by either the French verb *savoir* or *connaître*. The choice depends on whether the French verb is followed by a relative clause or infinitive (*savoir*) or a noun-phrase (*connaître*). The resolution of anaphoric expressions is also problematic. The English pronoun *it* may be translated in French as either *le, la, il* or *ça*. Finally, lexical gaps may also prove problematic. As an example, the French verb *gratiner* has no single English lexical equivalent. Its translation involves an amount of paraphrase which may not be consistent throughout a text.

Furthermore, the fact that translation patterns are extracted using an algorithm based on similar distributions of strings in a bilingual corpus means that there are fewer linguistic constraints than in traditional RBMT systems or EBMT systems that are based on generalising translations by (morpho-)syntactic means (Kaji et al, 1992; Carl, 1999 etc). The translation patterns extracted in this approach do not contain any restrictions on the number, gender or syntactic category of the text fragments and variables of which they are composed. As a result, many translations produced are ill-formed according to the syntax of the TL. Adding morphological analysis and POS annotation goes some way toward providing a greater set of constraints. However, increasing the amount of constraints is likely to result in a lower number of translation patterns. As a consequence, coverage of the system may suffer, although precision may increase. Moreover, incorporating external linguistic knowledge sources is no guarantee of increasing accuracy or coverage. Morphological analysers, lemmatisers, POS taggers and parsers etc. are all prone to a certain degree of error themselves. This error is subsequently passed on and multiplied in the system into which they are incorporated.

4.5 Complexity

As stated above, the aim of introducing linguistic knowledge is to remove certain sources of error, such as the production of false translations, and thereby to improve accuracy and coverage of the baseline EBMT methodology described in Chapter 3. The above

subsections have indicated the contexts in which false translation occur and suggested possible solutions to prevent them occurring. However, while adding external linguistic knowledge may improve performance, it comes at the cost of increased computational complexity, in terms of both time and space. The result is an increased processing overhead. The following sections illustrate the major sources of complexity of this approach to EBMT. This includes both the translation pattern extraction and recombination phases. More importantly, the effects on complexity of adding linguistic knowledge is investigated.

4.5.1 Translation Pattern Extraction

The translation pattern extraction process (3.2) is a notable source of complexity. This includes the algorithms in the monolingual phase designed to construct the collocations trees (3.2.1), the bilingual phase (3.2.2) where SL and TL leaf-node collocations are equated and the algorithms used to align the SL and TL text fragments and variables of which the translation patterns are composed (3.2.3).

The algorithms used to build the collocation trees are most notably complex in terms of space. As each word (surface form or lemma) from the list that occurs in two or more sentences is added to the tree, the size of the collocation trees constructed becomes an issue. As each lexical item may be added to the daughters of the root node and combine recursively with any subsequent daughters and so on, each lexical item may be added to the daughters of more than one node of the tree. As a consequence, the trees can grow to become very much larger than the original corpus. Therefore, space, or rather the size of the collocation trees, becomes a limiting factor. On account of their relatively high frequency, closed-class or function words are particularly problematic as they have the potential to dramatically increase the size of the collocation trees.

As the size of the corpus increases, the list of lexical items occurring in two or more sentences increases, as does the size of the collocation trees. The collocation trees have the potential to grow to such a size that they place a restriction on corpus size or extensibility - one of the key advantages of EBMT. Figure 43 gives an indication of the limitation on the size of the corpus incurred by the space-complexity of the collocation

tree building algorithms.[13] The graph 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 sentences. The same French corpus used in the experiments in 3.4 is used. The straight, dark line indicates the extrapolation of the data. Although the relationship appears to be linear, the trees use a substantial portion of memory. The limit on corpus size is therefore dictated by the memory of a particular machine or the efficient use of hard-drive space.

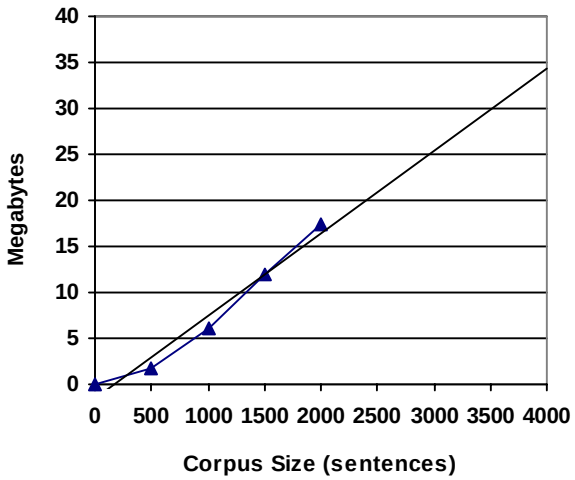

Figure 43: Size of Collocation Trees vs. Corpus Size

The size of the collocation trees represents the greatest obstacle to scaling-up this approach. One solution, as described above, is effective memory-management. The other is to employ alternative, more tractable strategies when building collocation trees. Presently, single lexical items or tokens are added, in turn, to the tree. As a result, the trees are made up of many levels of nodes, the number of which matches the number of single lexical items of which the terminal nodes are composed. In more detail, if a terminal node consists of a collocation containing 5 lexical items, then there are 5 levels

[13] Benchmarking the space-complexity of the collocation-tree construction algorithm is provided for convenience. From the highly recursive nature of the algorithms, it is unclear how to calculate their complexity exactly.

of nodes beneath the root node (including the terminal node itself), where each node contains one more lexical item than the node above it, as the tree is descended from the root to the leaves. In addition, a lexical item may combine to form a collocation with more than one node in the tree. As an example, in Figure 24, the lexical item *government* appears in two terminal nodes. Both leaf-node collocations are made up of three lexical items. As a consequence, two previous non-terminal nodes exist, from which the terminal node was constructed. Reducing the number of levels of nodes is one solution to reducing the size of the collocation trees (see section 6.2.1).

The bilingual phase (3.2.2), where SL leaf-node collocations are equated with TL leaf-node collocations when they have exactly the same sentence IDs, is also a notable source of complexity. An algorithm is employed that compares each SL leaf-node collocation with each TL leaf-node collocation. It therefore contains two embedded loops and so is clearly of a time-complexity in the order of $\mathcal{O}(mn)$, where m and n are the lengths of the lists of SL and TL collocations respectively. The polynomial nature of the algorithm ensures that it is practicable. However, as n and m can become very large, the processing overhead becomes significant.

The two-pass algorithm for aligning the text fragments and variables of which the translation patterns are composed (3.2.3) also displays polynomial time and space complexity. Both the well-known DP algorithm and the second-pass algorithm (Figure 29) contain two embedded loops. As a result, they too execute within the order of $\mathcal{O}(mn)$. In this case, m and n are the lengths of the SL and TL sequences of text fragments (or variables) respectively. Polynomial algorithms, such as DP, are practical but not fast. Due to the large number of translations patterns produced, alignment is the part of the translation pattern extraction phase that incurs the largest time processing overhead. However, the values of m and n are rarely large. Increasing the number of terms in the DP equation (43) also increases the processing overhead. The bilingual similarity metric also employs the polynomial DP algorithm when comparing word forms across languages using Levenshtein Distance (38). Again, the complexity is in the order of $\mathcal{O}(mn)$ and the values of m and n, the lengths of the SL and TL words respectively, are rarely large.

The addition of linguistic knowledge does not add significantly to the complexity of the translation pattern extraction phase. Morphologically analysing and POS-tagging the corpus is merely a pre-processing stage which is carried out before the word lists and the collocation trees are constructed. The algorithms are not significantly altered to take into account the extra information. The order of complexity remains the same.

4.5.2 Recombination

The recombination stage contains algorithms that are notable sources of complexity. These include filtering the set of candidate translation patterns to produce base patterns (3.3.1), the core recombination algorithms (3.3.2) and recursive matching (3.3.3). Updating these algorithms to include linguistic knowledge and processing on the lexical and morphological levels has the consequence that complexity is increased. Furthermore, new algorithms and therefore new sources of complexity are introduced as linguistic knowledge is incorporated. The `Layer()` algorithm (Figure 42), necessary due to ambiguous analyses, is the most notable example.

Given an SL input and a set of translation patterns composed of surface forms, the translation patterns are successively filtered until they qualify as base patterns (3.3.1). Recombination then proceeds as explained in (3.3.2). However, if the corpus is lemmatised to produce translation patterns composed of lemmas with an equivalent set of morphological patterns (4.1.2), the SL input is lemmatised. Subsequently, translation patterns that contain one or more lemmas in the SL input are retrieved and successively filtered until they qualify as base patterns. The fact that translation patterns are composed of lemmas does not increase the complexity or the processing overhead of this stage of the recombination process. However, the fact that morphological patterns sharing items with the sequence of morphological suffixes computed during analysis must also be retrieved and filtered to produce morphological base patterns increases the processing overhead, although not significantly.

The core recombination algorithm (Figure 36) is the stage where the amount of entropy incurred by translation ambiguity and the lack of constraints in the translation patterns produces the most notable source of computational complexity in the translation process. As any one SL fragment bound to the SL base pattern variables may have more

than one TL equivalent and there is no information present to favour one fragment over the other in terms of constraints, all possible TL fragments are considered. This results in the construction of (potentially) numerous translation solutions per base pattern.

The complex nature of the core recombination algorithm is clear. If a base pattern contains *n* local alignments between the SL and TL variables to cover the SL input, and there are T_i TL equivalents per alignment, such that $1 \leq i \leq n$, then the total number of translation solutions is the series $T_i * T_{i+1} \ldots T_n$ or (99). To illustrate the complexity, consider the example in the previous chapter, where there are two alignments between the SL and TL variables of the base pattern (55). The SL text fragment bound to the first SL variable in the base pattern has two possible TL equivalents such that $T_1 = 2$. The SL text fragment bound to the second SL variable of the base pattern has three TL equivalents, such that $T_2 = 3$. Therefore, the total number of translation solutions produced is $T_1 * T_2 = 2 * 3 = 6$. Six translation solutions are produced, as shown in (57).

(99) $$Solutions = \prod_{i=1}^{n} T_i$$

The complexity of the core recombination algorithm is unchanged whether the translation patterns are composed of surface forms, lemmas or morphological suffixes. However, where morphological analysis is included, the processing overhead of the core recombination stage is larger due to the fact that recombination takes place on both the lexical and the morphological levels to produce sequences of both TL lemmas and TL morphological suffixes. Moreover, on account of the fact that the set of unique morphological suffixes is far smaller than the set of unique lexical items, an SL fragment of morphological suffixes is likely to have many more TL equivalents than an SL text fragment, so that T_i in (99) is generally larger. Therefore, the number of solutions produced when recombining on the morphological level, again calculated by (99) is, in practice, frequently greater than that of recombination on the lexical level given the same value for *n*, despite the same degree of complexity.

Where recursive matches are included (3.3.3), the complexity of the core recombination process increases. Given an SL fragment that cannot be matched against

single SL fragments in the set of translation patterns, successively shorter portions of it are matched against multiple SL fragments from the remaining set of translation patterns. The translations of these portions are concatenated naively to form its TL equivalent. As each portion of the original SL fragment may be matched against more than one SL fragment in the translation patterns, the portion of the SL fragment is likely to have more than one translation equivalent. Therefore, there may be many more translation equivalents for the original SL fragment than if it were matched directly against a single SL fragment. It is this extra parameter that increases complexity.

The similarity between the core recombination algorithm (Figure 36), ensuring that all TL equivalents are considered, and the recursive matching algorithm enables the complexity of the recursive matching process to be deduced as above. If an SL fragment is partitioned k times, and there are t_j TL equivalents for each portion of the SL fragment at j, where $1 \leq j \leq k$, then the total number of TL equivalents for the original SL fragment is the series $t_j * t_{j+1} \ldots t_k$ or (100). During the core recombination process, where an SL fragment of a bijective alignment between an SL and TL variable of a base pattern cannot be matched directly against the SL fragments in the remaining set of translation patterns, a recursive match is sought. Where a recursive match is successful, occurring at variable position i, T_i in (99) is replaced by the series (100). Hence, the number of solutions in recombination is increased to (101).

(100) $$Equivalents = \prod_{j=1}^{k} t_j$$

(101) $$Recombination = \prod_{i=1}^{n} \prod_{j=1}^{k} T_{i,j}$$

Where translation patterns are composed of surface forms, the core recombination algorithm returns sequences of surface forms that are effectively translation solutions. Incorporating linguistic knowledge into recombination, so that processing takes place on both the lexical and morphological levels, returns sequences of TL lemmas and

sequences of morphological suffixes that need to be layered in order to produce translation solutions composed of surface forms. The `Layer()` algorithm (Figure 42), where sequences of TL lemmas and sequences of TL morphological suffixes are combined to produce translation solutions made up of surface forms, is unique to the variants of this approach that incorporate linguistic knowledge (4.1, 4.2). Therefore, at this stage of the recombination process, a substantial increase in the processing overhead is incurred, compared to the approach based on surface forms.

The complexity of the `Layer()` algorithm (Figure 42) is greatest where morphological analysis is the only linguistic resource incorporated (section 4.1). Given the list of all sequences of lemmas and suffixes produced by recombination on the lexical and morphological levels, for each sequence, all possible sequences of lemmas and suffixes must be produced where each alternative analysis of any ambiguous cases is specified. The number of solutions produced by the `Layer()` algorithm is calculated as follows. Suppose that recombination on the lexical level produces a list containing m sequences of lemmas and recombination on the morphological level produces a list containing n sequence of morphological suffixes and where each sequence in the two lists may contain ambiguous analyses. For each sequence of lemmas seq_i where $1 \leq i \leq m$, each sequence of lemmas, where each alternative analysis of ambiguous cases is specified, is constructed. Ditto for each sequence of morphological suffixes seq_j for $1 \leq j \leq n$. Assuming that the length (number of lemmas) of each sequence seq_i is given by p_i, and that there is a maximum of k analyses per surface form, then the maximum number of sequences of lemmas in seq_{i} with each ambiguous analysis specified is given by k^{p_i}. Similarly, given that the length of each sequence of suffixes seq_{j} is given by q_j, for each sequence of suffixes seq_{j}, the maximum number of sequences of suffixes with each alternative ambiguous case specified is given by k^{q_j}. Therefore, if a list of m sequences of lemmas is returned by recombination on the lexical level, the maximum number of sequences of lemmas with each ambiguous case specified is given by (102). Similarly, given a list of n sequences of suffixes returned by recombination on the morphological level, the total or maximum number of sequences of suffixes with each ambiguous case specified is given by (103). In the experiments undertaken, $k = 2$, denoting a maximum of 2 analyses per surface form.

As each sequence of lemmas is layered with each sequence of morphological suffixes, the maximum number of combinations required by the `Layer()` algorithm is given as the product of (102) and (103), which is calculated as (104). However, the maximum number of combinations in the layering algorithm is theoretically never reached as not all items contained in any sequence of *p* lemmas or *q* suffixes are ambiguous cases. However, the exponential nature of (102) and (103) introduces the greatest amount of complexity in the `Layer()` algorithm.

(102) $$Sequences_{lemmas} = \sum_{i=1}^{m} k^{p_i}$$

(103) $$Sequences_{suffixes} = \sum_{j=1}^{n} k^{q_j}$$

(104) $$Combinations = \sum_{i=1}^{m} k^{p_i} \sum_{j=1}^{n} k^{q_j}$$

Adding the *TreeTagger* POS-tagger to the list of knowledge sources (section 4.2) eliminates the possibility of ambiguous cases occurring in the lists of sequences of lemmas or suffixes returned by recombination. As a result, seq_i and seq_j are always composed of unambiguous analyses. As a result, $k=1$, therefore the lengths of the sequences *p* and *q* no longer exert any bearing on complexity. Consequently, there are a maximum of *mn* combinations in the `Layer()` algorithm, reducing the complexity of the `Layer()` algorithm to polynomial (quadratic if $m=n$). The inclusion of the POS tagger not only provides extra linguistic constraints in the collocation tree building stage (3.2.1), with the aim of producing more accurate translation patterns, but also provides the linguistic constraints necessary to eradicate the parameters associated with ambiguity with the effect of substantially reducing the complexity of the layering algorithm.

4.6 Initial Results

This section investigates the effects on performance of incorporating linguistic knowledge into the baseline EBMT methodology presented in Chapter 3. The hypothesis that incorporating linguistic knowledge improves both accuracy and coverage is tested. Furthermore, the hypothesis that an increase in the amount of linguistic knowledge incorporated results in a proportional increase in accuracy and coverage, is also tested.

In order to test these hypotheses, a fair comparison of the results for each variant of the EBMT methodology presented in this thesis is carried out. The first variant represents the baseline methodology, based on surface forms, outlined in Chapter 3. This methodology is then updated to included morphological analysis, as outlined in section 4.1, to form the second variant, and updated even further to include POS annotation, as outlined in section 4.2, to form the third variant. Three variants are therefore ready for comparison, where each variant contains increasing amounts of linguistic knowledge.

To make a fair comparison, for each variant, translation patterns were extracted from the same corpus and the same test set of SL input sentences was translated. In this experiment, for each variant, translation patterns were extracted from a 2,500-sentence-pair sample of the WHO AFI corpus of 4,858 English and French titles - the same corpus as that used in 3.4. Complexity issues, as outlined in section 4.5 prevented the use of a larger corpus. The variant based on surface forms extracted 9,327 translation patterns from the sample, whereas the variant including morphological analysis extracted 9,610 patterns and the variant including morphological analysis and POS-tagging extracted 7,237 patterns. From the remaining set of 2,358 sentence pairs in the corpus, 1,000 were selected at random and used as a test set of unseen SL input sentences.

The test set was translated using the translation patterns extracted for each variant. The translations were scored automatically using the normalised Levenshtein distance score between the translations produced and the reference translations given in the corpus (70). For each of the three variants, a comparison of the distributions of the results of translating the test set is given in Figure 44. The labels in the legend refer to, in the order given, the baseline methodology (based on surface forms), the variant incorporating morphological analysis and the variant updated further to include POS annotation. As in

the experiment undertaken in 3.4, only full, as opposed to partial translations, were considered.

The distributions appear largely similar in that, for each variant, there is a greater concentration of high-scoring (90-100% accurate) as opposed to low-scoring translations for each division of scores. However, the salient differences between the distributions of each variant are established in terms of recall and the number of high-scoring and accurate translations, particularly in relation to the baseline methodology represented by the distribution curve marked 'Surface'. Where a variant produces a greater number of high-scoring translations and a higher rate of recall than that of the baseline methodology, this represents an improvement and a positive result for the hypothesis that incorporating linguistic knowledge improves performance. The results of the experiments are summarised in Table 1.

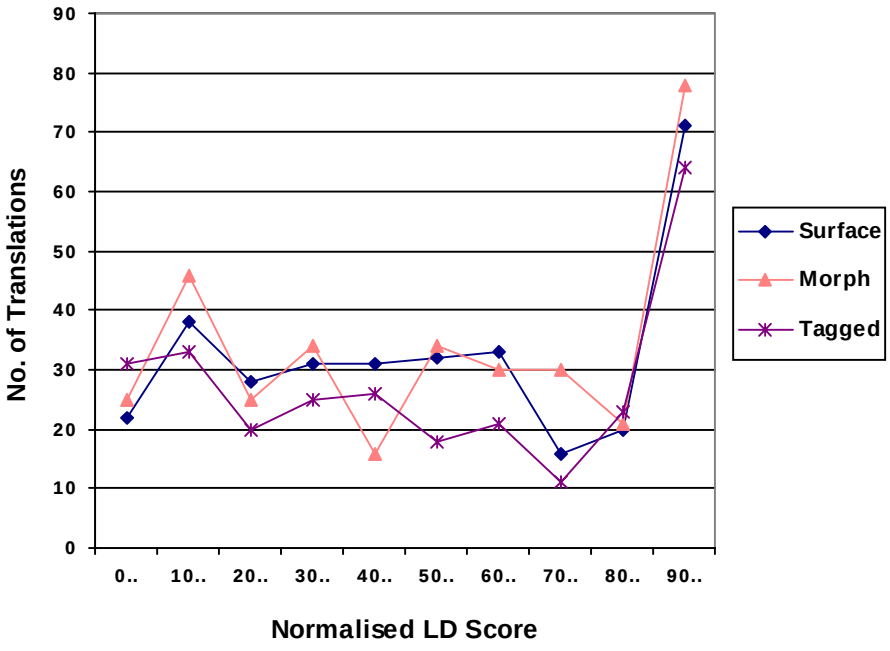

Figure 44: Comparison of Distributions of Results for each Variant

The rate of recall of the second variant including morphological analysis was greater than that of the baseline methodology. Out of the 1,000 sentences in the test set, 339 SL sentences were translated (33.9%) as opposed to the 322 SL sentences translated

(32.2%) by the baseline methodology. The maximally achievable rates of recall (46.5% vs. 44.2%), as described in section 3.4, also reflect this difference. The higher rate of recall achieved by the second variant including morphological analysis is largely due to the fact that a greater number of translation patterns (9,610) were extracted as opposed to the 9,327 translation patterns that were extracted by the first variant. Furthermore, the second variant produced a greater number of accurate (defined as 90-100% correct) or 100% correct translations. 78 accurate translations were produced by the second variant of which 68 were 100% correct. The baseline methodology produced 71 accurate translations of which 61 were 100% correct.

	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: Results of the Comparison of the Three Variants

The third variant incorporating morphological analysis and POS annotation performed less well than the baseline methodology based on surface forms. Out of the 1,000 SL sentences in the test set, 272 were translated, resulting in a rate of recall of 27.2%. The rate of recall maximally achievable (37.3%) was also lower than that of the baseline methodology. The lower rate of recall is a reflection of the fact that fewer translation patterns (7,237) were extracted. Fewer translation patterns were extracted on account of the fact that more constraints were placed on the translation pattern extraction process. Fewer accurate and 100% correct translations were produced, as compared to the baseline methodology. 64 accurate translations were produced, of which 53 were 100% accurate. Fewer high-scoring and accurate translations were produced on account of the fact that fewer translation patterns were extracted and that the POS-tagger introduced a greater amount of error than in previous variants.

The second variant, incorporating morphological analysis, clearly improved upon the baseline performance of the variant based on surface forms. The rate of recall is higher and a greater number of accurate translations were produced. The results of this experiment indicate that adding external linguistic knowledge sources, in the form of morphological analysis, does improve coverage and accuracy, although very slightly. This is to be expected as only a very small amount of linguistic knowledge is added. The third variant, incorporating both morphological analysis and POS annotation performed less well than the baseline approach. This result would indicate that increasing the amount of linguistic information does not necessarily improve performance incrementally. External linguistic knowledge sources are prone to a certain amount of error which is passed on to the system incorporating them. Inspecting the translation patterns produced by the third variant reveals many tagger errors which resulted in fewer accurate translations. The incorporation of POS annotation, as suggested in 4.2, may not be the optimal way to include POS information and improve performance.

5 Results and Evaluation

This chapter provides the reader with an indication of the performance of the EBMT methodology presented in this thesis. The experiment in 3.4 outlined the performance of the baseline methodology based on surface forms, using a training set of 3,000 sentence pairs of the English and French WHO AFI corpus. Initial results showed that a greater concentration of high-scoring (90-100% correct) translations were produced than for each previous division of scores. Chapter 4 introduced external linguistic knowledge sources into the methodology. The experiment presented in 4.6 outlined the effect on accuracy and coverage of adding linguistic knowledge to the baseline methodology presented in Chapter 3. The intention was to test the hypothesis that adding external linguistic knowledge increases the accuracy and coverage of the baseline methodology. The variant of the methodology that included morphological analysis did indeed translate more SL input sentences and produce a greater number of high-scoring translations than the baseline methodology. However, the third variant, which was updated further to include POS annotation, performed less well than the baseline approach. Fewer translation patterns were extracted, resulting in a lower rate of recall. Moreover, fewer high-scoring translations were produced. Therefore, the hypothesis that increasing the amount of linguistic knowledge into the EBMT methodology improves performance incrementally remains unproved.

The aim of the remainder of this chapter is to provide the reader with a greater understanding of the parameters in the methodology that affect performance. In more detail, the first part of this chapter (section 5.1) investigates methods of evaluating the translation patterns themselves. This includes testing the accuracy of the two-pass algorithm for aligning the text fragments and variables of which the translation patterns are composed. However, the most effective method of evaluating the translation patterns remains using them to translate unseen SL input sentences, as carried out in 3.4 and 4.6. In the remaining part of this chapter (section 5.2), the effect of changing certain system parameters and variables is investigated. This includes changing the size of the training set with the aim of judging the effect on recall and precision, changing the corpus text-type and even the language-pair. Further experiments include changing parameters such

as the co-occurrence or frequency threshold (3.2), so that lexical items must co-occur in not just a minimum of 2, but 3, 4 or even 5 sentence pairs in order to be considered translations of each other. The effect on recall and precision of not using recursive matching (3.3.3) in the recombination phase is also investigated along with the effect of including partial matches (3.3.4) in the translation process. It is also possible to compute the upper bound of the performance of the methodology, so that comparisons can be made between how the system translates a test set of SL input sentences and how well it *could* have translated the test set. The error rate, measured using plain as opposed to normalised Levenshtein Distance (LD) is also presented. Finally, a comparison is made between the results of this MT methodology and a commercial MT system.

5.1 Evaluation of the Translation Patterns

This section investigates methods of directly evaluating the translation patterns *per se*, rather than evaluating the EBMT methodology as a whole by using them to translate unseen SL input sentences. The experiments undertaken in McTait & Trujillo (1999) attempted to evaluate the translation patterns directly. Translation patterns were extracted from a 3,000-sentence-pair sample of the English and Spanish WHO AFI corpus. The accuracy of the translation patterns was determined by a panel of five native speakers of Spanish, fluent in English. It was unclear how to define recall, therefore it was not evaluated. However, precision was defined as the number of 'correct' translation patterns divided by the number of patterns in the evaluation sample. For each translation pattern evaluated, 'correct' was defined as, for each translation pattern, whether the SL and TL text fragments were translations of each other and whether the alignments were correct.

For the English-Spanish experiments, the precision of the translation patterns was calculated as 53% when the co-occurrence threshold was set at the default value of 2. When the threshold was raised to a minimum of 5, the precision of the patterns was raised to 76%. However, there are disadvantages to manually evaluating translation patterns. First, translation patterns are fairly abstract representations of bilingual phenomena. Therefore, there is no clear or widely accepted methodology to evaluate them. In addition, manual evaluation is a highly subjective task, which runs the risk of inconsistent assessment. A more effective method of evaluating translation patterns is to

use them to translate unseen SL input sentences, as carried out in 3.4, 4.6 and the remainder of this chapter (section 5.2), where the translations are evaluated automatically and therefore consistently.

A much clearer method of manually evaluating the translation patterns is to evaluate the alignments between the text fragments and variables. Vocabulary alignment is a well established field where evaluation methods are more widely known and accepted. Therefore, a meaningful evaluation of the two-pass alignment algorithm (3.2.3.2) is possible. In the experiments where translation patterns were extracted from a 3,000-sentence-pair sample of the English and French WHO AFI corpus using the baseline methodology, approximately 5% of the 10,767 patterns contained instances of non-adjacent local alignments between the text fragments. The low percentage of non-adjacent alignments is most likely due to the fact that English and French are closely related and, for the most part, structurally similar. From the 10,767 translation patterns, a sample of 220 was randomly selected for evaluation purposes. 200 of those patterns only contained instances of adjacent local alignments between the text fragments, whereas the remaining 20 contained instances of adjacent and non-adjacent local alignments. Table 2[14] indicates, for each local alignment type contained in the translation patterns, its frequency of occurrence in the sample and the number and percentage of correct alignments made. Alignments were considered correct or not by a panel of 4 judges who were either native speakers of French, fluent in English, or vice versa.

The table indicates that bijective or 1:1 local alignment types were by far the most frequent in the sample and were aligned with a high rate of precision. The more complex local alignment types, such as 2:1, occurred less frequently and were aligned with a lower degree of accuracy. Some alignment types, such as 1:0 and 0:1, are rare and did not occur at all in the sample. Furthermore, 188 of the 200 translation patterns containing only adjacent alignments (94%) and 13 of the 20 patterns including non-adjacent alignments (65%) were judged to have had *all* their text fragments correctly aligned.

[14] The column marked 'Frequency' does not total 220 as translation patterns frequently contain multiple local alignments.

In order to test the efficacy of the second-pass algorithm (Figure 29) and the gain in precision obtained over the plain DP alignment algorithm, the 20 translation patterns including non-adjacent alignments were realigned using only the DP algorithm. The result was that only 1 of the 20 patterns was judged to have had *all* its text fragments correctly aligned, as opposed to the 13 that were correct when the second-pass algorithm was included. Furthermore, only 23% of all the local alignments in the 20 translation patterns were correctly aligned as opposed to the figure of 73% achieved when including the second-pass algorithm.

Alignment type	Frequency	Correct	%
1:1	247	241	98
1:2	53	43	81
2:1	29	23	79
1:3	10	6	60
3:1	2	0	0
1:0	-	-	-
0:1	-	-	-
Non-adjacent	20	15	75

Table 2: Precision of Alignments between Text Fragments in Translation Patterns

The fact that only 1:1 or bijective non-adjacent alignments are possible would not appear to severely restrict the performance of the alignment algorithm. Out of the 200 translation patterns containing only adjacent alignments, in only one case would a non-bijective non-adjacent alignment have improved the global alignment. Furthermore, in only 4 out of the 200 cases would the computation of a bijective non-adjacent alignment have improved the overall global alignment.

5.2 Further Experiments

This section reports on the results of further experiments carried out where unseen SL input sentences were translated, using the translation patterns extracted, and automatically evaluated. These experiments were designed to investigate the effect on performance of certain parameters, such as the size of the training set, adjusting the co-occurrence threshold, changing the text-type and even the language pair.

5.2.1 Upper Bound

The aim of the following experiment is to determine the maximally achievable performance of this EBMT methodology. In order to do this, the upper bound of the system performance is computed. This enables a comparison to be made between the distributions of the results of the *actual* and *optimal* performances of the system. This compares how the system translates with how well it *could* translate a set of SL input sentences. In actual fact, what is being tested is the efficacy of the mechanism whereby a score of translation confidence is assigned to each translation solution (64). From the set of all translation solutions produced for each SL input, the string assigned the highest confidence score is selected as the final solution. However, the string selected may not always prove to be the most accurate translation according to the reference translation in the corpus. In the following experiment, the upper bound is computed by comparing all translation solutions produced, for each SL input, with the reference translation in the corpus and selecting the one with the highest normalised LD score (or highest similarity score) according to (70), irrespective of the translation confidence score assigned to it.

Using the approach based on surface forms, translation patterns were extracted from the same corpus of 3,000 sentence pairs of the English-French WHO AFI corpus, as in outlined in 3.2. The same test set of 1,000 unseen SL input sentences was also translated. The correctness of the translations was again determined by comparing them with the corresponding reference translation given by the corpus, using normalised LD.

The graph in Figure 45 shows the distributions of results between *actual* and *optimal* performance. The distribution curve representing optimal performance shows the extent to which fewer low-scoring and more high-scoring translations were produced than for the curve representing actual performance. As in the experiment in Chapter 3 (3.4), 82 high-scoring (90-100% correct) translations were produced, of which 73 were 100% accurate (actual performance). The upper bound (optimal performance) reveals that a maximum of 91 high-scoring translations were possible, of which 79 would be 100% accurate. The difference between actual and optimal performance is intuitive. The distance between the two curves indicates the efficacy of the translation confidence score. The closer the proximity of the two curves, the greater the efficacy.

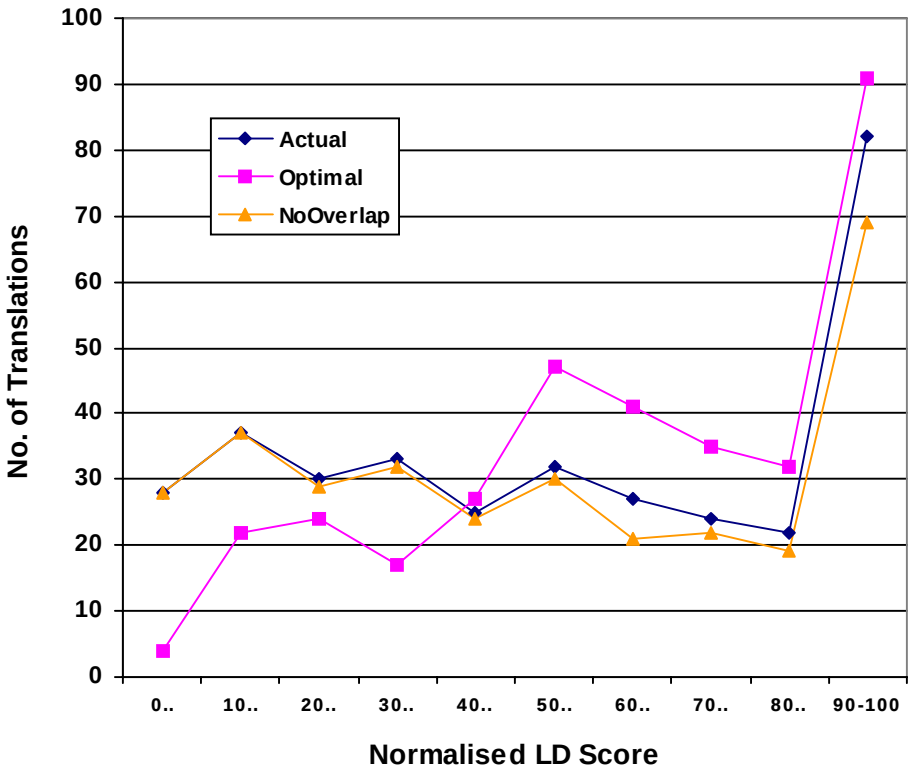

Figure 45: Actual vs. Optimal Performance

One further distribution curve shown is that marked 'NoOverlap'. This curve represents the results of the same test set but where the SL sentences that appear in the training set were removed. Hence there is no overlap between the sentences in the test set and the training set. From the 1,000 sentences in the original test set, 36 appeared in the training set and so were removed. As a result, there were 29 fewer SL sentences translated. However, there were only 13 fewer high-scoring (defined as 90-100% correct) translations produced. This experiment is intended to emphasise the fact that it is only the EBMT methodology that is evaluated in these experiments. No direct matches between SL input sentences and the SL sentences in the training are sought (as in a typical TM system). Moreover, removing *n* overlapping sentences between the training and test set does not necessarily entail *n* fewer 100% accurate – or very nearly so – translations.

The following experiment is a variation of that undertaken in 4.6 in Chapter 4. The experiment makes a comparison between the three variants of the EBMT methodology i.e. between the baseline methodology using surface forms, and the later

variants including linguistic knowledge. For each variant, translation patterns were extracted from the same 2,500-sentence-pair sample of the English and French WHO AFI corpus as that used in 4.6. The same test set of 1,000 unseen SL input sentences was also translated. The graph in Figure 46 shows a comparison of the distributions of results between the same three variants and the same training and test sets. This time the distributions of the upper bounds or optimal performances of each variant is compared.

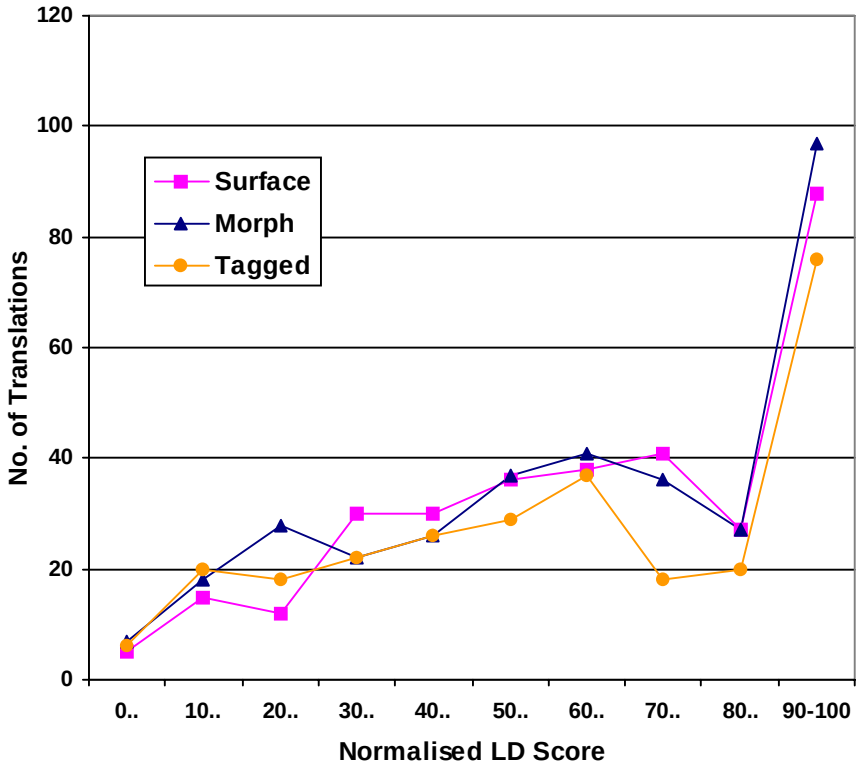

Figure 46: Comparison of Upper Bounds for each Variant

The distributions of the upper bounds appear much more similar and closer together than the distributions of the actual performances, as shown in 4.6 in Chapter 4. However, the distributions of the results of the upper bounds still display the same salient differences in that the upper bound of the variant updated to include morphological analysis translated more SL input sentences and produced a greater number of high-scoring translations than the upper bound of the baseline approach. Furthermore, the

upper bound of the variant including POS annotation translated less SL input sentences and produced fewer high-scoring translations.

5.2.2 Size of Training Set vs. Recall

This section investigates the relationship between the size of the corpus or training set and recall and precision. Experiments were undertaken where the size of the training set was varied and the same test set of SL input sentences was translated using the translation patterns extracted for each training set. The baseline methodology was used. Translation patterns were extracted using a training set of 500, 1,000, 1,500, 2,000, 2,500 and 3,000 sentence pairs of the English and French WHO AFI corpus. The same test set of 500 unseen SL input sentences, selected at random from the remainder of the WHO AFI corpus, was translated for each set of translation patterns extracted. The graph in Figure 47 depicts the relationship between recall and the size of the training set.

The curve represented by 'Full' in the legend shows the rate of recall where only full translations are considered. Recall is calculated as the number of SL input sentences translated divided by the number of SL sentences in the test set, returning the percentage of SL input sentences translated or rate of recall. The curve represented by 'Full+Partial' refers to the rate of recall where a partial translation was acceptable where a full translation was impossible. Finally, the maximally achievable rate of recall is represented by the curve marked 'Full(Maximal)'. This curve represents recall calculated by dividing the number of SL input sentences fully translated by the number of sentences in the test set that are composed uniquely of lexical items found in the training set. This provides a fairer method of calculating recall, as there is currently no method to translate words not seen in the training set.

The graph clearly indicates the trend where recall increases as the size of the training set increases. This behaviour is intuitive. The rate of recall rises fairly slowly when only full translations are considered. This indicates the need for much larger training sets if practical rates of recall are to be achieved. However, in cases where partial matches are considered acceptable and included, high rates of recall are rapidly achieved. Including partial translations results in a greater concentration of low-scoring as opposed to high-scoring translations (see section 5.2.5).

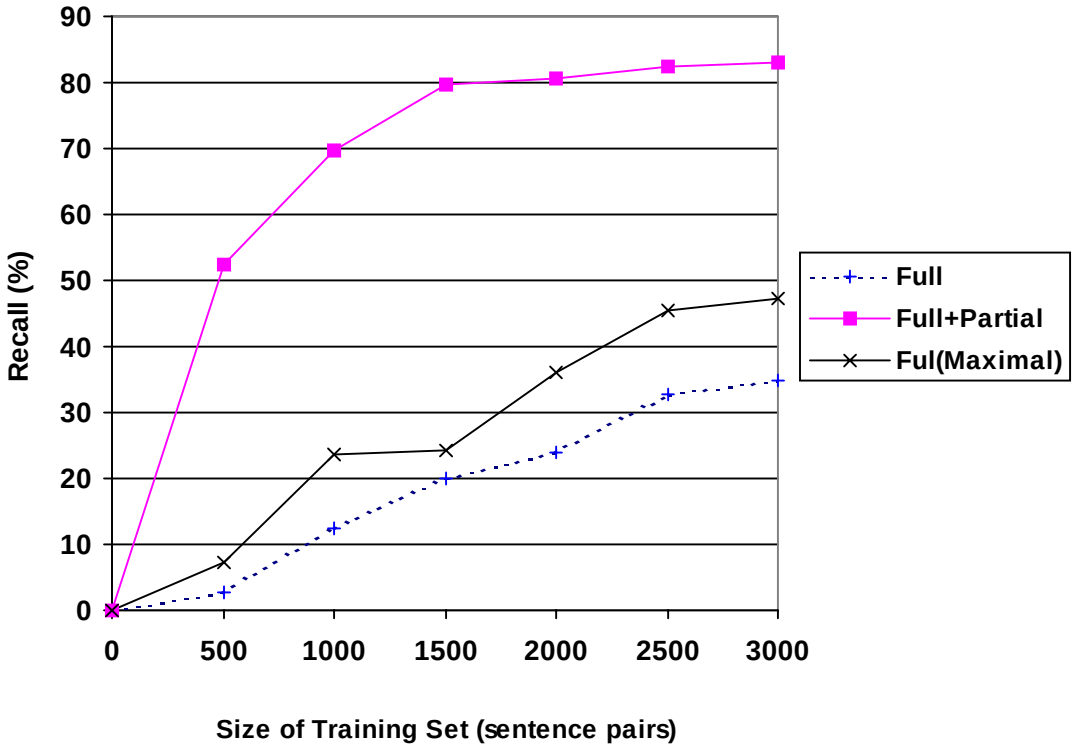

Figure 47: Size of Training Set vs. Rate of Recall

Furthermore, as the size of the training set increases, the number of high-scoring translations increases accordingly. For each training set of 500 to 3,000 sentence pairs, the experiment was repeated so that the same test set of 500 unseen SL input sentences was translated. This time a record was kept of the number of high-scoring (defined as 90-100% accurate) translations that were produced. Only full translations were considered. The number of accurate translations represented by the actual and optimal performances for each training set is also presented. The graph in Figure 48 clearly indicates the trend where a greater number of accurate translations are produced as the size of the training set increases. Therefore, as expected, both precision and recall increase with the size of the training set.

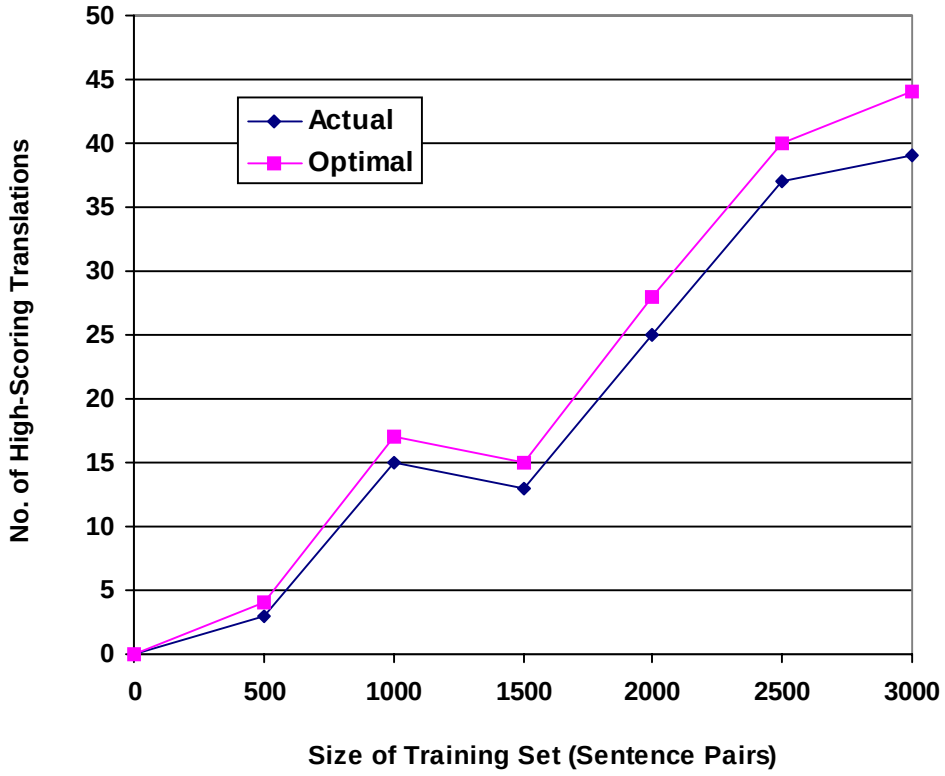

Figure 48: Size of Training Set vs. Accuracy

5.2.3 Error Rate

Each translation produced is evaluated automatically by comparing it with the reference translation given by the corpus. The LD – the minimum number of insertions, deletions and substitutions required to convert one string into another – is calculated between the translation produced and the reference translation. Our implementation of LD counts words rather than characters (as would be the case in a spelling-checker, for example). In order to produce a score representing the accuracy of the translation, a score representing how close it is to the reference translation is calculated. This is computed by normalising the LD between the two strings by the longest of the two strings (70). As substitutions, insertions and deletions each incur a penalty of 1, the length of the longer string represents the maximum distance between them or the maximum number of substitutions, insertions and deletions required to convert one string into the other (39).

It is also possible to show the distribution of word errors by returning, for each translation produced, the plain (un-normalised) LD between it and the reference translation. To demonstrate this, translation patterns were extracted using the same 3,000-sentence-pair sample of the English and French WHO AFI corpus using the baseline methodology. The same test set of 1,000 unseen SL input sentences was also translated. Figure 49 shows the distribution of word errors, for each translation, measured by the LD between the translation produced and the reference translation. Results for both the actual and optimal performances are shown. The LD score represents, for each translation, the number of operations (substitutions, insertions and deletions) required to convert the translation into the reference translation. As a consequence, the number of translations with an LD score of 0 matches the number of translations that were 100% accurate in the experiment undertaken in Figure 45. 73 translations were produced without errors (actual performance) and a total of 79 translations *could* have been produced without errors (optimal performance). The two distribution curves indicate, as expected, that the optimal performance produces a greater number of translations with a small number of errors (low LD) and fewer translations with a large number of errors (high LD).

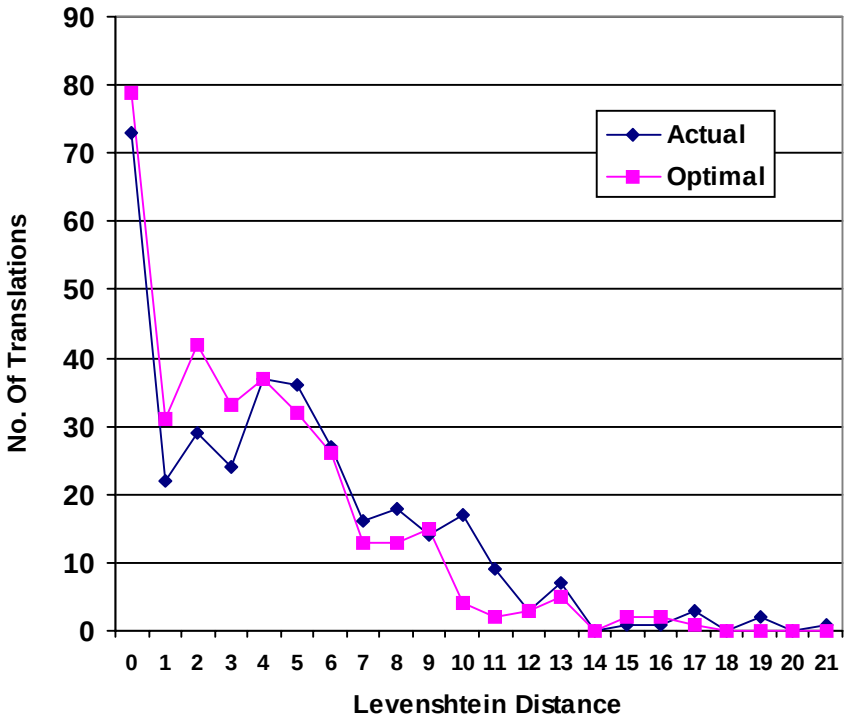

Figure 49: Distributions of Word Errors Measured by Plain Levenshtein Distance

5.2.4 Co-occurrence and Frequency Threshold

In Chapter 3, it was stated that strings that co-occur in a minimum of two translation examples are likely to be translations of each other. This formed the axiom of the translation-pattern extraction phase (3.2). The manual evaluation of the translation patterns extracted from a 3,000-sentence-pair sample of English and Spanish in McTait & Trujillo (1999), summarised in section 5.1, suggested that as the string co-occurrence threshold is increased, the accuracy of the translation patterns increases. The following experiment is designed to test this hypothesis. Using the baseline methodology, translation patterns were extracted from the same 3,000-sentence-pair sample of the English and French WHO-AFI corpus, as in previous experiments. The same test set of 1,000 unseen SL input sentences was also translated. However, the experiment was repeated so that the string co-occurrence threshold was changed from a minimum of 2 to 3, 4 and even 5. Figure 50 shows the distribution of results when translating the test set with the translation patterns extracted for each value of the string co-occurrence threshold.

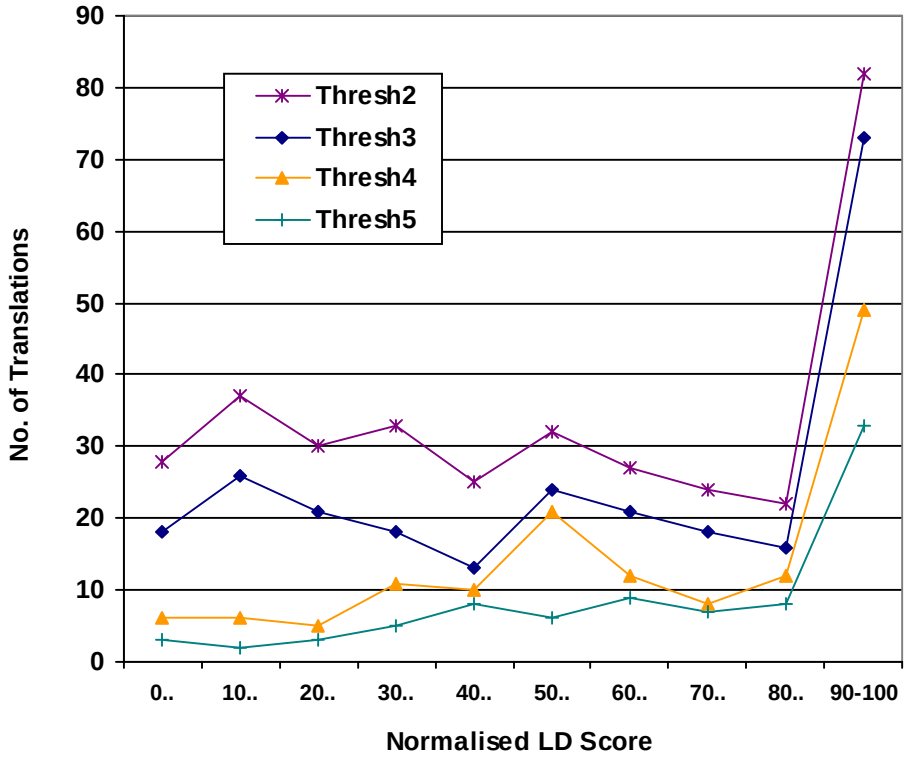

Figure 50: Effect on Performance by Changing the String Co-occurrence Threshold

The effect on performance of changing the string co-occurrence threshold is clear. The result of increasing the threshold is that fewer low-scoring translations are produced. However, fewer high-scoring translations are also produced. In fact, as the threshold increases, fewer translations in general are produced. The rate of recall drops from 34%, when the threshold is set at 2, to 8.4% when the threshold is raised to 5. The fact that fewer translations are produced is indicative of the fact that increasing the threshold increases the amount of constraints placed on translation-pattern extraction. However, the rate of precision of the translations increases as the threshold increases. Precision, defined as the number of high-scoring translations produced divided by the total number of translations produced, jumps from 24.1%, when the threshold is set at 2, to 39.2%, when the threshold is set to 5.

Threshold	Translated	Recall (%)	High-Scoring	Precision (%)
2	340	34.0%	82	24.1%
3	248	24.8%	73	29.4%
4	140	14.0%	49	35.0%
5	84	8.4%	33	39.2%

Table 3: Recall and Precision when Changing String Co-occurrence Threshold

5.2.5 Partial Translations

In the experiments where only full translations are considered, the rate of recall is improved by increasing the size of the training set (section 5.2.2). More text in the training set provides a greater range of lexical phenomena, increases the chance of string co-occurrence and therefore the number of translation patterns extracted. A larger number of translation patterns is conducive to increasing coverage. The addition of morphological analysis has also been shown to increase the rate of recall (4.6). One further method of increasing the rate of recall, but at the cost of accuracy, is to include partial matches between the SL input and the translation patterns, as shown by the experiments undertaken in section 5.2.2.

In order to demonstrate the behaviour of partial translations, the translation patterns extracted from the 3,000-sentence-pair sample of the English and French WHO

AFI corpus using the baseline methodology, were used to translate the same test set of 1,000 unseen SL input sentences. This time, partial translations were considered acceptable in the cases where it was not possible to produce full translations. Intuitively, more translations were produced. A total of 844 translations were produced as opposed to the 340 produced when only full matches were considered. As can be seen from the distributions of results in Figure 51, where partial translations are included (Full+Partial), many more low-scoring as opposed to high-scoring translations were produced. The results of this experiment confirm the intuitive fact that including partial translations results in a greater number of translations, albeit low-scoring ones, than when only full translations are considered.

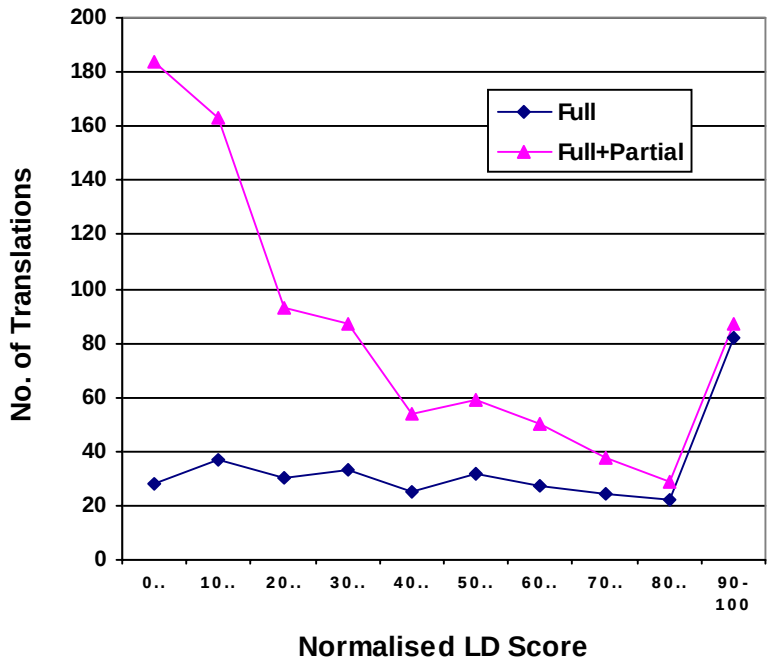

Figure 51: Comparison of Distributions of Results for Full and Partial Translations

5.2.6 Recursive Matching

The recursive matching algorithm in the recombination phase (3.3.3) is designed to retrieve TL equivalents of SL text fragments bound to the SL variables of the base pattern in cases where there were no direct matches with single SL fragments in the remaining set of translation patterns. Including recursive matching ensures higher recall than

without, even though it incurs a greater processing overhead. Recursive matching also introduces a greater amount of noise or entropy into the translation channel, as there is a greater scope for error. On account of this increased likelihood of error, translations formed as a product of one or more recursive matches are likely to be low-scoring.

To test this hypothesis, the translation patterns extracted, using the baseline methodology from the same 3,000-sentence-pair sample of the English and French WHO AFI corpus, were used to translate the same set of 1,000 unseen SL input sentences. This time, the recursive matching algorithm was excluded from the recombination phase so that translations were only formed from direct matches between SL fragments bound to the SL variables of the base pattern and single SL fragments in the remaining set of translation patterns. A comparison was made between the translations where recursive matching was included (WithRecursion) as in previous experiments and where it was excluded (NoRecursion). Figure 52 shows the distributions of results for the translations produced by both methods.

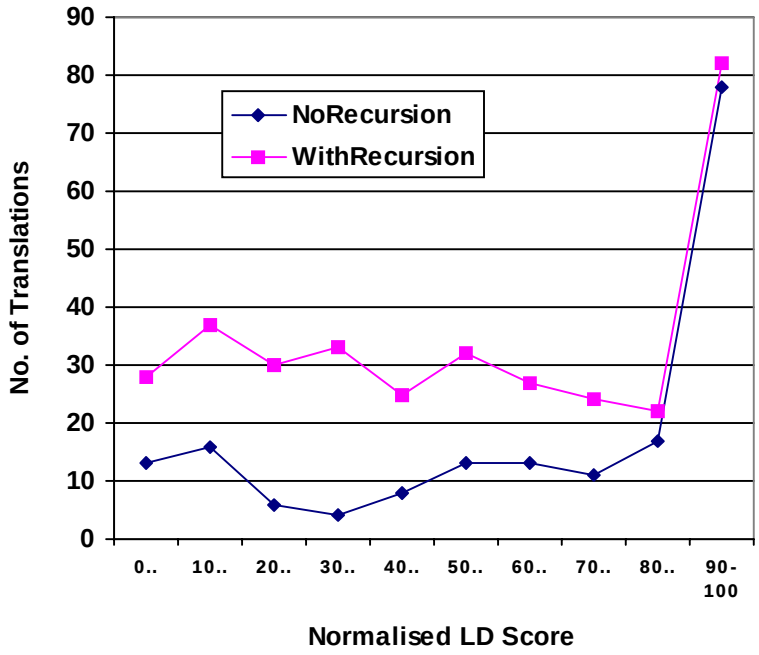

Figure 52: Comparison of Distributions with and without Recursive Matching

As expected, fewer translations were produced when recursive matching was excluded. A total of 179 SL input sentences were translated as opposed to the 340

translations produced when recursive matching was included. As can be seen from the distributions of results, including recursive matching produces a greater number of low-scoring translations. This is due to the fact that recursive matching introduces a greater amount of noise into the translation channel. However, as can be seen from the graph, a very small number of extra high-scoring translations were produced when recursive matching is included. A total of 82 high-scoring (90-100% accurate) translations were produced when recursive matching was included as opposed to the 78 that were produced when it was excluded.

A further experiment was designed to test the hypothesis that, although it increases the likelihood of producing a translation for each SL input sentence, recursive matching adds an amount of noise to the translation process which blurs, to some extent, the improvements to the baseline methodology made by incorporating linguistic knowledge. The experiment in 4.6, where a comparison was made between each variant of the methodology, is repeated. This time recursive matching is excluded from the recombination phase. The results are presented in Figure 53.

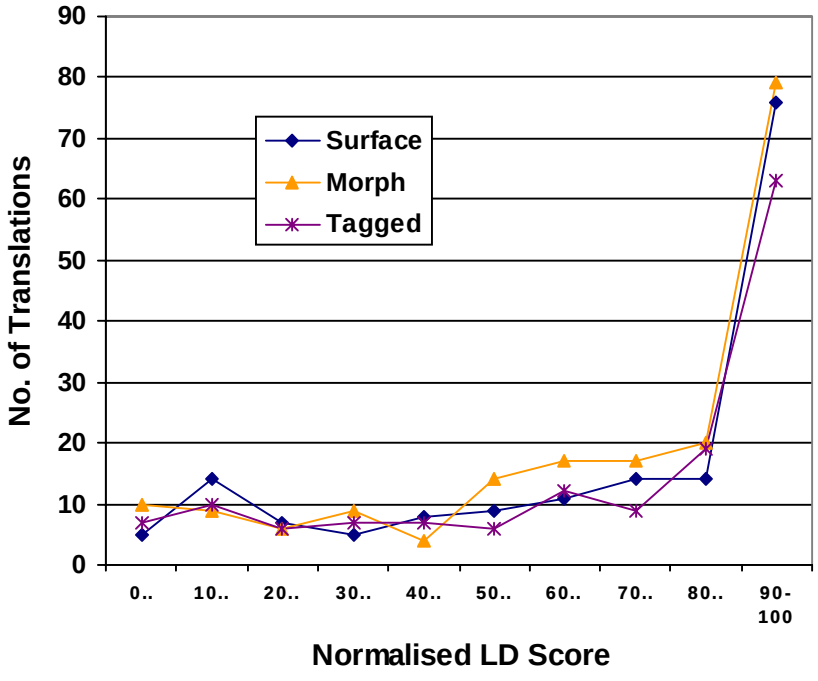

Figure 53: Comparison of each Variant Excluding Recursive Matching

The graph shows that there is much less variation between the distributions of results for each variant. However, the variant incorporating morphological analysis improves upon the baseline variant much more clearly than in 4.6 as a greater number of translations were produced that were 50-100% accurate. The morphological variant also improved upon the baseline approach in terms of recall and the number of high-scoring translations produced. A total of 185 SL input sentences were translated by the variant including morphological information as opposed to the 163 translations produced by the baseline methodology. Furthermore, 79 as opposed to 76 high-scoring translations were produced by the morphological variant. The variant augmented further with POS annotation again performed less well than the baseline approach. Fewer SL input sentences were translated and fewer high-scoring translations were produced.

5.2.7 New Text-Types

The experiments undertaken so far have been carried out by extracting translation patterns from only one corpus, the WHO AFI corpus of titles, and therefore one text-type. In order to test the portability of the methodology to new text-types and genres, further experiments were carried out using a different corpus. The corpus chosen for these experiments was the Xerox *ScanWorX* user-manual. *ScanWorX* is an application that performs optical character recognition for scanners. It is therefore considered to represent the text type 'software manual'. Software manuals, as instances of technical texts, have been used successfully in many previous EBMT experiments (Collins & Cunningham, 1997; Veale & Way, 1997; Meyers et al. 1998). The *ScanWorX* user manual is provided in English and French and has been sentence-aligned. It is distributed as part of the BAF corpus (Simard, 1998).

Translation patterns were extracted from a 600-sentence-pair sample of the Xerox *ScanWorX* user manual. A total of 9,274 translation patterns were extracted. Memory restrictions and complexity (see Chapter 4) prevented the use of a larger sample. The translation patterns extracted were, on average, longer than those extracted from the WHO AFI corpus. The most frequent number of SL and TL text fragments per translation pattern – p and q in Figure 12 – is 4 and 5 respectively. This is greater than that of the

translation patterns extracted form the WHO AFI corpus. Furthermore, the average number of lexical items contained in the translation patterns extracted from the WHO AFI corpus is 18, whereas the average number of lexical items contained in the translation patterns extracted from the Xerox corpus is 34, almost twice that figure. A test set of 500 unseen SL input sentences was selected at random from the remaining sentence-pairs in the Xerox corpus. Although 9,274 translation patterns were extracted, a figure comparable to the experiments in section 5.2.1, only 26 of the 500 SL sentences in the test set (approximately 5%) were translated. However, 13 of the 26 translations, half of those produced, were 100% accurate.

Despite the precision of the translations produced using the Xerox corpus, it is clear from the disappointing rates of recall that more training data is required in order to reach a satisfactory level of coverage. With more training data, the methodology can be effectively ported to new text-types and domains. Translation patterns were extracted from a relatively small sample of the Xerox corpus since it is much larger, in relative terms, than the WHO AFI corpus. In more detail, a sample of 3,000 sentence pairs was taken from the Xerox and the WHO AFI corpora. The WHO AFI corpus was found to contain 47,129 word forms, whereas the Xerox corpus contained 68,820, almost 1.5 times as many word forms. The sentences in the Xerox corpus were generally greater in length than those of the WHO AFI corpus. This is why the translation patterns extracted from the Xerox corpus were generally longer than those extracted from the WHO AFI corpus and why such a small sample was used. Figure 56 shows a distribution of the sentence lengths of the SL (English) sentences in the WHO AFI and Xerox corpora.

The graph shows that the sentences in the WHO AFI corpus are skewed towards short sentence lengths and that there are relatively few long sentences. On the other hand, the Xerox corpus has less short sentences and has more longer sentences i.e. those up to approximately 30 words long. As the Xerox corpus is larger, in comparative terms, than the WHO AFI corpus, the collocation trees constructed during translation pattern extraction (3.2.1) become very large when a much smaller sample of the corpus is used as training data. This restriction prevents the effective scaling-up of this approach and the use of larger training sets when using the Xerox and other corpora.

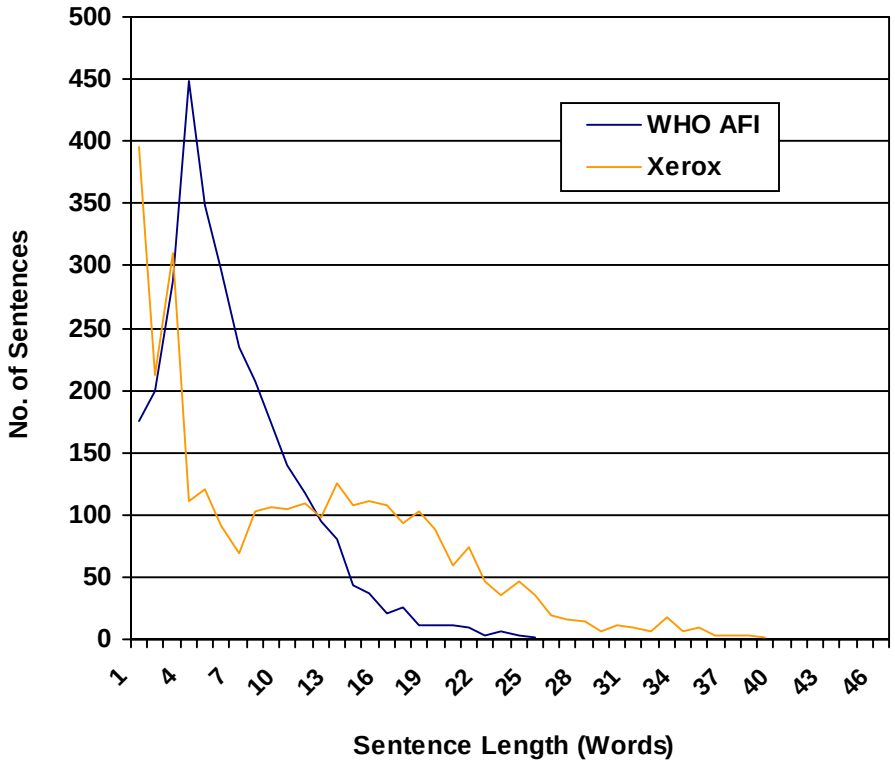

Figure 54: Distributions of SL Sentence Lengths in Xerox and WHO AFI Corpora

5.2.8 English-Spanish Experiment

The experiments undertaken so far have been based on translating unseen English sentences using translation patterns extracted from parallel sentence-aligned English and French corpora. As the methodology outlined in Chapter 3 is essentially language-neutral in nature, apart from the stop lists of closed-class words and information for tokenisation and reassembly, experiments are possible using bilingual corpora between new language pairs. Consequently, this section investigates the performance of the methodology on a different language pair, namely English and Spanish.

A comparable English-Spanish bilingual corpus was used to translate unseen English sentences. The WHO AFI English and Spanish corpus was chosen as a comparable corpus, although it does not share exactly the same sentences as those contained within the English-French WHO AFI corpus. A stop-list of Spanish function words and tokenisation and reassembly modules were rapidly constructed, as they contain

minimal amounts of linguistic information. As in the English-French experiments, no clitic expressions were expanded or reassembled. Translation patterns were extracted from a 3,000-sentence-pair sample of the English-Spanish WHO AFI corpus. From the remaining set of sentence pairs in the corpus, 1,000 were randomly selected and used as a test set of unseen SL input sentences. The distribution of results for the actual and optimal performance are shown in Figure 55.

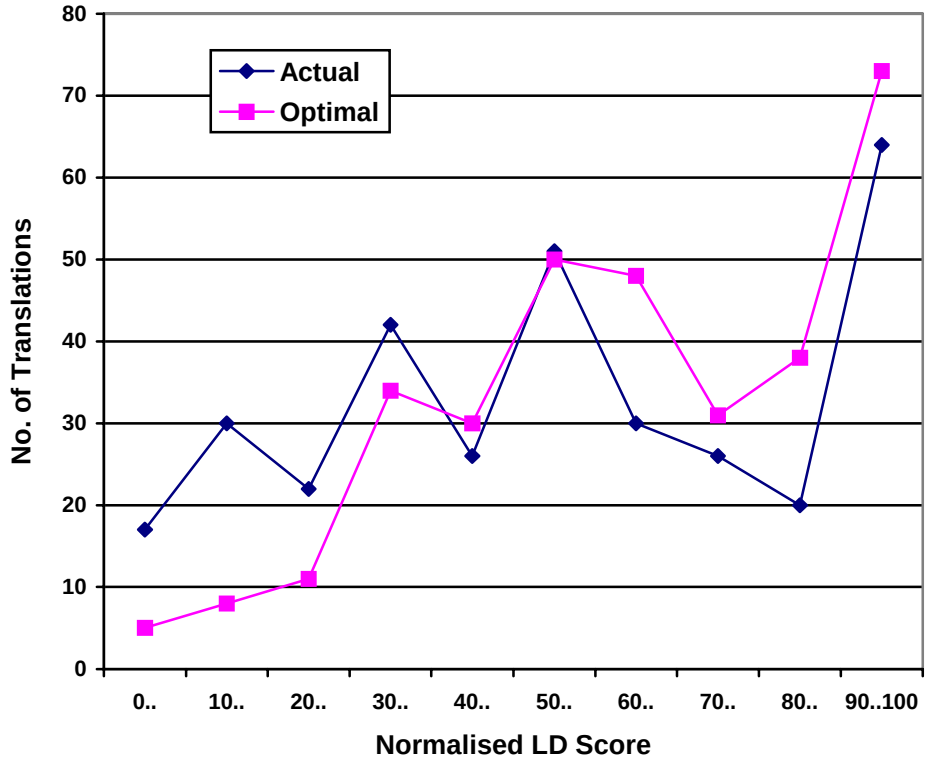

Figure 55: Comparison of Actual and Optimal Performance for English and Spanish

A total of 10,887 translation patterns were extracted, an amount comparable to that extracted during the English-French experiments (Figure 45). The most frequent values of *p* and *q* (Figure 12), or the number of text fragments or variables per translation pattern was 2 and 3, as it was for the English-French translation patterns. However, in comparison with the experiments undertaken for English and French, the results for English and Spanish are slightly less impressive. A total of 328, as opposed to 340, SL sentences were translated. Furthermore, fewer high-scoring translations were produced. A

total of 64 high-scoring (90-100% accurate) translations were produced, 59 of which were 100% accurate, as opposed to the 82 high-scoring translations, 73 of which were 100% accurate, produced during the English-French experiments. In addition, there would appear to be a greater number of mid-scoring translations produced, i.e. those that were around 50% accurate, than there were as a result of the English-French experiments.

5.2.9 Comparison with Commercial MT

It is difficult to compare this methodology with other approaches to EBMT for a number of reasons. The most notable include differing evaluation methodologies, access to existing implementations, different text types and language pairs, differing sizes of training sets etc. However, it would be interesting to gauge the difference in performance between this approach and a commercially available MT system. However, the test may seem unfair as any approach to EBMT is tuned to a certain text type (one of its main advantages) and commercially available MT systems are often more robust, capable of translating texts from a wider range of text-types and domains, with the effect that recall is often greater.

The commercial MT system chosen for this comparison was the Babelfish / Systran translation system, publicly available on the Alta-Vista search engine[15] and the Systran portal.[16] The same set of 1,000 unseen English sentences used in previous English-French experiments, such as that in Figure 45, was used when translating with the English-French Babelfish system. However, as Babelfish is a robust wide-coverage translation system, it can achieve around 100% recall for the test set. On account of its large bilingual dictionaries and robust methods of handling unknown words (transposing the SL items into the TL text for instance), Babelfish far outperforms this approach to MT in terms of recall. Therefore, only the 340 SL sentences from the original 1,000-sentence test set that were able to be translated by the methodology presented in this thesis (see section 5.2.1) were passed to Babelfish for translation. This enables a comparison to be made between the rates of precision of the two translation systems. A

[15] http://www.alta-vista.com

[16] http://www.systransoft.com

comparison of the distributions of results between Babelfish and the MT methodology presented in this thesis (marked as Thesis in the legend) is given in Figure 56.

The graph clearly shows that, for the test set of 340 SL input sentences, this methodology far outperforms Babelfish in terms of the number of high-scoring translations produced. In fact, Babelfish produced a greater number of low-scoring translations generally and far fewer high-scoring translations than the MT methodology presented in this thesis. Babelfish produced only 5 translations that were 90-100% accurate as opposed to the 82 that were produced by this approach. However, Babelfish is more robust and can translate a far greater percentage of SL sentences in the original test set of 1,000 sentences than this approach. However, if partial matches are considered acceptable, then this methodology approaches the rates of recall obtained by Babelfish.

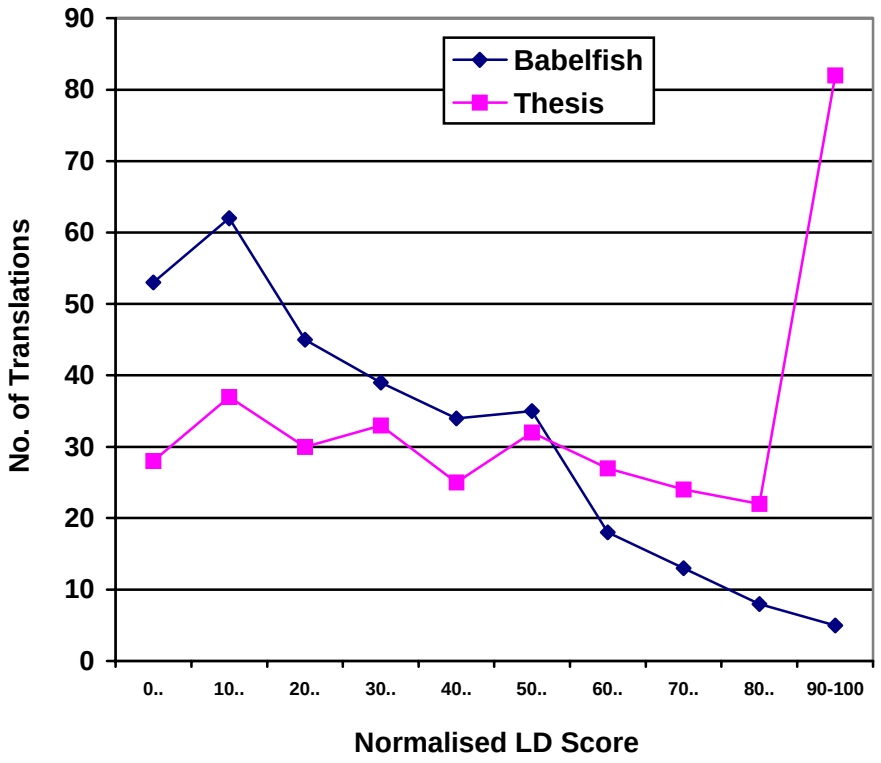

Figure 56: Distributions of Results between Babelfish and Baseline Methodology

5.3 Summary

Section 5.1 of this chapter has attempted to show the accuracy of the translation patterns extracted from bilingual sentence-aligned corpora. Manual evaluations of the translation

patterns and the alignments, although subjective, revealed that accuracy is improved when the co-occurrence threshold is increased, and when alignments in the translation patterns are of a simple or bijective nature. The second-pass algorithm was also considered to be effective in discovering non-adjacent alignments, even though for closely related languages such as English and French, non-adjacent alignments, representing structural differences, are relatively rare.

A more effective method of evaluating translation patterns, and of course the algorithms involved in recombination, is the automatic procedure where unseen SL input sentences are translated using the translation patterns extracted. The translations produced are evaluated automatically by comparing them with the reference translations given in the corpus. This has been carried out in the experiments in both Chapter 3 (3.4), Chapter 4 (4.6) and this chapter. The experiments in this chapter were aimed at investigating the parameters involved in extracting and recombining translation patterns to produce TL translations, in order to identify those parameters that contribute to improving performance. Furthermore, the upper bound of the methodology was investigated, as in section 5.2.1. Such experiments showed that actual performance was not much less impressive than optimal performance, particularly when there was no recursive matching in recombination (section 5.2.6). However, further investigation into assigning translation confidence is necessary in order to close the gap further.

The intuitive trend, where increasing the size of the training set increases recall and accuracy, was revealed in section 5.2.2. The distribution curves for full translations show steadily increasing rates of recall and precision, indicating the need for larger training sets. However, this requirement is at odds with the complexity issues, in particular the size of the collocation trees constructed during translation pattern extraction, as described in section 4.5.1. However, high recall is rapidly achieved by including partial translations but comes at the expense of precision. Given the same size training set, recall is also increased, if only minimally, by incorporating morphological analysis. As all morphological variants are grouped under their lemma, there is a greater chance of string co-occurrence, hence more translation patterns and greater recall.

The precision of the translations produced is also improved by increasing the co-occurrence threshold, as described in the experiments in section 5.2.4. However,

increasing the threshold provides tighter constraints in translation pattern extraction with the consequence that the rate of recall decreases. Precision is also increased by increasing the size of the training set. However, there are very real limits when increasing the amount of training data. Incorporating morphological analysis into the methodology does improve accuracy by producing more high-scoring translations for the same size training set, but comes at the cost of increased complexity and an increased processing overhead. When recursive matching is included in the recombination phase, a few extra high-scoring translations are produced as opposed to when recursive matching is excluded. But according to the experiments in section 5.2.6, although recursive matching increases recall, the vast majority of the translations produced are low-scoring.

Due to the limitations on the size of the training set, it has been difficult to gauge portability to new text-types effectively. Due to the relative size of the Xerox corpus, only a small training set was able to be used, resulting in low recall. However, the precision of the translations produced was encouraging. More encouraging were the results of porting the methodology to new language pairs. The experiments performed on the English-Spanish WHO AFI corpus in section 5.2.8 produced rates of recall and precision comparable to those produced using the English-French WHO AFI corpus.

Finally, comparing this methodology with Babelfish proved that the low rates of recall produced by this methodology must be improved in order to compete with robust, commercial MT systems which achieve close to 100% recall. However, when comparing the rates of precision returned using the test set of sentences that this methodology was able to translate, it outperformed the Babelfish/Systran system.

In conclusion, the variant of this methodology that performed most effectively was the variant with morphological analysis incorporated. Given the same size training set, the rates of precision and recall improve upon those of the baseline methodology, but at the cost of an increased processing overhead and increased person-labour to create the resources necessary. Including POS annotation does not improve the performance of the baseline methodology on account of the extra constraints. While recursive matching is necessary to maintain recall, it also increases the amount of noise in the translation channel. Including partial matches also improves recall but at the cost of precision. Increasing the size of the training set remains the largest and most important obstacle to

the scalability and portability of this approach and improved rates of recall and precision. Real improvements to recall and precision are only possible when the amount of training data is increased.

6 Conclusions and Future Work

6.1 Conclusions

This thesis has attempted to show that it is possible to create an EBMT system based on the extraction and recombination of translation patterns extracted from a bilingual corpus using language-neutral strategies. The translation patterns are extracted by means of a recursive machine-learning algorithm based on the principle that strings that co-occur in two or more translation examples are likely to be translations of each other. Despite the relative simplicity of this axiom, it has been shown that useful and linguistically meaningful translation patterns can be extracted without the need for significant linguistic knowledge sources. Furthermore, a number of accurate translations are produced when translating unseen SL input sentences using the translation patterns extracted.

Although the methodology is essentially language-neutral in nature, an amount of external linguistic knowledge was added in order to reduce the likelihood of false translations, increase coverage and improve the accuracy of the translations produced. It was revealed that incorporating morphological analysis did improve the rates of recall and precision slightly compared to those produced using the basic language-neutral methodology. However, updating the methodology further to include POS annotation did not improve upon the baseline methodology.

A language-neutral approach to extracting and recombining translation patterns is advantageous in that it is portable to new language pairs. This was demonstrated by the English-Spanish experiments in section 5.2.8. Moreover, the creation of linguistic resources is a time-consuming, expensive and error-prone task. Where linguistic knowledge has been added, in this case morphological analysis and POS tagging, it has been kept to a minimum and to the sort of knowledge that is either widely available or relatively easy to produce. Furthermore, the fact that strings must co-occur in just two translation examples in a bilingual corpus is advantageous in that the translation-pattern extraction algorithm is useful in instances of sparse data. This makes it ideal for less-studied or minority languages where there are limited amounts of resources. The co-

occurrence threshold is also adjustable to allow for more accurate translation patterns, if large amounts of text exists.

The two-pass algorithm (3.2.3.2) developed to align the text fragments and variables allows the computation of a variety of local alignment types to represent a range of linguistic phenomena. In addition to simple bijective alignments, non-bijective alignments are computed which are able to represent, as an example, the translation of phrasal verbs in English to languages such as French (Figure 1). The second-pass alignment algorithm (Figure 29) also enables simple structural divergences to be captured efficiently.

On account of the absence of significant linguistic knowledge in the methodology, such as parsers and bilingual dictionaries, translation patterns are extracted on the basis of the distributions of surface forms in a corpus. Consequently, they are shallower representations of bilingual phenomena and simply represent generalisations of sentences that are translations of each other. They resemble transfer rules in RBMT or knowledge-intensive approaches to EBMT (2.1.2), although they do not contain any constraints determined by some formal linguistic theory. As a result of this lack of constraints, for each SL input, recombination produces many translation solutions that are incorrect or ill-formed according to the syntax of the TL. This is inefficient and requires a high degree of accuracy from the mechanism by which the score of translation confidence is assigned to each possible translation solution. Relying on surface criteria, such as bigram models, to select the best translation proves only partially successful.

Despite the advantages gained by incorporating morphological information into the methodology, one has to ask whether the effort involved in creating the resources is worth the increase in recall and precision. The improvements made by adding morphological information are very slight, particularly with the inclusion of recursive matching in recombination. Creating the resources required for morphological analysis / generation incurs a cost in terms of effort, person-hours, and an increased processing overhead. The resources are also error-prone. Therefore, it may not be worth the extra increase in performance. Certain knowledge sources, POS tagging in this case, do not improve results due to a combination of increased constraints and tagger errors.

There are many reasons as to why there is only a slight increase in performance when morphological information is included. First, only a relatively small amount of linguistic knowledge is added. Second, the nature of the WHO AFI corpus, i.e. titles, is such that are there generally very few morphological variants for each lemma. There tends to be a higher density of uninflected forms and proper nouns in titles than in ordinary texts. Third, the noise incurred by recursive matching blurs, to some extent, the improvements made by adding morphological information. Finally, the training set may be too small to obtain significant improvements.

In order to effectively improve the performance of the methodology, without extra effort, cost or complexity, it is clear that larger training sets are required. It is intuitive, and the experiments in Chapter 5 confirm, that increasing the size of the training set improves the rates of recall and precision. However, scaling up this approach represents a fundamental challenge. As the size of the corpus increases, the size of the collocation trees becomes very large (Figure 43). Therefore, there are very real limits to the scalability of this approach in its current form. Therefore, alternative strategies are required if this approach is to be fully scalable and portable to new text types, language pairs and larger corpora. Larger training sets are also necessary in order to improve the currently low rates of recall if this approach is to be comparable to other approaches to MT, including commercial systems such as Systran.

6.2 Future Work

This section aims to address some of the shortcomings of this approach to EBMT, as discussed above, and discuss alternative strategies. Additionally, further applications of the methodologies presented in this thesis are investigated, particularly the translation-pattern extraction algorithm.

6.2.1 Alternative Strategies

The most pressing improvement to the methodology required is the improvement of precision and recall by scaling up the approach to use larger training sets to consist of at least 1 million words, rather like the EBMT experiments undertaken by Brown (1996, 1997, 1998, 1999, 2000) and Frederking & Brown (1996). This would enable the

methodology to be easily ported to new text types, domains and language pairs, enabling fair comparisons with existing MT systems.

The key to extracting translation patterns from larger training sets by the current methodology lies in reducing the size of the collocation trees (Figure 43). Currently, each single lexical item or token that appears in two or more sentences is added to the tree. Each lexical item may be added as the daughter of more than one node, and as collocation nodes are built up incrementally, token by token, the collocation trees frequently contain many levels of non-terminal nodes containing possibly numerous repeated instances of each lexical item added. Reducing the amount of repetition and the number of levels of non-terminal nodes is conducive to reducing the size of the trees. If the size of the trees cannot be reduced, efficient use of hard-drive space or databases with efficient indexing and retrieval mechanisms could be employed. Alternatively, the source code could be ported from Java, the language in which this implementation is built, to a different programming language, such as C/C++, which is more conducive to utilising hard-drive space when virtual memory is in short supply.

Alternative strategies to tackling the space complexity problem include filtering function words from collocation tree construction as they are of such high frequency that they increase the size of the trees significantly. However, this is unsatisfactory on account of the fact that meaningful phrases or terms contain function words. Therefore, this would involve *ad hoc* methods of reintroducing them at a later point to make the text fragments corresponding to the leaf-node collocations more meaningful. A second, more effective strategy is to reduce the number of levels of non-terminal nodes in the trees by adding, not single lexical items incrementally, but adding the longest possible phrases, or continuous multiword units that co-occur in two or more sentences and allowing them to combine to form longer collocations in the usual manner. The longest co-occurring continuous word sequences in a monolingual corpus may be identified by established algorithms such as that of McCreight (1976) which finds the longest recurrent sub-strings within a main string. This strategy was initially employed, but was found to ‘miss’ certain collocations. Therefore, it was abandoned in favour of the more provable, but space-consuming, method of adding single lexical items.

The current two-stage methodology, where translation patterns are extracted from the entire corpus of SL and TL sentences, could be modified so that translation patterns are computed dynamically during recombination, rather than off-line in an extraction phase. The SL sentences – and their TL equivalents – containing the lexical items of which the SL input was composed would be retrieved and it is from these selected sentences that collocation trees would be constructed to form the relevant translation patterns.

A more radical solution to the space bottleneck problem incurred by the construction of collocation trees would be to employ a different translation pattern extraction strategy altogether that did not attempt to build collocation trees at all. Strategies employed in previous attempts in the EBMT literature to extract translation patterns by language-neutral methods could be utilised (2.1.3.1). For example, an extraction method similar to those of Güvenir & Cicekli (1998), Echizen-ya et al. (2000) or Malavazos & Piperidis (2000) where similarities and differences between pairs of SL and TL sentences are compared could be used, where the similarities become the text fragments and the differences become the variables. An algorithm of quadratic time complexity is envisaged where each SL sentence is compared with each remaining SL sentence of a corpus and similarities and differences are computed between the two SL sentences, and the two corresponding TL sentences, possibly by DP, to produce translation patterns. Such an algorithm would be expensive in terms of time, but not space. The text fragments and variables of the translation patterns produced would subsequently be aligned using the same two-pass alignment algorithm. Recombination would then proceed as usual.

Inspection of the translation patterns and comments from the panel of judges who carried out the manual evaluations revealed that it is the segmentation of the translation examples into meaningful translation patterns, rather than the alignments computed between the text fragments and variables, that is an area that requires improvement. Techniques based on the distributions of surface forms are only effective up to a certain point and are more error prone than knowledge-intensive segmentation algorithms that employ structural information (2.1.2). The incorporation of structural information into translation-pattern extraction is an area worth investigating. Knowledge-intensive

techniques could be employed where a POS tagger and a small Context Free Grammar for each language could be used for shallow parsing. Obviously, this entails greater knowledge sources with consequences for portability, cost, labour and increased risk of error introduction. On the other hand, the literature on unsupervised learning of structure could be investigated. In fact, the algorithms of van Zaanen (2000) are very close to the principles behind this approach to translation pattern extraction and those in 2.1.3.1 where similar items in two SL sentences, and the corresponding TL sentences, are coupled to form the text fragments of translation patterns.

Improvements to the recombination phase are also possible. The lack of constraints in the translation patterns frequently leads to the production of numerous translation solutions per SL input. Once all translation solutions have been produced, one is selected according to a score of translation confidence. A best-first approach to producing translations is preferable and more efficient in that spurious and incorrect translation solutions are not produced. For example, the best TL equivalent for each local alignment of variables in a base pattern could be retrieved – using criteria similar to those already used in the translation confidence score such as frequency, amount of overlap etc. – and bound to the relevant TL variable positions in the base pattern, so that only one translation solution, presumably the best, is produced.

Alternative methods of evaluating the translations produced are also possible and even conducive. DP, as pointed out in 3.2.3, is unable to compute alignments between non-adjacent items. Therefore, when comparing the translations produced with the reference translations, LD does not take into account any correct lexical items inserted into ‘distant’ positions. This makes the current evaluation reasonably strict in that correct lexical items found in the wrong position in a translation solution may remain ignored in the final score. Furthermore, although the current evaluation methodology is automatic, and therefore consistent, it is possible that some of the lower-scoring translations may be ‘correct’ or useful according to some other criteria such as acceptability as raw MT output to a post-editor. Usefulness of the translations produced could be evaluated by the amount of time or effort required to revise it to the required standard, as in Minnis (1994) or Whyman & Somers (1999). Alternatively, one could use any of the various

methodologies in the MT evaluation literature that are based on human observations and scales of correctness (Arnold et al., 1993).

Some of the claims made in this thesis require further testing. Examples include the claim that the methodology is language-neutral and portable to new language pairs, is useful in instances of sparse data and capable of computing alignments between structurally divergent languages. Further experiments are envisaged on new text-types and new language pairs, particularly minority or less-studied languages or distant language pairs. Extracting bilingual corpora in new language pairs from the Word Wide Web (Resnik, 1998, 1999) may provide the necessary resources to test these claims. The EBMT methodology presented in this thesis was created with languages such as English, French and Spanish, which are closely related, in mind. Therefore, changes to the methodology may be required when considering new language pairs, particularly if Asiatic languages are to be considered.

6.2.2 EBMT vs. TM

It has been stated that the rates of recall returned by this EBMT methodology are relatively low. Unless the methodology is modified to be able to extract translation patterns from larger training sets, its potential as a full-scale EBMT system is questionable. However, if one considers the high rates of recall when partial matches are included, one has to wonder whether this methodology is more applicable as a 'flexible TM system' where a greater number of full translations are produced by incorporating techniques from EBMT, i.e. by recombining text fragments below the level of the sentence from more than one translation example. This new generation of TM systems is currently an emerging and active area of research (Langé et al., 1997; Planas & Furuse, 1999; McTait et al., 1999; McTait, 2000; Carl & Hansen, 2000). However, the distinction between TM and EBMT remains: TM is merely a translator's aid that proposes matches, whereas EBMT is an automatic translation system capable of producing new translations. Therefore, these newer TM systems effectively become EBMT systems requiring only the database of translation examples as the principle knowledge source.

The experiments in 5.2.1, where a training set of 3,000-sentence-pairs of French and English was used, revealed that from the test set of 1,000 unseen SL input sentences,

there were 36 direct matches between the SL sentences in the test set and the SL sentences in the training set. Using this methodology, a total of 340 full translations was produced. This represents a significant increase in the number of full matches compared to that capable by a traditional TM system. When recursive matching, which produces a number of low-scoring translations, is excluded from recombination, a total of 179 full translations were produced. Even compared to the number of 100% correct full translations produced, this methodology still improves upon the figure of 36 direct and therefore 100% correct translations produced by a traditional TM system. For example, from the 340 translations produced, 73 were 100% accurate according to the reference translations.

6.2.3 Further Applications

The algorithms developed in this thesis may also be of use to other NLP applications. The algorithm developed to extract collocations from a set of SL or TL sentences is particularly applicable to new domains. On account of the repetitive nature of text corpora, the algorithm essentially finds recurrent possibly non-contiguous sentential patterns in texts. This makes the algorithms useful for authoring, correction or language standardisation tools. Assuming that the algorithm is trained on monolingual texts of a specific genre to extract sentential patterns, an authoring tool could work as an auto-complete function in a word processing package. As the author begins typing a phrase, possible continuations, based on the patterns extracted, could be retrieved to aid the author, who may not be a native speaker of the given language, to complete the phrase or sentence. The advantage over existing auto-complete functions is that discontinuous phrasal or sentential patterns would be suggested. The extraction of discontinuous patterns may also be useful in diverse fields such as terminology extraction and topic identification (keyword extraction) for information retrieval. All potential applications would share the same advantage in that they would be portable to new languages.

Furthermore, outside the field of computational linguistics, the same algorithms may be useful in non-textual applications in the fields of machine-learning and artificial intelligence. Given a set of comparable data, the algorithms are able to extract possibly

discontinuous patterns which, in some way, describe recurrent phenomena within the data.

Bibliography

Ahrenberg, L., M. Andersson & M. Merkel (1998) A Simple Hybrid Aligner for Generating Lexical Correspondences in Parallel Texts. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 29-35.

Akiba, Y., M. Ishii, H. Almuallim & S. Kaneda (1995) Learning English Verb Selection Rules from Hand-made Rules and Translation Examples. *Proceedings of the Sixth Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium, 206-220.

Al-Adhaileh, M. H. & T. E. Kong (1998) A Flexible Example-Based Parser Based on the SSTC. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal Canada, 687-693.

Al-Adhaileh, M. H. & T. E. Kong (1999) Example Based Machine Translation Based on the Synchronous SSTC Annotation Schema. *Machine Translation Summit VII*, Singapore, 244-249.

Almuallim, H., Y. Akiba, T. Yamazaki, A. Yokoo & S. Kaneda (1994) Two Methods for Learning ALT-J/E Translation Rules from Examples and a Semantic Hierarchy. *COLING-94*: *The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 57-63.

Alshawi. H, S. Bangalore & S. Douglas (2000) Learning Dependency Translation Models as Collections of Finite State Head Transducers. *Computational Linguistics* **26**, 45-60.

Andriamanankasina, T., K. Araki & K. Tochinai (1999) Example-Based Machine Translation of Part-of-Speech Tagged Sentences by Recursive Division. *Machine Translation Summit VII*, Singapore, 509-517.

Antworth, E. L. (1990) PC-Kimmo: A Two-Level Processor for Morphological Analysis. Technical report, Summer Institute of Linguistics, Dallas, Texas. `http://www.sil.org/pckimmo/pc-kimmo.html`

Arnold, D., R. Lee Humphreys & L. Sadler (1993) Evaluation: An Assessment. *Machine Translation* **8**, 1-24.

Baum, L. E. (1972) An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process. *Inequalities* **3**, 1-8.

Benson, M. (1990) Collocations in General Purpose Dictionaries. *International Journal of Lexicography* **3**, 23-35.

Bescherelle, L. N. (1986) Complete Guide to Conjugating 12000 French Verbs (English Edition). Paris: Hatier.

Bod, R. (1992) A Computational Model of Language Performance: Data Oriented Parsing. *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, COLING-92*, Nantes, 855-859.

Boutsis, S. & S. Piperidis (1998a) Aligning Clauses in Parallel Texts. *Proceedings of the Third Conference on Empirical Methods in Natural Language Processing*, Granada, Spain, 17-26.

Boutsis, S. & S. Piperidis (1998b) OK with Alignment of Sentences. What About Clauses? *Proceedings of the Panhellenic Conference on New Information Technology-(NIT'98)*, Athens, Greece, 288-297.

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

Brown, P. F., J. Cocke, S.A. Della Pietra, V.J. Della Pietra, F. Jelinek, R. Mercer & P. Roossin (1988) A Statistical Approach to Language Translation. *Coling Budapest: Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, Hungary, 71-77.

Brown, P. F., J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. Lafferty, R. L. Mercer & P. S. Roossin (1990) A Statistical Approach to Machine Translation. *Computational Linguistics* **16**, 79-85.

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, P. F., J. C. Lai & R. L. Mercer (1991) Aligning Sentences in Parallel Corpora. *29th Annual Meeting of the Association for Computational Linguistics*, Berkeley, California, 169-176.

Brown, R. D. (1996) Example-Based Machine Translation in the PANGLOSS System. *COLING-96: The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 169-174.

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, 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, 22-32.

Brown, R. D. (2000) Automated Generalization of Translation Examples. *Proceedings of the 18th International Conference on Computational Linguistics: COLING 2000 in Europe*, Saarbrücken, Germany, 125-131.

Brown, R. D. & R. E. Frederking (1995) Applying Statistical Language Modelling to Symbolic Machine Translation. P*roceedings of the Sixth International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium, 354-372.

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

Carl, M. & S. Hansen (1999) Linking Translation Memories with Example-Based Machine Translation. *Machine Translation Summit VII*, Singapore, 617-624.

Chen, S. (1993) Aligning Sentences in Bilingual Corpora Using Lexical Information. *31st Annual Meeting of the Association for Computational Linguistics*, Columbus, Ohio, 9-16.

Church, K. W. (1993) Char_align: A Program for Aligning Parallel Texts at the Character Level. *31st Annual Meeting of the Association for Computational Linguistics*, Columbus, Ohio, 1-8.

Cicekli, I. (2000) Similarities and Differences. *Proceedings of SCI2000*, Orlando, FL, 331-337.

Cicekli, I. & 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, 90-97.

Cicekli, I. & H. A. Güvenir (2001) Learning Translation Templates from Bilingual Translation Examples. *Applied Intelligence* **15**, 57-76.

Collins, B. & P. Cunningham (1995) A Methodology for EBMT. *4th International Conference on the Cognitive Science of Natural Language Processing,* Dublin.

Collins, B., P. Cunningham, & T. Veale (1996) Adaptation-Guided Retrieval for Example-Based Machine Translation. Expanding MT Horizons: *Proceedings of the 2nd Conference of the Association for Machine Translation in the Americas*, Montreal, Canada, 1-13.

Collins, B. & 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, 119-126.

Cranias, L., H. Papageorgiou & S. Piperidis (1994) A Matching Technique in Example-Based Machine Translation. *COLING-94: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 100-105.

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.

Daille, B., E. Gaussier & J-M. Langé (1994) Towards Automatic Extraction of Monolingual and Bilingual Terminology. *COLING-94: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 515-521.

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 (1996) Machine Translation Method Using Inductive Learning with Genetic Algorithms. COLING-96: *The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 1020-1023.

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.

Farwell, David, Laurie Gerber and Eduard Hovy (eds.) *Machine Translation and the Information Soup, Third Conference of the Association for Machine Translation in the Americas, AMTA '98*, Langhorne, PA, Berlin: Springer.

Federici, S. & V. Pirrelli (1994) The Compilation of Large Pronunciation Lexica: The Elicitation of Letter to Sound Patterns through Analogy Based Networks. *Papers in Computational Lexicography, Complex '94*, Budapest, 59-67.

Frakes, W. B. & R. Baeza-Yates (eds.) (1992) *Information Retrieval – Data Structures & Algorithms*. New Jersey: Prentice Hall PTR.

Frederking, R. & R. D. Brown (1996) The Pangloss-Lite Machine Translation System. *Expanding MT Horizons: Proceedings of the Second Conference of the Association of Machine Translation in the Americas*, Montreal, Canada, 268-272.

Frederking, R., S. Nirenburg, D. Farwell, S. Helmreich, E. Hovy, K. Knight, S. Beale, C. Domashnev, D. Attardo, D. Grannes & R. Brown (1994) Integrating Translations from Multiple Sources within the PANGLOSS Mark III Machine Translation System. *Technology Partnerships for Crossing the Language Barrier: Proceedings of the First Conference of the Association for Machine Translation in the Americas*, Columbia, Maryland, 73-80.

Fung, P. & K. W. Church (1994) K-Vec: A New Approach for Aligning Parallel Texts. *COLING-94: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 1096-1101.

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 (1992a) An Example-Based Method for Transfer-Driven Machine Translation. *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation. Empiricist vs. Rationalist Methods in MT. TMI-92*, Montreal, Canada, 139-150.

Furuse, O. & H. Iida (1992b) Cooperation between Transfer and Analysis in an Example-Based Framework. *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, COLING-92*, Nantes, France, 645-651.

Furuse, O. & H. Iida (1994) Constituent Boundary Parsing for Example-Based Machine Translation. *COLIN-94: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 105-111.

Furuse, O. & H. Iida (1996) Incremental Translation Utilizing Constituent Boundary Patterns. *COLING-96: The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 412-417.

Gale, W. A. & K. W. Church (1991) A Program for Aligning Sentences in Bilingual Corpora. *29th Annual Meeting of the Association for Computational Linguistics*, Berkeley, California, 177-184.

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

Gaussier, E. (1998) Flow Network Models for Word Alignment and Terminology Extraction from Bilingual Corpora. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 444-450.

Green, T. R. G. (1979) The Necessity of Syntax Markers: Two Experiments with Artificial Languages. *Journal of Verbal Learning and Verbal Behavior* **18**, 481-496.

Grishman, R. (1994) Iterative Alignment of Syntactic Structures for a Bilingual Corpus. *Second Annual Workshop on Very Large Corpora (WVLC2)*, Kyoto, Japan, 57-68.

Güvenir, H. A. & A. Tunç (1996) Corpus Based Learning of Generalized Parse Tree Rules for Translation, in G. McCalla (ed.) *Advances in Artificial Intelligence*, Berlin: Springer Verlag, 121-132.

Güvenir H. A. & I. Cicekli (1996) Learning Translation Templates from Examples. *Proceedings of the 6th Annual Workshop on Information Technologies and Systems (WITS '96)*, Cleveland, Ohio, 112-121.

Güvenir, H. A. & I. Cicekli (1996) Learning Translation Templates From Bilingual Texts. *Proceedings of the 7th International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA '96)*, Sozopol, Bulgaria, 3-11.

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

Jones, D. & M. Alexa (1994) Towards Automatically Aligning German Compounds with English Word Groups in an Example-Based Translation System. *International Conference on New Methods in Language Processing*, Manchester, England, 66-71; reprinted in D. B. Jones & H. L Somers (eds.), *New Methods in Language Processing*, London 1997: UCL Press, 199-206.

Juola, P. (1994) A Psycholinguistic Approach to Corpus-Based Machine Translation. *CSNLP 1994: Third Conference on the Cognitive Science of Natural Language Processing*, Dublin, Ireland, 1-8.

Juola, P. (1997) Corpus-Based Acquisition of Transfer Functions using Psycholinguistic Principles, in D. B. Jones & H. L. Somers (eds.), *New Methods in Language Processing*, London 1997: UCL Press, 207-218.

Kaji, H. & T. Aizono (1996) Extracting Word Correspondences from Bilingual Corpora based on Word Co-Occurrence Information. *COLING-96: The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 23-38.

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.

Katz, S. (1987) Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, volume ASSP-35, 400-401.

Kay, M. (1980) The Proper Place of Men and Machines in Language Translation. Research Report CSL-80-11, Xerox PARC, Palo Alto, Calif. Reprinted in *Machine Translation* **12**, 3-23 (1997).

Kay, M. & M. Röscheisen (1993) Text-Translation Alignment. *Computational Linguistics* **19**, 121-142.

Kinoshita, S., A. Kumano & H. Hirakawa (1994) Improvement in Customizability Using Translation Templates. COLING-94: The *15th International Conference on Computational Linguistics*, Kyoto, Japan, 25-31.

Kirkpatrick, S., C. Gelatt & M. P. Vecchi (1983) Optimisation by Simulated Annealing. *Science* **20**, 671-680.

Kitamura, M. & Y. Matsumoto (1997) Automatic Extraction of Word Sequence Correspondences in Parallel Corpora. *Proceedings of the Fourth Workshop on Very Large Corpora*, Copenhagen, Denmark, 79-87.

McCreight, E. M. (1976) A Space-Economical Suffix Tree Construction Algorithm. *Journal of the Association for Computing Machinery* **23**, 262-272.

Kitano, H. & T. Higuchi (1991a) High Performance Memory-Based Translation on IXM2 Massively Parallel Associative Memory Processor. *AAAI-91 Proceedings of the Ninth National Conference on Artificial Intelligence*, Menlo Park: AAAI Press, 149-154.

Kitano, H. & T. Higuchi (1991b) Massively Parallel Memory-Based Parsing. *IJCAI-91 Proceedings of the Twelfth International Conference on Artificial Intelligence*, Sydney, Australia, 918-924.

Kitano, H. (1993) A Comprehensive and Practical Model of Memory-Based Machine Translation. *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Chambéry, France, 1276-1282.

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

Kruskal, J. B. (1983) An Overview of Sequence Comparison. In Sankoff & Kruskal (1983), 1-44.

Kukich, K. (1992) Techniques for Automatically Correcting Words in Text. *ACM Computing Surveys* **24**, 377-439.

Kupiec, J. (1993) An Algorithm for Finding Noun Phrase Correspondences in Bilingual Corpora. *31st Annual Meeting of the Association for Computational Linguistics*, Columbus, Ohio, 17-22.

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.

McTait, K. (2000) Building Blocks Translation Memory. *Communicator* **6**(9), 10-12.

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

Malavazos, C., S. Piperidis & G. Carayannis (2000) Towards Memory and Template Based Translation Synthesis. *MT 2000, Machine Translation and Multilingual Applications in the New Millennium*, Exeter, England.

Maruyama, H. & H. Watanabe (1992) Tree Cover Search Algorithm for Example-Based Translation. *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation. Empiricist vs. Rationalist Methods in MT, TMI-92*, Montreal, Canada, 173-184.

Matsumoto, Y., H. Ishimoto & T. Utsuro (1993) Structural Matching of Parallel Texts. *31st Annual Meeting of the Association for Computational Linguistics*, Columbus, Ohio, 23-30.

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

Medin, D. L. & M. M. Schaffer (1978) Context Theory of Classification Learning. *Psychological Review* **85**, 207-238.

Melamed, I. D. (1997) A Word-To-Word Model of Translation Equivalence. *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, Madrid, Spain, 490-497.

Melby, A. K. (1986) Lexical Transfer: A Missing Element in Linguistic Theories. *11th International Conference on Computational Linguistics: Proceedings of COLING-86*, Bonn, West Germany, 104-106.

Meyers, A., R. Yangarber & C. Grishman (1996) Alignment of Shared Forests for Bilingual Corpora. *COLING-96: The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 459-465.

Meyers, A., R. Yangarber, R. Grishman, C. Macleod & A. Moreno-Sandoval (1998) Deriving Transfer Rules from Dominance-Preserving Alignments. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 843-847.

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

Mitamura, T. & E. Nyburg (1992) The KANT System: Fast, Accurate, High-Quality Translation in Practical Domains. *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, COLING-92*, Nantes, France, 1069-1073.

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

Nagao, M. (1984) A Framework of a Mechanical Translation between Japanese and English by Analogy Principle, in A. Elithorn and R. Banerji (eds.), *Artificial and Human Intelligence*, Amsterdam: North-Holland, 173-180.

Nirenburg, S., S. Beale & C. Domashnev (1994) A Full-Text Experiment in Example-Based Machine Translation. *Proceedings of the International Conference on New Methods in Language Processing (NeMLaP)*, Manchester, England, 78-87.

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, TMI-93: MT in the Next Generation*, Kyoto, Japan, 47-57.

Noetzel, A. S. & S. M. Selkow (1983) An Analysis of the General Tree-Editing Problem, in Sankoff & Kruskal (1983), 237-252.

Nomiyama, H. (1992) Machine Translation by Case Generalization. *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, COLING-92*, Nantes, France, 714-720.

Och, F. J. & H. Weber (1998) Improving Statistical Natural Language Translation with Categories and Rules. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 985-989.

Oommen, B. J. & R. K. S. Loke (1997) Pattern Recognition of Strings with Substitutions, Insertions, Deletions and Generalized Transpositions. *Pattern Recognition* **30**, 789-800.

Öz, Z. & I. Cicekli (1998) Ordering Translation Templates by Assigning Confidence Factors. In Farwell et al. (1998), 51-61.

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

Poutsma, A. (1998) Data-Oriented Translation. *Computational Linguistics in the Netherlands: Ninth CLIN Meeting*, Leuven, Belgium.

Resnik, P. (1998) Parallel Strands: A Preliminary Investigation into Mining the Web for Bilingual Text. In Farwell et al. (1998), 72-82.

Resnik, P. (1999) Mining the Web for Bilingual Text. *37th Annual Meeting of the Association for Computational Linguistics*, College Park, Maryland, 527-534.

Sadler, V. (1991) The Textual Knowledge Bank: Design, Construction, Applications. *International Workshop on Fundamental Research for the Future Generation of Natural Language Processing (FGNLP)*, Kyoto, Japan, 17-32.

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

Sato, S. & M. Nagao (1990) Toward Memory-based Translation. *Papers Presented to the 13th International Conference on Computational Linguistics,* COLING-90, Helsinki, Finland, 247-252.

Sato, K. & M. Nakanishi (1998) Maximum Entropy Model Learning of the Translation Rules. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal Canada, 1171-1175.

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.

Schütze, H. (1993) Part-of-speech Induction from Scratch. *31st Annual Meeting of the Association for Computational Linguistics*, Columbus, Ohio, 251-258.

Simard, M., G. F. Foster & P. Isabelle (1992) Using Cognates to Align Sentences in Bilingual Corpora. *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation. Empiricist vs. Rationalist Methods in MT: TMI-92*, Montreal, Canada, 67-81.

Simard, M. (1998) The BAF: A Corpus of English-French Bitext. *First International Conference on Language Resources and Evaluation*, Granada, 489-494.

Simard, M. & P. Plamondon (1998) Bilingual Sentence Alignment: Balancing Robustness and Accuracy. *Machine Translation* **13**, 59-80.

Skousen, R. (1989) Analogical Modeling of Language. *Dordrecht: Kluwer*.

Smadja, F. (1993) Retrieving Collocations from Text: X-Tract. *Computational Linguistics* **19**, 143-177.

Smadja, F., K. McKeown & V. Hatzivassiloglou (1996) Translating Collocations for Bilingual Lexicons: A Statistical Approach. *Computational Linguistics* **22**, 1-38.

Sobashima, Y., O. Furuse, S. Akamine, J. Kawai & H. Iida (1994) A Bi-directional, Transfer Driven Machine Translation System for Spoken Dialogues. *COLING-94: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 64-68.

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.

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

Steiner, E. (ed.) (1991) Special Issue on Eurotra. *Machine Translation* **6**(2/3)

Sumita, E., K. Oi, O. Furuse, H. Iida, T. Higuchi, N. Takahashi & H. Kitano (1993) Example-Based Machine Translation on Massively Parallel Processors. *IJCAI-93*

Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, Chambéry, France, 1283-1288.

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

Takeda, K. (1996b) Pattern-Based Machine Translation. *COLING-96: The 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, 1155-1158.

Turcato, D. (1998) Automatically Creating Bilingual Lexicons for Machine Translation from Bilingual Text. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 1299-1305.

Utsuro, T., Y. Matsumoto & M. Nagao (1992) Lexical Knowledge Acquisition from Bilingual Corpora. *Proceedings of the fifteenth [sic] International Conference on Computational Linguistics, COLING-92*, Nantes, France, 581-587.

van der Eijk, P. (1993) Automating the Acquisition of Bilingual Terminology. *Sixth Conference of the European Chapter of the Association for Computational Linguistics*, Utrecht, The Netherlands, 113-119.

van Rijsbergen, C. J. (1979) *Information Retrieval* (2nd ed.). London: Butterworths.

van Zaanen, M. (2000) ABL: Alignment-Based Learning. *Proceedings of the Eighteenth International Conference on Computational Linguistics, COLING 2000 in Europe*, Saarbrücken, 961-967.

Vauquois, B. (1968) A Survey of Formal Grammars and Algorithms for Recognition and Transformation in Machine Translation. *IFIP Congress-68*, Edinburgh, 254-260; reprinted in C. Boitet (ed.) *Bernard Vauquois et la TAO: Vingt-cinq Ans de Traduction Automatique – Analectes*, 201-213, Grenoble (1988): Association Champollion.

Veale, T. & A. Way (1997) Gaijin: A Bootstrapping Approach to Example-Based Machine Translation. *Recent Advances in Natural Language Processing*, Tzigov Chark, Bulgaria, 239-244.

Vintsyuk, T. K. (1968) Speech Discrimination by Dynamic Programming. *Cybernetics* **4**, 52-57.

Wagner, R. A. & M. J. Fischer (1974) The String-to-String Correction Problem. *Journal of the Association for Computing Machinery* **21**, 168-173.

Wang, Y-Y & A. Waibel (1998) Modeling with Structures in Statistical Machine Translation. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 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 Fifth International Conference on Theoretical and Methodological Issues in Machine Translation TMI '93*: *MT in the Next Generation*, Kyoto, Japan, 292-301.

Watanabe, H. (1994) A Method for Distinguishing Exceptional and General Examples in Example-Based Transfer Systems. *COLING-94: The 15th International Conference on Computational Linguistics*, Kyoto, Japan, 39-44.

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

Watanabe, H. & K. Takeda (1998) A Pattern-based Machine Translation System Extended by Example-based Processing. *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Montreal, Canada, 1369-1373.

Way, A. (1999) A Hybrid Architecture for Robust MT using LFG-DOP. *Journal of Experimental and Theoretical Artificial Intelligence* **11**, 441-471.

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

Wu, D. (1994) Aligning a Parallel English-Chinese Corpus Statistically with Lexical Criteria. *32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, New Mexico, 80-87.

Wu, D. (1995) Grammarless Extraction of Phrasal Translation Examples from Parallel Texts. *Proceedings of the Sixth International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium. 354-372.

Zhang, K. & D. Shasha (1997) Tree Pattern Matching in A. Apostolico and Z. Galil (eds.), *Pattern Matching Algorithms*, New York: Oxford University Press, 341-371.

Zhao, G. & J-I. Tsujii (1999) Transfer in Experience-Guided Machine Translation. *Machine Translation Summit VII*, Singapore, 501-508.

Appendix A: Corpora

The following is a sample of the English and French WHO AFI corpus:[17]

```
<TrU>
<Txt L=AFI Budget>1998-1999
<Seg L=UKE>ASSISTANCE TO DISPLACED PEOPLE AND OTHER VULNERABLE GROUPS
IN ARMENIA, AZERBAIJAN AND GEORGIA
<Seg L=FRE>Aide aux personnes déplacées et autres groupes vulnérables
en Arménie, en Azerbaïdjan et en Géorgie
</TrU>
<TrU>
<Seg L=UKE>NATIONAL TRAINING ACTIVITIES FOR DEVELOPING COUNTRIES ON
TOXIC CHEMICALS, ENVIRONMENT AND HEALTH
<Seg L=FRE>Activités nationales de formation pour les pays en
développement sur les substances chimiques toxiques, l'environnement et
la santé
</TrU>
<TrU>
<Seg L=UKE>SPECIALIZED TRAINING IN PHARMACEUTICAL MANAGEMENT (FRENCH-
SPEAKING COUNTRIES)
<Seg L=FRE>Formation spécialisée à la gestion pharmaceutique (pays
francophones)
</TrU>
<TrU>
<Seg L=UKE>EQUATORIAL GUINEA AIDS CONTROL PROGRAMME
<Seg L=FRE>Programme de lutte contre le SIDA de la Guinée équatoriale
</TrU>
<TrU>
<Seg L=UKE>TRAINING OF TRADITIONAL BIRTH ATTENDANTS
<Seg L=FRE>Formation d'accoucheuses traditionnelles
</TrU>
<TrU>
<Seg L=UKE>MALARIA CONTROL ACTIVITIES: TRAINING
<Seg L=FRE>Activités de lutte contre le paludisme : Formation
</TrU>
<TrU>
<Seg L=UKE>SERVICES OF DR A. VAN DEN BROEK
<Seg L=FRE>Services du Dr A. Van Den Broek
</TrU>
<TrU>
<Seg L=UKE>Vaccines, sera, drugs and biological products
<Seg L=FRE>Vaccins, sérums, médicaments et produits biologiques
</TrU>
<Seg L=UKE>FOLLOW UP ON PARIS AIDS SUMMIT INITIATIVES
<Seg L=FRE>Suivi des initiatives du Sommet de Paris
</TrU>
<TrU>
<Seg L=UKE>DRUG LEGISLATION AND REGULATION DEVELOPMENT
<Seg L=FRE>Développement de la législation et de la réglementation
pharmaceutique
</TrU>
```

[17] The full form of this and further samples may be found on the accompanying CD

The following is the same sample of the English and French WHO AFI corpus tagged and lemmatised by TreeTagger:

```
<TrU>
<Seg L=UKE>assistance_NN_assistance to_TO_to displaced_VB_displace
people_NN_people and_CC_and other_JJ_other vulnerable_JJ_vulnerable
groups_NN_group in_IN_in armenia_NN_<unknown> azerbaijan_NN_<unknown>
and_CC_and georgia_NN_<unknown>
<Seg L=FRE>aide_NOM_aide à_PRE_à les_DET_le personnes_NOM_personne
déplacées_VER_déplacer et_CON_et autres_ADJ_autre groupes_NOM_groupe
vulnérables_ADJ_vulnérable en_PRE_en arménie_NOM_<unknown> en_PRE_en
azerbaïdjan_NOM_<unknown> et_CON_et en_PRE_en géorgie_NOM_<unknown>
</TrU>
<TrU>
<Seg L=UKE>national_JJ_national training_NN_training
activities_NN_activity for_IN_for developing_VB_develop
countries_NN_country on_IN_on toxic_JJ_toxic chemicals_NN_chemical
environment_NN_environment and_CC_and health_NN_health
<Seg L=FRE>activités_NOM_activité nationales_ADJ_national de_PRE_de
formation_NOM_formation pour_PRE_pour les_DET_le pays_NOM_pays
en_PRE_en développement_NOM_développement sur_PRE_sur les_DET_le
substances_NOM_substance chimiques_ADJ_chimique toxiques_NOM_toxique
le_DET_le environnement_NOM_environnement et_CON_et la_DET_le
santé_NOM_santé
</TrU>
<TrU>
<Seg L=UKE>specialized_JJ_specialized training_NN_training in_IN_in
pharmaceutical_JJ_pharmaceutical management_NN_management
french_JJ_french speaking_NN_speaking countries_NN_country
<Seg L=FRE>formation_NOM_formation spécialisée_VER_spécialiser à_PRE_à
la_DET_le gestion_NOM_gestion pharmaceutique_ADJ_pharmaceutique
pays_NOM_pays francophones_ADJ_francophone
</TrU>
<TrU>
<Seg L=UKE>equatorial_JJ_<unknown> guinea_NN_guinea aids_NN_aid
control_VB_control programme_JJ_<unknown>
<Seg L=FRE>programme_NOM_programme de_PRE_de lutte_NOM_lutte
contre_PRE_contre le_DET_le sida_NOM_sida de_PRE_de la_DET_le
guinée_NOM_guinée équatoriale_ADJ_équatorial
</TrU>
<TrU>
<Seg L=UKE>training_NN_training of_IN_of traditional_JJ_traditional
birth_NN_birth attendants_NN_attendant
<Seg L=FRE>formation_NOM_formation de_PRE_de
accoucheuses_NOM_accoucheur traditionnelles_ADJ_traditionnel
</TrU>
<TrU>
<Seg L=UKE>malaria_NN_malaria control_NN_control activities_NN_activity
training_NN_training
<Seg L=FRE>activités_NOM_activité de_PRE_de lutte_NOM_lutte
contre_PRE_contre le_DET_le paludisme_NOM_paludisme
formation_NOM_formation
</TrU>
<TrU>
<Seg L=UKE>services_NN_service of_IN_of dr_NN_<unknown> a_DT_a
van_NN_van den_NN_den broek_NN_<unknown>
```

<Seg L=FRE>services_NOM_service de_PRE_de le_DET_le dr_NOM_<unknown>
a_VER_avoir van_NOM_van den_NOM_<unknown> broek_ADJ_<unknown>
</TrU>
<TrU>
<Seg L=UKE>vaccines_NN_vaccine sera_NN_<unknown> drugs_NN_drug
and_CC_and biological_JJ_biological products_NN_product
<Seg L=FRE>vaccins_NOM_vaccin sérums_NOM_sérum
médicaments_NOM_médicament et_CON_et produits_NOM_produit
biologiques_ADJ_biologique
</TrU>
<TrU>
<Seg L=UKE>follow_VB_follow up_RP_up on_IN_on paris_NN_<unknown>
aids_VB_aid summit_NN_summit initiatives_NN_initiative
<Seg L=FRE>suivi_VER_suivre de_PRE_de les_DET_le
initiatives_NOM_initiative de_PRE_de le_DET_le sommet_NOM_sommet
de_PRE_de paris_NOM_pari
</TrU>
<TrU>
<Seg L=UKE>drug_NN_drug legislation_NN_legislation and_CC_and
regulation_NN_regulation development_NN_development
<Seg L=FRE>développement_NOM_développement de_PRE_de la_DET_le
législation_NOM_législation et_CON_et de_PRE_de la_DET_le
réglementation_NOM_réglementation pharmaceutique_ADJ_pharmaceutique
</TrU>

The following is a sample of the English and Spanish WHO AFI corpus:

```
<TrU>
<Seg L=UKE>VACCINES AND IMMUNIZATION, INCLUDING POLIOMYELITIS
ERADICATION
<Seg L=SPA>Vacunas e inmunización, incluida la erradicación de la
poliomielitis
</TrU>
<TrU>
<Seg L=UKE>United Nations joint services
<Seg L=SPA>Servicios comunes de las Naciones Unidas
</TrU>
<TrU>
<Seg L=UKE>Pakistan
<Seg L=SPA>Pakistán
</TrU>
<TrU>
<Seg L=UKE>GLOBAL SCHOOL HEALTH INITIATIVE: DEVELOPMENT OF GUIDELINES,
DOCUMENTS AND MATERIALS
<Seg L=SPA>Iniciativa Mundial de Salud Escolar: desarrollo de
directrices, documentos y material
</TrU>
<TrU>
<Seg L=UKE>OFFICE OF PROGRAMME ADMINISTRATION: GENERAL SUPPORT
<Seg L=SPA>Oficina de administración del programa: apoyo general
</TrU>
<TrU>
<Seg L=UKE>WORLD FOOD PROGRAMME
<Seg L=SPA>Programa Mundial de Alimentos
</TrU>
<TrU>
<Seg L=UKE>CLINICAL THERAPY
<Seg L=SPA>Terapia clínica
</TrU>
<TrU>
<Seg L=UKE>DIARRHOEAL DISEASES INCLUDING CHOLERA
<Seg L=SPA>Enfermedades diarreicas, incluido el cólera
</TrU>
<TrU>
<Seg L=UKE>EMERGENCY HEALTH PROGRAMME IN FORMER YUGOSLAVIA
<Seg L=SPA>Programa sanitario de emergencia en la antigua Yugoslavia
</TrU>
<TrU>
<Seg L=UKE>ASSISTANCE TO HOSPITALS IN BOSNIA
<Seg L=SPA>Asistencia a los hospitales en Bosnia
</TrU>
<TrU>
<Seg L=UKE>POISON INFORMATION SOFTWARE, EIGHTH WORKING GROUP MEETING
(BERLIN, GERMANY, 16 - 20 OCTOBER 1995)
<Seg L=SPA>Programas informáticos de información sobre intoxicaciones,
octava reunión del grupo de trabajo (Berlín, Alemania, 16 - 20 de
octubre de 1995)
</TrU>
<TrU>
```

The following is a sample of the Xerox (ScanWorX User Manual) corpus:

```
<TrU>
<Seg L=UKE>ScanWorX User's Guide
<Seg L=FRE>Guide de l'utilisateur ScanWorX pour OPEN LOOK
</TrU>
<TrU>
<Seg L=UKE>Preface
<Seg L=FRE>Préface
</TrU>
<TrU>
<Seg L=UKE>Welcome to Xerox Imaging Systems ScanWorX, the most powerful
text and graphics scanning system available for Sun Microsystems
workstations.
<Seg L=FRE>Bienvenue à ScanWorX de Xerox Imaging Systems, le système le
plus puissant de lecture de texte et de graphiques disponible pour les
postes de travail Sun Microsystems.
</TrU>
<TrU>
<Seg L=UKE>ScanWorX incorporates Xerox Imaging Systems exclusive
Intelligent Character Recognition (ICR) software, which uses artificial
intelligence techniques to achieve unparalleled character recognition
accuracy.
<Seg L=FRE>ScanWorX comporte le logiciel exclusif de Xerox Imaging
Systems de Reconnaissance intelligente de caractères (ICR), qui fait
appel aux techniques d'intelligence artificielle pour assurer une
précision inégalée de reconnaissance de caractères.
</TrU>
<TrU>
<Seg L=UKE>With ScanWorX and your Sun workstation, you can quickly and
accurately scan text from printed documents and create formatted text
files to use in desktop publishing and other text applications.
<Seg L=FRE>Avec ScanWorX et votre poste de travail Sun, vous pouvez
rapidement et avec précision lire le texte de documents imprimés et
créer des fichiers de texte formatés devant vous servir pour la
publication assistée par ordinateur et d'autres applications de
traitement de texte.
</TrU>
<TrU>
<Seg L=UKE>In addition, you can scan photographs and other
illustrations into image files in any of a number of standard image
formats.
<Seg L=FRE>Vous pouvez, de plus, lire des photographies et autres
illustrations dans des fichiers d'image et dans un certain nombre de
formats d'image standards.
</TrU>
<TrU>
<Seg L=UKE>Before going on to find out more about ScanWorX, please read
this preface, as it describes these important items:
<Seg L=FRE>Avant de vous lancer à la découverte de ScanWorX, veuillez
lire cette préface car elle donne les informations importantes
suivantes:
</TrU>
```

Appendix B: Translations

The following is a sample of the test set of 1,000 unseen SL input sentences, their translations and scores as carried out in the experiment in section 3.4:

```
CARIBBEAN EPIDEMIOLOGY CENTRE CAREC
      centre d'épidémiologie des caraïbes carec 100
PREVENTION AND TREATMENT OF NEUROLOGICAL DISORDERS
      prévention et traitement de la troubles neurologiques 71
COMMUNICABLE DISEASE CONTROL CAMBODIA MEDICAL OFFICER
      lutte contre les maladies transmissibles lutte contre les maladies
      diarrhéiques cambodge médecin 58
DEVELOPMENT OF THE NATIONAL HIV AIDS STRATEGY
      développement des populations de programmes nationaux de le vih
      sida dans la stratégie  15
HUMANITARIAN ASSISTANCE TO FORMER YUGOSLAVIA STRENGTHENING OF WHO FIELD
CAPACITY GENERAL SUPPORT
      humanitaire dans l'aide ex yougoslavie renforcement de l'oms sur
      les terrain appui général           50
MENINGITIS EPIDEMIC IN AFRICA
      en afrique de méningite en afrique  66
VACCINES AND IMMUNIZATION INCLUDING POLIOMYELITIS ERADICATION
      des  vaccins  et  vaccinations  y  compris  éradication  de  la
      poliomyélite70
NATIONAL ESSENTIAL DRUGS PROGRAMME
      programme national pour les médicaments essentials    66
WORLD CONFERENCE ON WOMEN BEIJING 4 15 SEPTEMBER 1995
      conférence  mondiale  sur  le  femmes  beijing  4  15  septembre  1995
      90
EMERGENCY ASSISTANCE PLAGUE
      aide d'urgence de peste 75
BURKINA FASO AIDS CONTROL PROGRAMME
      programme de lutte contre le sida du burkina faso     100
GLOBAL ERADICATION PROGRAMME ON DRACUNCULIASIS IN GHANA
      mondial  des  programme  d'éradication  du  sur  le  la  dracunculose
      réunions au Ghana 33
PLANNING AND MANAGEMENT VACCINE RESEARCH AND DEVELOPMENT
      planification et gestion recherche et développement en matière de
      vaccines    80
ACUTE RESPIRATORY INFECTIONS IN CHILDREN
      les infections respiratoires aiguës dans des enfants  42
SUPPORT TO CINDI NETWORK
      appui au réseau cindi   100
POLIOMYELITIS ERADICATION IN MYANMAR CONSULTANT
      éradication de la poliomyélite au myanmar consultant  100
MALARIA CONTROL PROGRAMME IN THE TIGRAY REGION
      lutte antipaludique programme des tigray région de la 33
MANAGEMENT AND GENERAL ADMINISTRATIVE SUPPORT
      gestion  de  le  général  administratives  comprises  règlement  des
      engagements d'appui aux 0
MALDIVES AIDS CONTROL PROGRAMME
      programme de lutte contre le sida des Maldives  100
FAMILY PLANNING PROGRAMME IN BULGARIA
      planification familiale programme de lutte contre le dans bulgarie
      33
REPUBLIC OF KOREA AIDS CONTROL PROGRAMME
      programme  de  lutte  contre  le  sida  de  la  république  des  de  corée
      91
THAILAND AIDS CONTROL PROGRAMME
      programme de lutte contre le sida de la thaïlande     100
```

Appendix C: Morphological Analysis

The following is a sample of the two-level spelling rules & finite-state tables for French:

```
; -----------------------
; mang+ons
; mangeons
; -----------------------
;
RULE " +:e <=> g ___ [o|a]" 4 6

      +  +  g  o  a   @
      e  @  g  o  a   @
 1:   0  1  2  1  1   1
 2:   4  3  2  1  1   1
 3:   0  1  2  0  0   1
 4.   0  0  0  1  1   0


; -----------------------
; plac+ais
; plaç0ais
; -----------------------
;
RULE " c:ç <=> [VO|CO] ___ +:0 [o|a]" 6 8

      c  c  CO  +  a  o  VO   @
      ç  @  CO  0  a  o  VO   @
 1:   0  1   2  1  2  2   2   1
 2:   5  3   2  1  2  2   2   1
 3:   0  1   2  4  2  2   2   1
 4:   0  1   2  1  0  0   2   1
 5.   0  0   0  6  0  0   0   0
 6.   0  0   0  0  2  2   0   0


; ------------------------------------------------
; journal+x
; journau0x
; beware of exceptions: finals, bancals, navals
; ------------------------------------------------

RULE " l:u <=> a ___ +:0 x#" 8 7

      l  l  a  +  x  #   @
      u  @  a  0  x  #   @
 1:   0  1  2  1  1  1   1
 2:   6  3  2  1  1  1   1
 3:   0  1  2  4  1  1   1
 4:   0  1  2  1  5  1   1
 5:   0  1  2  1  1  0   1
 6.   0  0  0  7  0  0   0
 7.   0  0  0  0  8  0   0
 8.   0  0  0  0  0  1   0
```

The following is a sample of the *Englex* two-level spelling rules for English:

```
;==========
;Epenthesis
;==========

; LR: fox+s  kiss+s  church+s  spy+s
; SR: foxes  kisses  churches  spies

RULE
"+:e <=> [CNsib | y:i | o] +:@ ___ s [+:@ | #]" 7 8
     +  CNsib  +    s    #    y    o   @
     e  CNsib  @    s    #    i    o   @
 1:  0    2    1    2    1    2    7   1
 2:  3    2    5    2    1    2    7   1
 3.  0    0    0    4    0    0    0   0
 4.  0    0    1    0    1    0    0   0
 5:  0    1    1    6    1    1    1   1
 6:  0    1    0    1    0    1    1   1
 7:  3    2    1    2    1    2    7   1


;============
;y:i-spelling
;============

; LR: spy+s  happy+ly  spot+y+ness
; SR: spies  happi0ly  spotti0ness

RULE
"y:i <= @:CN ___ +:@ ~[i | ']"  4 7
      @   y   y   +   i   '   @
      CN  i   @   @   i   '   @
 1:   2   1   1   1   1   1   1
 2:   2   1   3   2   1   1   1
 3:   2   1   1   4   1   1   1
 4:   0   0   0   0   1   1   0

RULE
"y:i => @:CN ___ +:@ ~[i | ']"  4 6
      @   y   +   i   '   @
      CN  i   @   i   '   @
 1:   2   0   1   1   1   1
 2:   2   3   2   1   1   1
 3.   0   0   4   0   0   0
 4.   2   1   1   0   0   1
```

The following is a sample of the list of root forms for French morphological analysis:

```
\lf âg
\lx
\alt Root
\fea
\gl âger

\lf âge
\lx
\alt Root
\fea
\gl âge

\lf île
\lx
\alt Root
\fea
\gl île

\lf échange
\lx
\alt Root
\fea
\gl échange

\lf échang
\lx
\alt Root
\fea
\gl échanger

\lf échelle
\lx
\alt Root
\fea
\gl échelle

\lf échographie
\lx
\alt Root
\fea
\gl échographie

\lf éclairage
\lx
\alt Root
\fea
\gl éclairage

\lf écoépidémiologie
\lx
\alt Root
\fea
\gl écoépidémiologie
```

```
\lf école
\lx
\alt Root
\fea
\gl école

\lf écologique
\lx
\alt Root
\fea
\gl écologique

\lf écologiquement
\lx
\alt Root
\fea
\gl écologiquement

\lf économie
\lx
\alt Root
\fea
\gl économie

\lf économique
\lx
\alt Root
\fea
\gl économique

\lf écrir
\lx
\alt Root
\fea
\gl écrire

\lf écriv
\lx
\alt Root
\fea
\gl écrire

\lf écri
\lx
\alt Root
\fea
\gl écrire

\lf édition
\lx
\alt Root
\fea
\gl édition
```

The following is a sample of the list of root forms for English morphological analysis:

```
\lf ability
\lx
\alt Root
\fea
\gl ability

\lf abolition
\lx
\alt Root
\fea
\gl abolition

\lf abortion
\lx
\alt Root
\fea
\gl abortion

\lf abuse
\lx
\alt Root
\fea
\gl abuse

\lf academic
\lx
\alt Root
\fea
\gl academic

\lf academy
\lx
\alt Root
\fea
\gl academy

\lf accelerate
\lx
\alt Root
\fea
\gl accelerate

\lf acceleration
\lx
\alt Root
\fea
\gl acceleration

\lf acceptability
\lx
\alt Root
\fea
\gl acceptability
```

```
\lf access
\lx
\alt Root
\fea
\gl access

\lf accident
\lx
\alt Root
\fea
\gl accident

\lf accordance
\lx
\alt Root
\fea
\gl accordance

\lf account
\lx
\alt Root
\fea
\gl account

\lf acid
\lx
\alt Root
\fea
\gl acid

\lf acquire
\lx
\alt Root
\fea
\gl acquire

\lf acquisition
\lx
\alt Root
\fea
\gl acquisition

\lf act
\lx
\alt Root
\fea
\gl act

\lf action
\lx
\alt Root
\fea
\gl action
```

The following is a sample of the list of inflectional suffixes for French:

```
\lf +e
\lx
\alt Infl
\fea
\gl +e

\lf +es
\lx
\alt Infl
\fea
\gl +es

\lf +ons
\lx
\alt Infl
\fea
\gl +ons

\lf +ez
\lx
\alt Infl
\fea
\gl +ez

\lf +ent
\lx
\alt Infl
\fea
\gl +ent

\lf +erai
\lx
\alt Infl
\fea
\gl +erai

\lf +eras
\lx
\alt Infl
\fea
\gl +eras

\lf +era
\lx
\alt Infl
\fea
\gl +era

\lf +erons
\lx
\alt Infl
\fea
\gl +erons

\lf +erez
\lx
\alt Infl
\fea
\gl +erez

\lf +eront
\lx
\alt Infl
\fea
\gl +eront

\lf +ais
\lx
\alt Infl
\fea
\gl +ais

\lf +ait
\lx
\alt Infl
\fea
\gl +ait

\lf +ions
\lx
\alt Infl
\fea
\gl +ions

\lf +iez
\lx
\alt Infl
\fea
\gl +iez

\lf +aient
\lx
\alt Infl
\fea
\gl +aient
```

The following is a sample of the list of inflectional suffixes for English:

```
\lf +s
\lx
\alt Infl
\fea
\gl +s

\lf +ing
\lx
\alt Infl
\fea
\gl +ing

\lf +ed
\lx
\alt Infl
\fea
\gl +ed

\lf +'s
\lx
\alt Infl
\fea
\gl +'s

\lf +s'
\lx
\alt Infl
\fea
\gl +s'

; zero ending

\lf 0
\lx
\alt Infl
\fea
\gl
```

The following is a sample of the word list of French surface forms (spelling dictionary):

```
âpretés                 âtre                    âtres
à                       abâtardir               abaissa
abaissaient             abaissais               abaissait
abaissant               abaisse                 abaissé
abaissée                abaissées               abaissés
abaissèrent             abaissement             abaissements
abaissent               abaisser                abaissera
abaisserai              abaisseraient           abaisserais
abaisserait             abaisseras              abaisserez
abaisseriez             abaisserions            abaisserons
abaisseront             abaisses                abaissez
abaissiez               abaissions              abaissons
abandon                 abandonna               abandonnaient
abandonnais             abandonnait             abandonnant
abandonne               abandonné               abandonnée
abandonnées             abandonnés              abandonnèrent
abandonnent             abandonner              abandonnera
abandonnerai            abandonneraient         abandonnerais
abandonnerait           abandonneras            abandonnerez
abandonneriez           abandonnerions          abandonnerons
abandonneront           abandonnes              abandonnez
abandonniez             abandonnions            abandonnons
abandons                abasourdi               abasourdie
abasourdies             abasourdir              abasourdira
abasourdiraient         abasourdirait           abasourdirent
abasourdiront           abasourdis              abasourdissaient
abasourdissait          abasourdissant          abasourdissante
abasourdissantes        abasourdissants         abasourdisse
abasourdissent          abasourdit              abat
abat-jour               abats                   abattage
abattages               abattaient              abattais
abattait                abattant                abattants
abatte                  abattement              abattements
abattent                abattes                 abattez
abattiez                abattions               abattirent
abattis                 abattit                 abattoir
abattoirs               abattons                abattra
abattrai                abattraient             abattrais
abattrait               abattras                abattre
abattrez                abattriez               abattrions
abattrons               abattront               abattu
abattue                 abattues                abattus
abbaye                  abbayes                 abbé
abbés                   abcéder                 abcès
abdication              abdications             abdiqua
abdiquaient             abdiquais               abdiquait
abdiquant               abdique                 abdiqué
abdiquée                abdiquées               abdiqués
abdiquèrent             abdiquent               abdiquer
abdiquera               abdiquerai              abdiqueraient
abdiquerais             abdiquerait             abdiqueras
abdiquerez              abdiqueriez             abdiquerions
```

Appendix D: Miscellaneous Data

The following is a sample of the stop lists of closed-class words for English, French and Spanish:

of	à	de
the	au	la
to	aux	y
for	avec	en
by	ça	el
in	chez	del
and	comme	para
since	comment	sobre
on	contre	las
onto	dans	a
from	de	los
without	dedans	e
at	depuis	con
or	des	al
above	dessous	por
a	dessus	un
as	du	sin
with	elle	una
an	elles	
all	en	
but	entre	
among	et	
under	il	
upon	ils	
over	jusque	

Expansions of the French clitic expressions:

au	→	à + le
aux	→	à + les
du	→	de + le
s'il	→	si + il
s'ils	→	si + ils
jusqu'au	→	jusque + à + le
des	→	de + les

The following is a sample of the information required for morphological analysis. It has been created from the information in Appendix C, but is stored in database format to facilitate retrieval during program execution. The information for generation is stored in a similar way.

Surface	Lemma	Ending
accelerated	accelerate	+ed
accelerating	accelerate	+ing
acceleration	acceleration	[]
accident	accident	[]
accidents	accident	+s
accordance	accordance	[]
account	account	[]
acid	acid	[]
acquired	acquire	+ed
acquisition	acquisition	[]
acting	act	+ing
acting	acting	[]
action	action	[]
actions	action	+s
activities	activity	+s
activity	activity	[]
acupuncture	acupuncture	[]
addition	addition	[]
additions	addition	+s
additives	additive	+s
address	address	[]
adjustment	adjustment	[]
adolescent	adolescent	[]
adolescents	adolescent	+s
adults	adult	+s
advanced	advance	+ed
advanced	advanced	[]

Surface	Lemma	Ending
âgées	âger	+ées
âge	âger	+e
âge	âge	[]
échange	échanger	+e
échange	échange	[]
école	école	[]
écoles	école	+s
écologiquement	écologiquement	[]
écologiques	écologique	+s
économie	économie	[]
économique	économique	[]
économiques	économique	+s
écrire	écrire	[]
édition	édition	[]
éducatifs	éducatif	+s
éducation	éducation	[]
éducative	éducatif	+e
également	également	[]
égard	égard	[]
élèvent	élever	+ent
éléments	élément	+s
élaboration	élaboration	[]
élargi	élargir	+i
élargies	élargir	+ies
électriques	électrique	+s
électromagnétiques	électromagnétique	+s
élevées	élever	+ées
élevés	élever	+és
élevaient	élever	+aient
élevant	élever	+ant