To be presented at EWCBR'96

Adaptation-Guided Retrieval in EBMT: A Case-Based Approach to Machine Translation

Bróna Collins, Pádraig Cunningham.
Dept. of Computer Science, Trinity College, Dublin 2, Ireland.
email: bcollins@cs.tcd.ie

Abstract. In this paper we describe a methodological analysis of EBMT (Example-based Machine Translation) based on a CBR (Case-Based Reasoning) perspective. This analysis focuses on adaptation. We argue that, just as in CBR, the overall power of an EBMT system is its ability to adapt examples retrieved to suit the new problem translation. Here we describe a technique whereby reusability is a function of the abstract "adaptability" information stored in the cases. This information is exploited during both the adaptation and retrieval stages.

1. Introduction

Adaptation-guided retrieval is an important emerging notion in CBR research (Smyth & Keane, 1993; Leake, 1995; Smyth, 1996). In this paper we explore the relevance of this idea in the related area of example-based machine translation (EBMT). The motivating principle in adaptation guided retrieval is that cases are retrieved in CBR in order to be reused. This retrieval is normally based on the similarity of selected features. In the end this similarity is acting as a proxy for the adaptability of the retrieved case. This is not necessarily a good thing because semantic similarity may not always signal ease of adaptation. Adaptation guided retrieval (AGR) recognises this and emphasises retrieval based on adaptability. This means that the adaptability of a case should be a key consideration in case retrieval and selection.

We argue that this policy is particularly appropriate in example-based MT. Because of the *sparse data* problem in EBMT (Gale & Church, 1993; Moonjoo, 1995) an EBMT system must have a sophisticated adaptation mechanism. In addition, adaptation is complicated by translation divergences that are brittle in reuse. This means that a translation example involving a target language segment that is structurally different to the source language segment can only be adapted with great care. An AGR policy in EBMT is useful because it helps retrieve cases that can be adapted safely. In this paper we present an approach to EBMT that quantifies this concept of 'safety' and makes it a key criterion in case selection. We describe *ReVerb*, an EBMT system that implements this idea. We illustrate that ReVerb retrieves adaptable cases in the translation of software manuals from English to German.

In section 2 we provide an overview of EBMT and present our model of EBMT that emphasises the transformations in EBMT that dictate the adaptability of a translation case in a particular context. In section 3 we describe the overall architecture of ReVerb and in section 4 we describe how adaptation-guided retrieval works in ReVerb. In section 5 we illustrate how ReVerb adapts cases, followed by an evaluation in section 6.

2. EBMT

Example Based Machine Translation (EBMT) is the marriage of MT and Case Based Reasoning (CBR) techniques (see Nagao, 1984; Sumita, 1995). It involves translating the source language (SL) into the target language (TL) via *remindings* from previous translation cases. Natural language translation can be seen as a heavily memory-dependent problem, especially when the text at hand is a *sublanguage*, (a subset of language which is specific to one domain, e.g. the language of software documentation). Professional translators often get the feeling that much of their skill is wasted on translating very repetitive passages of text. The motivation for EBMT is thus clear: when faced with a novel sentence, chances are that the expert translator will have translated something very similar before, will recognise this (matching and retrieval), and will make the changes necessary (adaptation) to produce a good translation. Approaches differ with respect to the amount of linguistic processing performed on the example texts. At one end of the spectrum are string-based approaches (Nirenburg 1995, MacLean, 1993) in which examples are retrieved on the basis of string similarity only and the input is covered by combining substrings retrieved from different cases, giving preference to longer substrings. Combining substrings in this linguistically unprincipled manner inevitably leads to the problem of 'boundary friction' -a well recognised problem in EBMT literature, but the technique has the advantage that a huge data base of examples can be built easily. Other approaches incorporate shallow syntactic indexing (Juola 1994, Cranias 1995) and then use clustering techniques or simulated annealing during retrieval. At the linguistic end of the scale, language is usually represented in tree forms which typically represent the 'dependency relations' between syntactic constituents and then tree operations (e.g. deleting and replacing subtrees) are performed in order to cover the input string (Sato 1990). The surrounding context of the subtree is taken into account, along with the number of tree operations which the template must undergo, when choosing candidate translation examples.

In ReVerb, translation examples consist of aligned sentences/ clauses which are extracted from a bilingual corpus, abstracted to a dependency level of syntactic representation, and then cross-linked during the indexing stage. Linking the respective SL and TL sentence sub-parts, or, "chunks" (where a 'chunk' is a syntactic constituent consisting of one or more words, e.g. "on the screen" is an adverbial chunk) in this way results in a translation process which accounts for structural and lexical divergences that may have occurred during initial human translation of the corpus. Thus, the *ReVerb* system uses shallower linguistic processing to that of Sato and Nagao, and, instead of calculating the exact tree operations necessary to adapt the chosen case for both the source and target language at run-time, ReVerb compares flat lists of features and values and chooses the case which will unify best with the input. ReVerb's templates contain variables which indicate the chunks can be replaced, which increases the likelihood of the input and examples unifying because in these positions, only the features (syntactic functions) and not the values (actual words) have to match. In Sato's approach, the context of the original case is taken into account when inserting it into a new tree. This means that subtrees are replaced by subtrees coming from a similar syntactic and semantic environment, which is desirable. However, it does not guarantee that the resulting tree is syntactically correct in the case of the target language. Hacking a tree on the basis of subtree similarity in the source language can never be guaranteed to result in a correspondingly hacked up target language tree with its syntactic structure still intact. This is the problem which our approach seeks to address in particular.

One way of addressing this problem is by quantifying just how "dependent" chunks within a sentence are on other chunks, not only with respect to other chunks in the sentence (within-language dependency) but also with respect to its mapping to the target text (between-languages dependency). A high dependency of either kind is penalised and receives a low-*adaptability* score.

This linking information – represented as features attached to each word – is also used for building the system's lexicon as examples are added to the case base. As words are associated with the cases in which they appear, thus giving them an explicit context, the system can exploit this by using only those words in the dictionary that were previously translated in a similar (syntactic) context to the present translation problem. Once this detailed linguistic information has been stored during the generalisation phase, the system can then proceed with the typical CBR cycle of indexing the input sentence, matching it against the case-base, retrieving a best-match, and then adapting this example to suit the new problem. At run-time, the system treats case-data as it would any other data set composed of frame-structured features and values, i.e., it does not assume any linguistic knowledge.

2.1 Mappings in EBMT

In general, an EBMT system architecture will contain the following data in the example base:

SL	Source Language in original string form.
TL	Target Language in original string form.
SLA	Source Language Abstraction.
TLA	Target language Abstraction.

The SL is abstracted into the SLA by means of some standard linguistic processing, likewise the TL (see Figure. 1). The SLA and TLA representations are then linked at the sub-sentential level. The level of linguistic abstraction will determine how substrings can be linked. After a syntactic analysis of the example SL and TL, the resulting SLA and TLA parse-trees can be linked at each node using an index (see Iida, 1991). This method allows similar syntactic constituents to be linked with no consideration of their meaning. A higher level of abstraction is that of syntactic functionality or dependency; each syntactic constituent has a certain role to play in a sentence, whether this be the subject or object or verb. Dependence relations are often expressed in hierarchical trees and this level of abstraction tells us which constituents affect which others in a sentence, for example, the subject of a sentence/clause always has to agree with the main finite verb. A node's position in such a dependency tree can be used in determining the respective contexts of the SLA and TLA nodes (see Sato 1990).

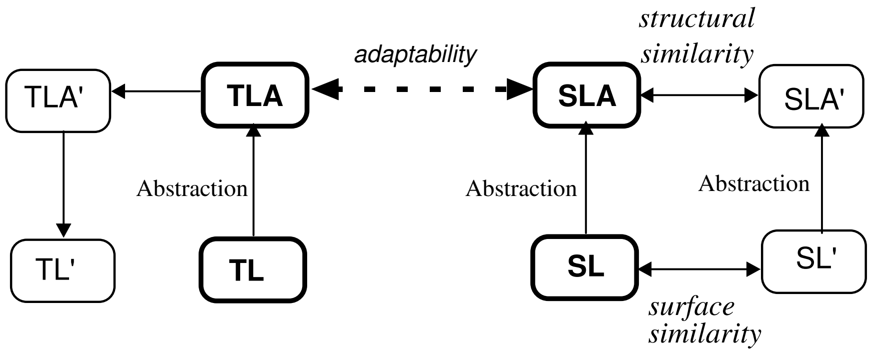

Fig.1. Multi-level correspondences between an input sentence(SL') and a given example(SL)

In order to define a subsentential mapping between two natural languages it is not sufficient to link two nodes which have the same syntactic feature. Translators often find it necessary to express SL words as different syntactic functions in the TL in order to create a grammatically or even stylistically correct TL. For example, the verb, say, in the SLA, may be realised as the subject in the TLA. Such divergences are the main obstacles to MT and there have been several attempts to formalise the problem (see Dorr, 1993; Streiter, 1995) but they inevitably suffer from the knowledge-acquisition bottleneck problem. In CBR, the idea is to compare enough previous translations so that useful generalisations fall out of the data. Therefore, one needs firstly to link two constituents on the basis of their *lexical* meaning and then to determine how the corresponding TLA constituent was realised syntactically compared to the SLA (see Appendix 1). Each case therefore is a description of a TLA derivation. A scoring mechanism can be used to penalise SLA ↔ TLA mappings where the syntactic function was changed. A case base of such derivations constitutes a subset of contrastive linguistic data, which, if suitably generalised, can be regarded as an abstract model of translation for the SLA and TLA in that particular domain. More importantly for our purposes, it represents how *adaptable* each translation case is, for only cases with good SLA ↔ TLA mappings can be adapted to compensate for any differences between the SL and the SL'.

EBMT Systems which base their retrieval mechanisms on *both* similarity and adaptability would then differentiate between the following scenarios given a case base of examples and an input sentence representation, SL'. Here, we ignore the surface similarity (of two sentences, SL' → SL) but see Section 4 for a discussion of the Base Filtering Stage.

- Good SLA' ↔ SLA Good SLA ↔ TLA –Retrievable and adaptable
- Poor SLA' ↔ SLA Good SLA ↔ TLA –Difficult to retrieve, easy to adapt.
- Good SLA' ↔ SLA Poor SLA ↔ TLA. –Easy to retrieve but adaptation is "unsafe". This means that even if only one TLA word needs changing, it will probably affect other constituents. What is more, this effect may not be determinable.
- Poor SLA' ↔ SLA Poor SLA ↔ TLA. –The case will not be retrieved.

Imitation of the retrieved example may involve nothing more than simple word substitution or it could involve integration of standard MT techniques (Kaji, 1992) (Sumita, 1991). A good way to proceed is to determine where simple chunk substitution is "safe" with respect to the above named dependencies and to perform it as much as boundary friction will allow.

3. ReVerb

3.1 Overall Architecture

The overall CBR model adopted for this work is best understood by consideration of Figure 2 following (adapted from Bergmann, 1995). While providing a coherent organisation for all the necessary components of a full EBMT system, we presently concentrate on only three components of this model, namely: *Case Organisation, Case Retrieval,* and *Case Adaptation* (described in sections 3, 4 and 5 respectively). The central philosophy of CBR employed in the design of these components is that of *Adaptation Guidance* (see Smyth & Keane 1993, Leake, 1995 Smyth 1996) –that is, these components are geared toward generating data structures (e.g., cases and chunks of cases) that are most likely to be adaptable at later stages of processing.

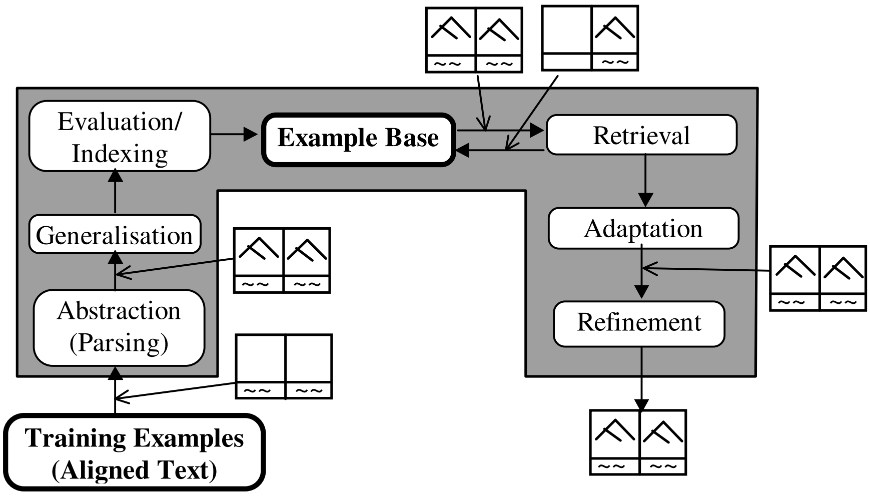

Fig. 2. A Case-Based-Reasoning Architecture for Translation

3.2. An Associative Model of Case-Memory

The current case-base is derived from a corpus of bilingual English/German CorelDRAW manuals. Each case is represented in a uniquely-labelled frame which in turn contains a number of "chunk"-frame pointers. A frame-based language KRELL is used to represent the cases. KRELL is a generic frame-management system in which specific knowledge bases may be implemented. Since a case must be retrieved from memory on the basis of its string/word content, each such word is defined in ReVerb's frame-structured memory, providing pointers to the chunks and cases in which it is employed; this situation is illustrated in Figure 3 below. The links between case frames, chunk frames, and word frames are bidirectional, with associated *demon* procedures to ensure their consistency after any updates or deletions (see Cunningham & Veale 1991).

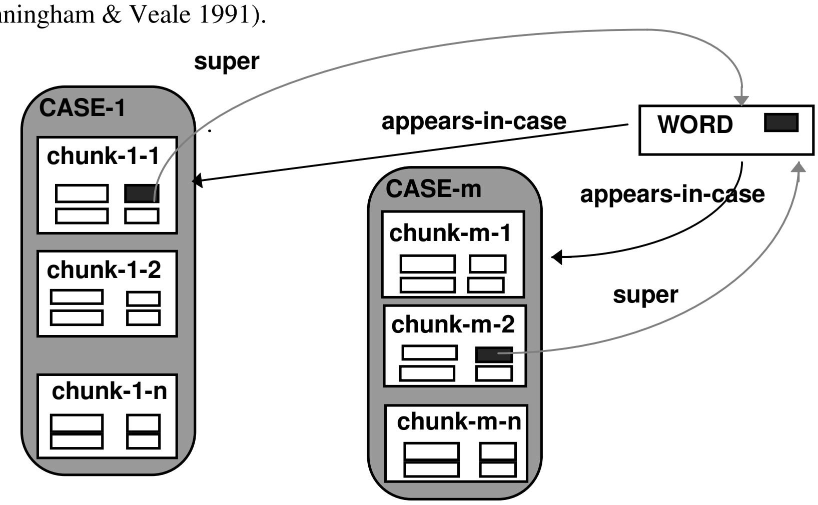

Fig. 3. Concepts linked in the ReVerb translation memory

3.3. Aligning the Examples

While we are against case-decomposition in general, due to the boundary friction problem it causes, in some situations it is perfectly safe (and advisable) to split a given SL sentence into its component clauses if there is a direct mapping into its TL counterpart, as in the following example taken from the CorelDRAW corpus:

1) <**1:** When blending on a path > <**2:** you can specify the spacing between the intermediate shapes>
<**1:** Beim Überblenden entlang einer Strecke**>** <**2:** können Sie den Abstand zwischen den Zwischenformen festlegen>
*(**gloss:** **<**During the blending along a path* **>** **<***can you the spacing between the intermediate-shapes specify* **>***)*

Such separate clauses can then be combined with the clauses arising from other sentences in the corpus without experiencing the ill effects of boundary friction, because it has already been established that the clauses in question are independent of the sentence in which they originally occurred. In our test corpus of 200 examples, about 50% of the sentences consist of two or more clauses and of those, about 10% could be split up in this manner.

3.4. Abstraction

The model of EBMT described in section 3.1 supports different levels of abstraction. At present, a surface syntactic structure is obtained for the English component of each case by passing it through the constraint based grammar *engcg* of Voutilanien (1995). This grammar tags its input with morphological and categorial information, while also annotating words with the correct syntactic function. It does not completely parse the input, as syntactic functions are not used for building a hierarchical representation of the sentence. Attachment ambiguities may thus be ignored completely. However this under-specification leaves us with a problem, namely that some fertile words have two syntactic functions. Post processing of the grammar output is currently necessary to eliminate this ambiguity of syntactic function.

3.5. Linking

ReVerb uses a *mappability scale* of 0 .. 3 to classify the different levels of correspondence that may exist between chunks (see Fig. 4.). A mappability of 3 is granted only if the correspondence is perfect: i.e., all the words in an SLA chunk are mapped in 1:1 fashion to the TLA. A mappability of 2 indicates there is a morphological difference in the words surrounding the *head* (H) of the chunk, but that the syntactic functions are the same. Morphological variation would include many-to-one mappings in either direction, inflection, case-marking, additional particles, etc.). A mappability of 1 indicates that the SLA : TLA chunks differ in syntactic function, but have a lexical correspondence which would be awarded a 3 or 2 mapping had the syntactic function been the same, while zero is reserved for chunks which exhibit no SLA:TLA commonality. Because it cannot be discerned what these 'zero' chunks are linked to in the TL, it is dangerous to make any changes to them. In general, their function in the TL is incorporated into one of the chunks of mappability 1 or 2 in such a way as to make a dictionary-based connection impossible.

```
If  the H(SL_chunk) corresponds to H(TL_chunk) for some chunk pair then
   If the syntactic-function(SL chunk)= syntactic-function(TL_chunk) then
      If the other words in  SL chunk have a 1:1 mapping to the TL chunk
               then score = 3
      else score = 2
  else score = 1
else score = 0
```

Fig. 4(a). The linking algorithm

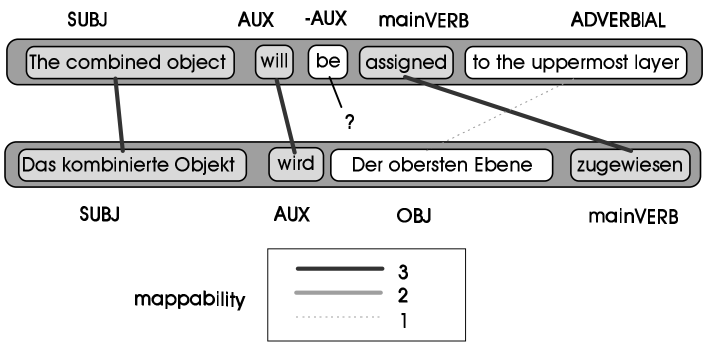

Fig. 4(b). Linking the abstracted SL and TL

3.5 Generalisation of Cases for Indexed Retrieval

Generalisation of cases (see figure 2) in ReVerb is achieved by a process of *case templatisation*, whereby a case template is generated by combining either the source text of each chunk, or a representative variable encoding the syntactic function of that chunk. Variablisation is performed by substituting SL elements of a given level of mappability/adaptability (or higher) with their *part-of-speech* (POS) tags; the intuition at work here is that the higher a chunk's mappability, the safer it will be to adapt at a later stage. As an example, consider that the SL string "*Move allows you to move the active window with the direction keys on the keyboard*" which is generalisable at four levels of variablisation, each level corresponding to a different mappability score (0 ... 3), as follows:

Level 0: SUBJ -0 FMAINV-1 OBJ -2 NFMAINV-3 OBJ-4 ADVL-5 ADVL-6
Level 1: SUBJ -0 FMAINV-1 OBJ -2 NFMAINV-3 OBJ-4 ADVL-5 ADVL-6
Level 2: SUBJ -0 *allows* OBJ -2 NFMAINV-3 OBJ-4 ADVL-5 *on the keyboard*
Level 3: *Move allows you to move the active window with the direction keys on the keyboard*

The success of this indexing scheme in syntax-based retrieval was tested on a case base of 210 cases at different levels of chunk variablisation. The results, shown in Figure 5 following, give an indication as to the trade-off between precision and recall. One can observe that as the mappability threshold for generalisation increases, the recall potential, or average number of retrievable cases, decreases. At the very safe mappability level of 3, therefore, a case-base of 20,000 cases would by our calculations contain 20 very adaptable templates.

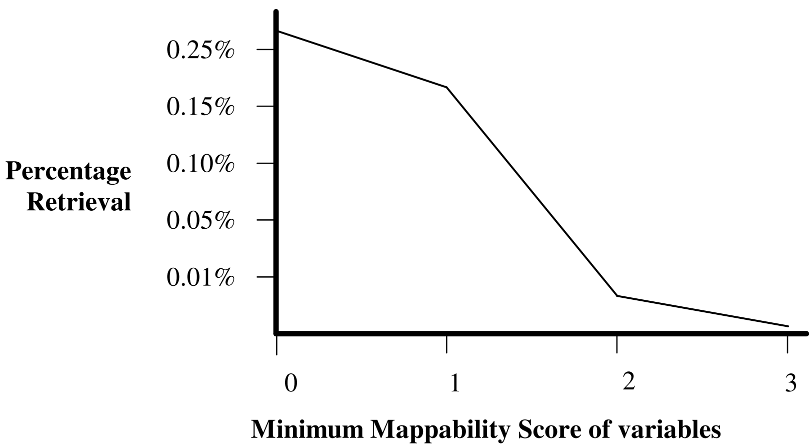

Fig. 5. This graph illustrates the probability of any two cases (from our 214 case memory) matching at different levels of variablisation. As we increase the mappability threshold for variablisation, the percentage of matches drops considerably.

4. Retrieval in ReVerb

ReVerb currently utilises two different levels of case retrieval, using both surface and structural criteria; our aim is to compare the nature of the cases retrieved at each level independently, given the same input, to determine if string-based retrieval can act as a useful filter for subsequent structural retrieval. This arrangement is illustrated in Figure 6. We shall now consider each of these levels in turn:

4.1. String Matching Retrieval : Phase 1

At run time, the user inputs a sentence, or clause, for which examples must be selected from the corpus. No linguistic judgements are made by the system regarding this input: only exact words are matched, and near morphological neighbours (e.g. "object" and "objects") are not considered. Frame *demons* will activate all cases containing exact word matches, allocating the highest scores to those cases which have been activated the greatest number of times. This brute-force technique omits all consideration of part of speech criteria, syntactic dependency, or word count of the input, but any information stored in each case (e.g., POS, head, syntactic function, linear order and mappability of the SL: TL) can be exploited if the user so wishes. The first pass is not reliable in and of itself, but is subsequently used as a loose-grain filter to select candidates for the structural matching stage, Phase 2.

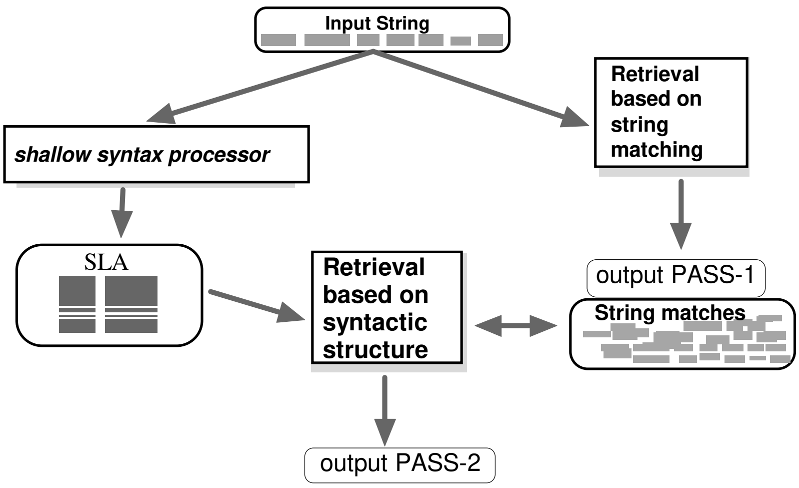

Fig. 6. Retrieval levels in ReVerb

4.2. Activation Passing for Retrieval

In the loose-grain stage, retrieval can be regarded as a process that satisfies the constraints imposed by a memory probe (i.e., the input). Marker passing is a suitable technique for retrieval at this level, which is well documented in AI literature (see Charniak 1983; Hendler 1989), and is used here to give any retrieved case a similarity score relative to the input. Activation passing is appropriate because it propagates simple numeric values throughout memory. The scoring mechanism can be viewed then as involving three phases:

Initiating Activation: The input case places activation on all nodes which contain the same words.

Summing Activation: Activation is propagated from each individual word to the chunks which contain that word.

Selection Criterion: Activation is propagated upwards from the chunk to the case which contains it such that the collection of chunk activations will produce a score of similarity for that case.

4.3. Activation Passing for Syntactic Retrieval: Phase 2

For structural retrieval, the input sentence is first pre-chunked, such that each chunk has an explicit head-word. The algorithm described below initiates activation from each word in the chunk, giving the head word an increased weighting to reflect its pivotal role in the chunk. The summation of activation proceeds as for the purely string based retrieval method, save that adaptability/mappability scores are integrated into the calculation of activation levels.

Initiating Activation: From the head word $H(C_{Si})$ of each chunk $C_{Si,}$ issue forth a wave of activation of a fixed level (1.0) say. For every non-head word of each chunk, also initiate a wave of a lower level (e.g. 0.2)

Summing Activation: When an activation wave of strength I reaches a memory chunk C_{Sj} of an example E_K<S:T> , add I to the internal activation of E_K. Then add the mappability score for C_{Sj} to produce the adaptability-based activation.

Selection Criterion: When all waves have terminated, collate in a list and sort (in descending order) the internal activation levels of each visited example E_K. return the first R examples on this ordered list.

5. Adaptation

By adaptation, we mean here on-line, or run-time adaptation of the TLA frame of a retrieved example to accommodate any differences between the input SL and the example SL'. We restrict our present discussion to the dependency-syntactic level of case representation. No changes are made to the syntactic structure (the linear ordering of syntactic functions) during adaptation. ReVerb employs the following assumption:

If the SL' and SL are very similar at a string and grammatical level, and the SLA to TLA mappability is good, then the corresponding adaptation of the TLA will result in the TL' whose meaning is that of the SL', or else the SL' is divergent.

If an SL' and an SL sentence are extremely similar on a syntactic level and yet semantically totally different, then they will almost always have different patterns of syntactic links to the target language. Awareness of this fact will prevent a translator from blindly replacing the offending word with the new word because (s)he knows that in the TL, it will not just be a matter of replacing the corresponding word there. ReVerb will always choose an example which can be adapted easily at a syntactic level. So, if the input sentence (unlike the straightforward example) actually requires a divergence, caused by a certain verb, say, then the resultant translation will not be correct and the user will have to correct it and it will be stored as a new case. This stored example will have a poor mappability but it will be specialised for that particular verb. Now, if a new input sentence also has this divergency-causing verb in it and the surrounding context is sufficiently similar to render the example retrievable, then the poor mappability of the example is less likely to result in a poor translation this time around.

5.1. Using the Corpus as A Bilingual Dictionary

If a chunk has been seen before by the system it will be stored in the system's bilingual lexicon, which is simply a table of word-frames which are linked, in turn, to their host cases and thus to their TL-equivalents within those cases. In other words, no effort goes into building the lexicon as it is a by-product of ReVerb's case-memory organisation. Chunks which have a high mappability score (3 being best) are preferred by the system, as are those which have the same syntactic function as the problem word (2+), and we are currently experimenting with incorporating a chunk's context as an added contributor to the score.

The new target word (i.e., in German) is then inserted into the new TLA structure (a copy of the old TLA example), and assumes the same syntactic function, linear order and syntactic category as the original. If the system lexicon is used, the mapping score of the substitution is used as a basis for quality determination. Depending on whether the mappability of the substitution is 3,2 or 1, the system awards a local *translation quality* of 100%, 67% and 33% respectively. The translation quality of a chunk which does not need to be substituted by lexical lookup is similarly determined on the basis of its SL': TL' mappability. In this way a *global translation quality* can be compositionally determined for the whole sentence.

If the system has not previously "seen" a particular SLA chunk to be adapted, then a bilingual dictionary outside the system has to be consulted. In this case, the word will be in its root form and so the system must consult the morphological markings on the original example's TL words to modify the dictionary entry correspondingly. The result of this process is a new case representation from which the TL string can be trivially generated without linguistic processing, by simply concatenating the chunk contents of the TLA representation in their correct linear order. In effect, this is the *strong assumption* of section 5 at work: stringing the adapted TLA chunks together according to their original linear order in this way allows the target language syntax and meaning to be preserved.

6. Evaluation

Our experiments to adapt input structures have produced encouraging results in that the output translations are quite accurate and helpful to the user. The input sentences we have used are taken from the case base itself –that is the SLA components of the cases are used to index the case base. At this stage, we are primarily interested in the correct adaptation of each struture with respect to its nearest structural matches in the case base, provided there are any. For some 74% of cases at present, however, there are no exact structural matches even when all of the chunks are variablised (level 0). This indicates how varied English language structures can be, even in the subdomain of software documentation where sentences and clauses tend to be short. Of the 24% SLAs which do retrieve structurally similar examples, the results of the matching and adaptation modules are outlined in sections 6.2 and 6.3 respectively. The base-filtering (phase 1) is discussed in section 6.1 but it is clear that it will only prove its usefulness when the case base has been scaled up. We evaluate the system by examining the respective TLA' templates retrieved at successive levels of adaptability in terms of their grammaticality and stylistics. This indicates how useful the system will be as an interactive tool for machine translation.

6.1 Phase 1 Retrieval - Sample Result

The first retrieval phase acts as a base-filter which extracts a group of candidate translations which are then input into phase two. At present, the case base is too small for the retrieved examples to be significantly similar but when the data is scaled up to, say, 20,000 examples then this stage will become more useful. Figure 7 illustrates the ordered output set of string matches for a sample input. One can immediately see that there is little similarity between these sentences and that abstraction is certainly necessary to retrieve sentences which are structurally similar to the input case.

Input: **YOU CAN ALSO WELD SINGLE OBJECTS WITH INTERSECTING LINES**

Source: YOU CAN USE THE FILL AND OUTLINE TOOLS TO CHANGE THE ATTRIBUTES OF THE ORIGINAL OBJECT
Target: MIT DEN HILFSMITTELN FUELLUNG UND UMRISS KOENNEN SIE DIE ATTRIBUTE DES ORIGINAL OBJEKTS AENDERN

Source: YOU CAN ALSO FIT OBJECTS YOU HAVE BLENDED TO A PATH
Target: UEBERBLENDETE OBJEKTE KOENNEN AUCH AN EINER STRECKE AUSGERICHTET WERDEN-

Source: YOU CAN SPECIFY THE FORMAT THE INFORMATION WILL BE IN AND OPTIONALLY CREATE A LINK TO THE SOURCE FILE
Target: SIE KOENNEN DAS FORMAT DER INFORMATION FESTLEGEN UND WENN SIE MOECHTEN EINE VERKNUEPFUNG ZUR QUELLDATEI ERSTELLEN -

Source: YOU CAN ALSO USE THIS COMMAND TO MOVE APPLICATION ICONS AND DIALOG BOXES THAT HAVE A CONTROL MENU
Target: MIT DIESEM BEFEHL KOENNEN SIE AUCH PROGRAMMSYMBOLE UND DIALOGFELDE DIE HABEN EIN SYSTEMMENUE VERSCHOBEN WERDEN

Source: YOU CAN ALSO RESTORE A WINDOW BY CLICKING ON ITS RESTORE BUTTON
Target: EIN FENSTER LAESST SICH AUCH WIEDERHERSTELLEN INDEM SIE AUF DAS FELD WIEDERHERSTELLEN KLICKEN

Fig. 7. Sample output from the base-filter from a case-base of size of 214 (Phase 1)

6.2 Phase 2: Structural Comparison and Adaptability

From our AGR point of view, this is the more interesting phase. At this level of retrieval only those example templates which are known to be adaptable will be retrieved. The threshold adaptability level with respect to the SLA' can be set to 3, 2, 1 or 0 depending on whether the user wishes to include templates with only high-level chunk mappings(3) or to include all templates(0). When the adaptability level is set to zero it is equivalent to similarity-guided retrieval only. Here we show examples of template retrieval and adaptation at each level of adaptability. At present, the system's performance at each level of adaptability is evaluated on the basis of the quality of output sentences, that is, the number of mistakes found in the output translations, as judged by a human evaluator (see figure 8). The 'translation score' is ReVerb's prediction regarding translation quality and is based on its knowledge about chunk similarity and mappability.

Mappability level of templates' variables	Number of matching pairs in 214 cases	Human evaluation (Average #mistakes per translated case)	ReVerb's own evaluation (translation score)
0	104	2.57	47.2%
1	50	1.28	55.3%
2	8	0.25	90.2%
3	6	0.00	100%

Fig. 8. Results of matching ReVerb's 214 cases against each other

6.2.1 Retrieval at adaptability threshold: 3

At this highly restrictive level, only those templates are retrieved whose SLA differences are all reconcilable by adapting a chunk of mappability 3. This usually involves a simple noun-phrase substitution. In Figure 9 below, we display the full process of adaptation for a trivial case (for clarity) at level 3.

Result: 6 matches in 214 cases
Quality: perfect translations

Example:

```
[CASE-92]  Found Match of strength 2 with [CASE-93]
Probe Case:   ENTER A PREFIX TO BE ATTACHED TO THE DIMENSION TEXT HERE
Example Source: ENTER A SUFFIX TO BE ATTACHED TO THE DIMENSION TEXT HERE
Example Target: geben sie hier ein Praefix ein das dem Dimensionstext angehaengt werden soll

Translate:
  Direct: SOLL
  Direct:  DAS
  Direct:  EIN
  Direct:  SIE
  Direct:  GEBEN
  Adapt:  A PREFIX  :was (A SUFFIX) :lookup  (OBJ)  =  (EIN PRAEFIX)  :quality  1.0
  Direct:  WERDEN
  Direct:   ANGEHAENGT
  Direct:   DEM DIMENSIONSTEXT
  Direct:  HIER
-
Translating case [CASE-93] ...
Target [GERMAN]:  geben sie hier ein Praefix ein das dem Dimensionstext angehaengt werden soll
```

Fig. 9. Adaptation at Level 3 of a highly similar Example

6.2.2 Retrieval at adaptability threshold: 2

At this level, one or more of the SLA chunks which need adaptation is different in *word-form* to the TLA, but syntactic functionality is isomorphic.

Result: 8 matches in 214 cases
Quality: Same as for threshold 3, but is dependent on the chunk's translation being correct in the lexicon for the context of the new case.

Example:

Probe Case: Rotate rotates the intermediate blend objects.
Example Source: Delete Command deletes selected objects.
Example Target: Befehl loeschen loescht markierte Objekte.
Translate: Drehung dreht die Zwischenformen.
Human evaluation: no mistakes
ReVerb Evaluation: 0.889

6.2.3 Retrieval at adaptability threshold: 1

here, the SLA chunks that need adaptation have different corresponding syntactic functions in the TLA.
Result: 50 matches in 214 cases
Quality: Varies from perfect (no mistakes) in some 36% of the cases to mediocre (3 errors or less) in most others. What is very interesting here is that almost two-thirds of the cases (62%) had only one error or less, which would be easy for even an inexperienced translator to correct. Only 2% of the cases had more than 3 errors. The most common problem is the verb positioning and the incorrect form of the verb appearing in the adapted case. Another common problem is that of prepositions vanishing because an ADVL chunk containing a preposition is suddenly replaced by a SUBJ or OBJ chunk (as is allowed at this threshold of adaptability) then the preposition will be overwritten by a single noun. The first example below has one such preposition omission and a misplaced verb ("geoeffnet") and a missing auxiliary verb "wird".

Examples:

Probe Case: Pressing ALTF2 opens the Dimension Rollup.
Example Source: Specifying negative values changes the direction of the rotation
Example Target: Durch Druecken der Tastenkombination AltF2 wird das Dimensionstext geoeffnet
Output: __betaetigen der Tastenkombination AltF2 geoeffnet das dimension rollup
Human Evaluation: 2 mistakes
ReVerb Evaluation: 0.583

Probe Case: Some or all the intermediate shapes may be drawn as open paths.
Example Source: Both formats can be imported into CorelDRAW
Example Target: Beide Formate koennen in CorelDRAW importiert werden.
Output: einige oder alle Zwischenformen moeglicherweise als ungeschlossene Strecken gezeichnet werden.
Human Evaluation: 1 mistake (need a modal verb instead of an adverbial)
ReVerb Evaluation: 0.75

6.2.4 Retrieval at adaptability threshold: 0

One or more chunks in the SLA have no lexically determinable links to the TLA.

Result: 104 matches in 214 cases
Quality: Words which needed adaptation were often not recorded in the system dictionary. The translation quality suffered from extra TLA' words being present which were irrelevant to the SLA'. Also, linear order and word-forms of many chunks

were incorrect, especially verbs. The example below shows a good translation, but this is due to the fact that the verb "use" which has a 0 mapping to the TLA, plays the same role in both SLA and SLA':

Example:
Probe Case: Use the Zoom Tool to bring all objects into view.
Example Source: Use the Offset Command to specify the spacing between the shapes.
Example Target: Mit der Option Abstand legen Sie den Abstand zwischen den Formen fest.
Output: bringen mit der Option Zoom des Hilfmittels Zoom Sie alle Objekte in den sichtbaren bereich
Human Evaluation: 1 mistake (word order)
ReVerb Evaluation: 0.583

7. Conclusions & Future Work

In testing our experimental system, *ReVerb*, we found that it can accurately adapt exemplar TLA templates when the corresponding SLA exemplar has sufficient structural similarity to the input sentence. Indeed, the CorelDRAW corpus contains several examples of such near-mappings, which are typical of computer software manuals. We have implemented the system in such a way as to be extensible in the following important respects. Firstly, we wish to bring such near-misses within the reach of the system. For instance, the input sentence may differ from a memory exemplar only in the addition (or omission) of a trailing temporal adjunct such as "yesterday". Redundancy rules governing the organisation of the case base can easily generate new well-formed cases from old by deleting or adding such constituents (if they have a 3 mapping between SLA and TLA), thus increasing the coverage of the case-base while avoiding an overcomplication of the system's matching algorithms.

Secondly, we wish to refine the on-line adaptation process so that mappings of 2 and 1 will cause specialised adaptation operators to come into effect which will refer to the POS knowledge stored in the cases, or will use a derivational analysis approach to reconciling SLA ↔ TLA differences and transmitting them to the TL. Finally, the mechanisms are already in place for semi-automatic case creation from bilingual text, so our long-term aim is to increase the number of cases until such time that the process can proceed completely automatically. This will require that all possible structures will be present in the case base, perhaps in generalised form, and also most (contextualised) words which one may expect to find in the given application domain. Our results have lead us to believe that there is an upper limit to the number of generalised templates necessary for producing perfect translations with the help of specialised adaptation specialists. On reaching this limit, a system could perform fully automatic MT. In the meantime, however, we are confident AGR can produce translation tools which assist users –who are not necessarily bilingual –in producing high quality translations of software documentation texts.

8. References

Bergmann R., Wilke W., (1995) Building and Refining Abstract Planning Cases by change of Representation Language, *Journal of AI Research,* **3,** 53-118.

Charniak, E. (1983). Passing Markers: A Theory of Contextual Influence in Language Comprehension. *Cognitive Science* **7**. 171 -190.

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

Cranias, L., Papageorgiou, H., Piperdis, S. (1995) A Matching Technique in EBMT. *The Computational Linguistics Archive,* cmp-lg/9508005.

Cunningham, P. , Veale, T. (1991). Organizational issues arising from the integration of the Concept Network & Lexicon in a Text Understanding System, *Proceedings of 12th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 986-991.

Cunningham P., Smyth B., Veale T. (1995). On the Limitations of Memory Based Reasoning" *Advances in Case-Based Reasoning*, Lecture Notes in Artificial Intelligence, J-P. Haton, M. Keane & M. Manago (eds), 75-86, Springer Verlag.

Dorr, B.J. (ed.) (1993) *Machine Translation: A View from the Lexicon.* MIT Press.

Gale, W.A., Church, K.W. (1993) A Program for Aligning Sentences in Bilingual Corpora**.** *Computational Linguistics. 19(1).* 75-103.

Hendler, J. A. (1989). Marker Passing over Micro-Features: Toward a Hybrid Symbolic/Connectionist Model, *Cognitive Science* 13(1).

Juola, P. (1994). Corpus-based Acquisition of Transfer functions using Psycholinguistic Principles, *in the proceeding of NeMLaP-1, New Methods in Language Processing,* UMIST 1994.

Kaji, H.,Kida, Y., Morimoto, Y. (1990) Learning Translation Templetes from Bilingual Text. *COLING* 1992, Nantes. 672-678.

Leake, D. (1995). Adaptive similarity assessment for case-based explanation. International Journal of Expert Systems, 8(2): 165-194.

Moonjoo, K., Young, S.H., Choi, K. (1995) Collocation Map for overcoming Data Sparseness. *EACL 1995,* Dublin. 53-59

Nagao, M. (1984). A framework for mechanical translation between Japanese and English by analogy principle. In *A. Elithorn & R. Banerji (Eds.), Artificial and Human Intelligence.* NATO Publications.

Nirenburg, S. C. (Ed.). (1995). The Pangloss Mark III machine translation system *(Tech. Rep. No. CMU-CMT-95-145)*. Pittsburgh: Carnegie Mellon University, Center for Machine Translation

Sato, S., Nagao, M. (1990) Toward Memory-based Translation. *COLING '90.*

Sumita E. , Tsutsumi, H. (1988) A translation aid system Using Flexible Text retrieval Based on Syntax Matching. TRL Research Report, IBM.

Smyth, B. (1996) *Case-Based Design***.** PhD Thesis, Trinity College Dublin.

Smyth, B, Cunningham, P. (1995) *The Utility Problem Analysed: A Case-Based Reasoning Perspective in preparation*

Smyth, B., Keane, M.T. (1993) *Retrieving Adaptable Cases.* Lecture Notes in AI 837. EWCBR-93, Kaiserslautern.

Streiter, O. (1995) Patterns of Derivation, *TMI-95,* Leuven, 1995.

Voutilainen, A. (1992) NPtool, a detector of English noun phrases. *Proceedings of Workshop on Very Large Corpora*, Ohio, 1992.
Voutilainen, A. (1995) A syntax-based part-of-speech analyser. *Proceedings of the EACL, Dublin* 1995

Appendix 1

CorelDRAW File: *c_edit*
Sentence: *14*

CASE: *16*

Page:

SOURCE: ***The combined object will be assigned to the uppermost layer***

TARGET: ***Der kombinierte objekt wird der obersten Ebene zugewiesen***

SYN_FUN	ENGLISH CHUNK	L.O	CAT	HEAD	GERMAN CHUNK	L.O	CAT	SYN_FUN	HEAD	MAP
+FMAINV										
+FMAINV										
FMAINV	*assigned*	4	v	assign	zugewiesen	4	v	-FMAINV	zuweisen	3
+FAUXV	*will*	2	vmo	will	wird	2	v	+FAUXV	werden	3
+FAUXV			d							
-FAUXV	*be*	3	vau	be						0
SUBJ	*the combined object*	1	x NP	object	Der kombinierte Object	1	NP	SUBJ	Objekt	3
SUBJ										
OBJ										
OBJ										
OBJ										
ADVL	*to the uppermost layer*	5	PP	layer	der obersten Ebene	3	NP	OBJ	Ebene	1
ADVL										
ADVL										

SENTENCE-TYPE: GERMAN omissions:

A complete chart description of a typical ReVerb case