Exploiting multi-lingual text potentialities in EBMT systems

Federica Mandreoli Riccardo Martoglia Paolo Tiberio
Università di Modena e Reggio Emilia,
Dip. di Ingegneria dell'Informazione, Modena, Italy
{tiberio.paolo, mandreoli.federica, martoglia.riccardo}@unimo.it

Abstract

Translating documents from a source to a target language is a repetitive activity. The attempt to automate such a difficult task has been a long-term scientific dream. Among the several types of approaches in Machine Translation (MT), one of the most promising paradigms is Example-Based Machine Translation (EBMT). An EBMT system translates by analogy, using past translations to translate other, similar source-language material into the target language.

In this paper we introduce EXTRA (EXample-based TRanslation Assistant), a complete EBMT system that exploits some innovative ideas in information retrieval and multilingual text management to effectively and efficiently extract useful suggestions from past translations and present them to the translator. This work has been developed as a joint work with the LOGOS group, a worldwide leader in multilingual document translation.

1. Introduction

Translation is a repetitive activity requiring a high degree of attention and the full range of human knowledge. The attempt to automate such a difficult task has been a long-term scientific dream of enormous social, political and scientific importance. Research in this field has acquired a growing interest in the past years, and, thanks also to recent technological advances, some degree of automatic translation (or Machine Translation - MT) is nowadays a reality.

Among the various translation paradigms, one of the most promising ones is Machine-Aided Human Translation (MAHT) and, in particular, Example-Based Machine Translation (EBMT). Example-based translation is essentially translation by analogy. An EBMT system is given a set of sentences in the *source language* (from which one is translating) and their corresponding translations in the *target language*, and uses those examples to translate other, similar source-language sentences into the target language. Such bilingual knowledge base is maintained in a database usually named *Translation Memory*. The basic premise of the EBMT approach is that, if a previously translated sentence occurs again, the same translation is likely to be correct. Although in many cases EBMTs do not provide a translation by themselves, they suggest *similar sentences* in the target language thus helping to ensure the consistency of style and terminology. Such suggestions should be as close as possible to the actual translation of the source sentences so that editing the translation would take less time than generating a translation from scratch.

In this paper we introduce EXTRA (Example-based TRanslation Assistant), a complete EBMT system that exploits some innovative ideas in information retrieval and multilingual text management to effectively and efficiently extract useful suggestions from the Translation Memory and present them to the user. The system is able to exploit the Translation Memory potentialities, by providing the translator with target language suggestions in the form of whole sentences and parts of them. The search mechanisms are independent from the involved languages, thus allowing to store text corpora of many occidental languages. As far as the design of the system is concerned, we opted for a solution fitting into a DBMS context, which represents a smart choice for managing the large bilingual corpora of the translation memory. We show how search mechanisms can be mapped into SQL expressions and optimized by conventional optimizer. The immediate practical benefit of our techniques is that searches in translation memory can be widely and efficiently deployed without changes to the underlying database. This work has been developed as a joint work with LOGOS group, worldwide leader in multilingual technical document translation.

The rest of the paper is organized as follows: In Section 2 we present an overview of the EXTRA architecture. Section 3 discusses some details about how the system is able to provide useful target-language suggestions to the user. In Section 4 we show the results of the conducted experiments. Finally, Section 5 presents related works on EBMT and concludes the paper.

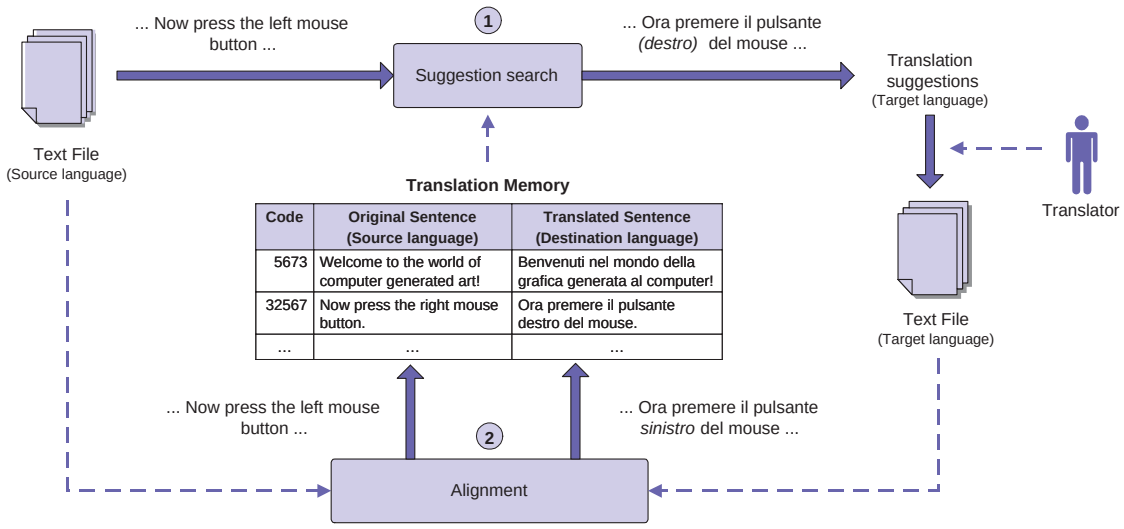


Figure 1. EXTRA architecture

2. The EXTRA Architecture

The basic conceptual architecture of EXTRA reflects, with some additions, the architecture on which standard EBMT systems rely on, and is shown in Figure 1. The Figure outlines two "paths" corresponding to as many processes: the search process (step 1) and the translation memory update process (step 2). When EXTRA is about to translate a document from a source language into a target one, it searches the translation memory (TM) for useful target-language suggestions (**Suggestion Search**). The effectiveness of the suggestion search increases with the amount of related translations it stores: As the user translates text, his/her translations become available in the database and they are particularly effective when translating a new document in the same field. The process of inserting new data in the TM is achieved by performing an automatic operation of **Alignment**. The sentences of the source language document are first aligned (i.e. combined one-to-one) with those of the target one, then the resulting pairs of sentences are stored in the translation memory together with some additional useful information.

In this paper, we will mainly focus on the system capability in extracting target-language suggestions from the translation memory multi-lingual knowledge base. In particular, in the next section we will discuss the custom search and alignment techniques we developed to this purpose.

3. Searching for useful suggestions in the target language

Given a document to be translated from a source to a target language, the extraction of useful suggestions follows two steps.

First we try to match each document sentence or part of it with the source language corpora stored in the translation memory. Due to the high complexity and extent of languages, in most cases it is rather difficult that a translation memory stores the exact translation of a given sentence. Thus, EXTRA supports *similarity searching*, i.e. searching for translation memory text which is similar or close to a given *document sentence* or *part* of it.

Second, we identify the corresponding translations in the translation memory. Whenever similar whole sentences are identified, the corresponding translation in the target language can be straightforwardly extracted since sentences in the source and target language are pairwise stored in the translation memory. Completely different is the case of similar sentence parts. Indeed, to be able to find the right correspondences between parts in the two languages we propose a *word alignment* algorithm which is able to solve the problem completely automatically and without language-specific information.

3.1. Similarity searching

A similarity search is usually based on a similarity metric which quantifies how well an object matches a query [1]. To this end, we consider a sentence as a sequence of terms and we introduce a similarity metric which is exploitable for any language since it relies on the approximate equality between sequences of terms: The parts most close to a given one are those which maintain most of the original form and contents. The (dis)similarity between sentences has to be based on a *distance function*. As far as the underlying distance function is involved, we opt for exploiting the

analogy between a sequence of terms, i.e. a sentence, and a sequence of characters, i.e. a string. In most cases, dealing with sequences of characters implies adopting the *edit distance* notion [9]. In our context, the edit distance applies to sequences of terms (*normalized sentences* in the following) obtained by applying some syntactic operations to the original sentences: we remove suffixes and stopwords and stem sentences by converting the set of the remaining terms to a common root form [1].

Definition 1 (Edit Distance between sequences) *Let σ_1 and σ_2 be two sequences, i.e. two normalized sentences. The edit distance between σ_1 and σ_2 ($ed(\sigma_1, \sigma_2)$) is the minimum number of edit operations (i.e., insertions, deletions, and substitutions) of single elements needed to transform the first sequence into the second.*

For example, let us consider S_1 ="Welcome to the world of computer art!" and S_2 ="Welcome to the world of music.". The normalized sequences are σ_1 ="welcome world compute art" and σ_2 ="welcome world music". Then, $ed(\sigma_1, \sigma_2) = 2$.

In EXTRA, edit distance is the basis of two matching mechanisms for the extraction of material similar to the source language document. Given a document sentence to be translated (*query* in the following), the system first efficiently retrieves and ranks the most similar sentences available in the translation memory and proposes them to the translator together with their target-language counterparts. They correspond to those translation memory sentences whose sequences σ are far from the query sequence σ_q less than a specified distance threshold. As to the distance threshold, the maximum number of allowed errors is defined as a user-specified percentage d of the query sequence length (i.e. $ed(\sigma_q, \sigma) < round(d * \sigma_q)$). We denote the above described search process as *approximate whole matching*. Approximate whole matching is not the only search mechanism provided by EXTRA. Experiences with several language pairs has shown that producing an EBMT system which provides reasonable translation coverage of unrestricted texts requires a great number of pre-translated text [2]. Consequently, translators may submit sentences for which no whole match exists. Anyway, the sentences stored in the translation memory could be partially useful. Thus, to further exploit the Translation Memory potentialities, EXTRA introduces new and more powerful similarity matching techniques, solving a much more complex problem we named *approximate sub^{2}sequence matching*. It attempts to match *any* parts of TM sentences against *any* query parts. Although complex, this kind of search enables the detection of similarities that could otherwise be unidentified.

As far as the design of the system is concerned, existing edit distance computation algorithms have a quadratic complexity [9]. For this reason, applying approximate whole and sub^{2}sequence matching techniques to a given query document, i.e. a set of query sentences, against a collection of TM sentences is extremely time consuming. Efficiency in retrieving the most similar parts available in the sentence repository is ensured by exploiting *filtering techniques*. Filtering is based on the fact that it may be much easier to state that two sequences do not match than to state that they match. EXTRA exploits new filters for both approximate whole and sub^{2}sequence matching, which quickly discard sequences that cannot match, efficiently ensuring no false dismissals and few false positives. Most of the filters we developed for EXTRA rely on matching short parts of length q, denoted as *q-grams* [10], of the involved sequences. Given a sequence σ, its *positional q-grams* are obtained by "sliding" a window of length q over the elements of σ.

The whole and sub^{2}sequence matching algorithms and the corresponding filtering techniques have been implemented on top of a standard DBMS by mapping them into vanilla SQL expressions. Designing a solution that fits into a DBMS context allowed us to efficiently manage the large bilingual corpora of the translation memory and ensure the full compatibility with other applications. The immediate practical benefit of our techniques is that approximate search in translation memory can be widely and efficiently deployed without changes to the underlying database. Due to the lack of space, we do not show all the involved queries. We present the one we derived from the proposal in [4] which solves the whole matching problem. While being one of the most simple, it highlights the underlying concepts and methods. Some details about sub^{2}sequence matching can be found in [5].

Let `TM` be the translation memory containing the data sentences and `Q` an auxiliary table storing the query sentences with schema `(COD, SOURCE_SENT, NORM_SENT, LEN, TARG_SENT)` where `LEN` is the length of the normalized sentence `NORM_SENT` and `TARG_SENT` obviously only applies to the translation memory. In order to enable filtering techniques based on q-grams, the database has been augmented with the q-grams of the TM and query sentences and stored in two auxiliary tables `TMq` and `Qq`, respectively. For each sentence S, its positional q-grams are represented as separate tuples in the above tables, where `POS` identifies the position of the q-gram `Qgram`. The positional q-grams of S share the same value for the attribute `COD`, which serves as the foreign key attribute to the table storing S.

The query for whole matches presented in Figure 2 shows that filters can be expressed as an SQL expression and efficiently implemented by a commercial relational query engine. It joins the auxiliary tables for q-gram sentences, `TMq` and `Qq`, with the query table `Q` and the translation memory `TM` to retrieve the sentence pairs

```
SELECT    R2.COD AS COD2, R1.COD AS COD1, R1.TARG_SENT AS SUGGESTION
          edit_distance(R1.NORM_SENT,R2.NORM_SENT,ROUND(d *R2.LEN)) AS DIST
FROM      TM R1, TMq R1q, Q R2, Qq R2q
WHERE     R1.COD = R1q.COD
AND       R2.COD = R2q.COD
AND       R1q.Qgram = R2q.Qgram
          -- position filtering
AND       ABS (R1q.POS - R2q.POS) <= ROUND(d * R2.LEN)
          -- length filtering
AND       ABS (R1.LEN - R2.LEN) <= ROUND(d * R2.LEN)
GROUP BY  R2.COD, R1.COD, R1.NORM_SENT, R2.NORM_SENT, R1.LEN, R2.LEN
          -- count filtering
HAVING    COUNT(*)>= (R1.LEN - 1 - (ROUND(d * R2.LEN) - 1) * q)
AND       COUNT(*) >= (R2.LEN - 1 - (ROUND(d * R2.LEN) - 1) * q)
AND       edit_distance(R1.NORM_SENT,R2.NORM_SENT,ROUND(d *R2.LEN))>=0
ORDER BY  COD2, DIST, COD1
```

Figure 2. Query for Whole matching. d and q are the distance threshold and q-gram size, respectively

with the corresponding suggestions in the target language. Matches are first ordered on the basis of the query sentences then, for each query sentence, we rank similar sentences on the basis of the value returned by the distance function implemented as a User Defined Function (UDF) `edit_distance(_,_,_)`.

The filtering techniques basically take the total number of q-gram matches and the position of individual q-gram match into account: *Count Filtering* requires the cardinality of the common q-gram set to be at least $max(|\sigma_1|, |\sigma_2|) - 1 - (d-1) * q$. *Position Filtering* states that a positional q-gram in one cannot correspond to a positional q-gram in the other that differs from it by more than d positions. Finally, *Length Filtering* filters sequences on the basis of their length: if two sequences are within an edit distance of d, their lengths cannot differ by more than d. Proof and explanations of the above filters can be found in [10].

3.2. Word Alignment

One of the highlights of our system is the ability to search not only for similar whole sentences, but also for similar parts. The similarity search is performed on the source language, while the translator is interested in target language suggestions. To be able to find the right correspondences between parts in the two languages we propose a word alignment algorithm which is able to solve the problem without human aid and language-specific information.

The word alignment problem can be stated in the following way: Given a source language sentence and the corresponding target language sentence, find the best mapping function between the involved words. Consider a bitextual space where words of the two sentences are placed along the two coordinates. The mapping function is then

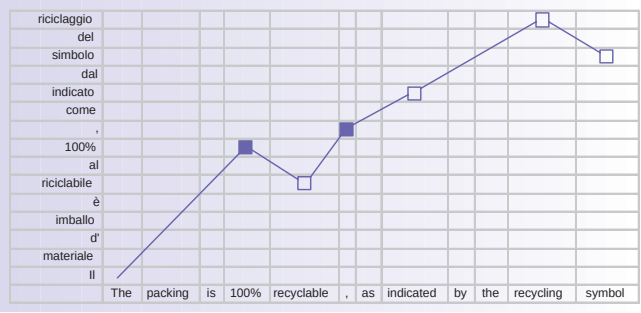


Figure 3. A word alignment example

a two-dimensional line which, in case of an "ideal" correspondence, would be a simple straight line running from the bottom-left corner to the top-right one. Since the order (and number) of the words is language dependent, the actual correspondence function between sentence words in two given languages will typically be a segmented line. Figure 3 depicts an alignment result between two example sentences. Obviously, it is not always possible to define this line in a unique and exact way. On the other hand, the goal of the word aligner is not to find the rigorous matching between each of the words, but to be able to determine, with good approximation, what target segment a given source segment corresponds to.

EXTRA word alignment follows these steps: 1) individuate sentence tokens; 2) categorize tokens (punctuation marks, numeric tokens, other words); 3) for each token, search the best points of correspondence on the basis of its category; 4) check and interpolate the points to obtain the final alignment. To find the best mapping function between tokens, the algorithm makes use of a scoring function: given

a target language token T_d, it is associated with the source language token T_s which maximizes $score(T_s, T_d)$. The scoring function is very flexible and is able to return a matching score not only for identical tokens (i.e. punctation marks, numeric tokens, proper nouns, and so on depicted as solid squares in Fig. 3.2) but also, with a lower value, to similar ones (depicted as hollow squares in Fig. 3.2). Such a similarity is based on the Longest Common Subsequence (LCS) computation between words and is often able to find a good number of further actual correspondences [6]. For example the italian word "simbolo" and the english word "symbol" are not identical, but they are very similar and their correspondence can be automatically identified by the EXTRA word aligner without relying on external language-dependant data, such as bilingual dictionaries. Furthermore, the matching score reflects the tokens positions in the sentences: the more two tokens relative distance increases, the more the matching score decreases by an appropriate decay function. For example, it is more probable that a token found at the beginning of the source sentence matches a token at the beginning of the target one rather than another near to the end. Among several fine-tuning parameters, two thresholds are defined in order to improve the quality of the results: one on the word character lengths (typically not so short words are the most significant and easily alignable), the other on the score, pruning out low scoring (and therefore loosely related) couples.

Starting from the normalized source-language parts returned by the similarity search and by combining alignment information, our system is able to identify the corresponding parts in the target-language sentence. Besides the outcome of the word alignment algorithm presented above, such an identification process relies on an additional information computed during the stemming process and telling which part of the original sentence corresponds to which part of the stemmed sentence. Figure 4 depicts such an identification process by means of an example where numbers over words correspond to the alignment and the stemming coordinates. If, for instance, the part suggestion in the stemmed source-language sentence is from word "collect" to "image" then EXTRA is able to identify the corresponding part in the target language sentence as the subsequence from "collezionare" to "immagini".

4. Experimental Evaluation

In this section we present the results of an experimental evaluation of the system performance. To effectively test the system, we used two real data sets: *Collection1*, taken from two versions of a software technical manual (consisting of 1499 reference sentences and 400 query sentences), and *Collection2*, a complete Translation Memory provided by LOGOS (34551 reference sentences and 421 query sentences).

The system performance was tested both with respect to the effectiveness and the efficiency of the proposed techniques. As far as effectiveness is concerned, it was tested in

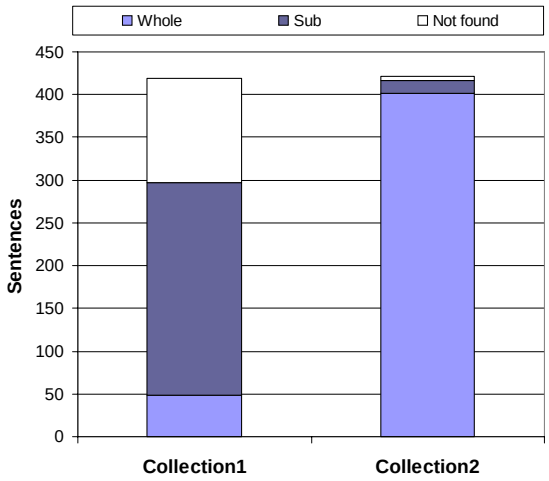


Figure 5. Coverage of the collections as outcome of the similarity search process

relation to the quality of suggestions and the notion of *coverage*. Coverage is a newly added effectiveness evaluation parameter representing the percentage of query sentences for which at least one suggestion, obtained both from whole and sub^{2}sequence matching, has been found in the translation memory. Figure 5 shows that our search techniques ensure a good coverage for the considered collections. In particular, the good size and consolidation of Collection 2 implies a very high level of coverage (over 99%), where most of the suggestions concerns whole matches. As to Collection 1, which is relatively small and not so well established, notice that, as we expected, sub^{2}sequence matching covers a remarkable percentage of query sentences and becomes essential to further exploit the Translation Memory potentialities. Figure 6 shows some examples of suggestions retrieved in Collection 2 by applying whole and sub^{2}sequence matching. The emphasized parts denote interesting parts for suggestions: notice that the suggestions in the target language are extracted by applying the alignment techniques described in 3.2.

As far as the efficiency of the similarity searching is concerned, due to the lack of space we only present an analysis of the whole matching search performance (Figure 7). We refer interested readers to [5] for sub^{2}sequence performance evaluation. The efficiency of the similarity search framework is strictly related to the effectiveness of the filters employed. We measured the size of the candidate set with respect to the cross product of the query sentences

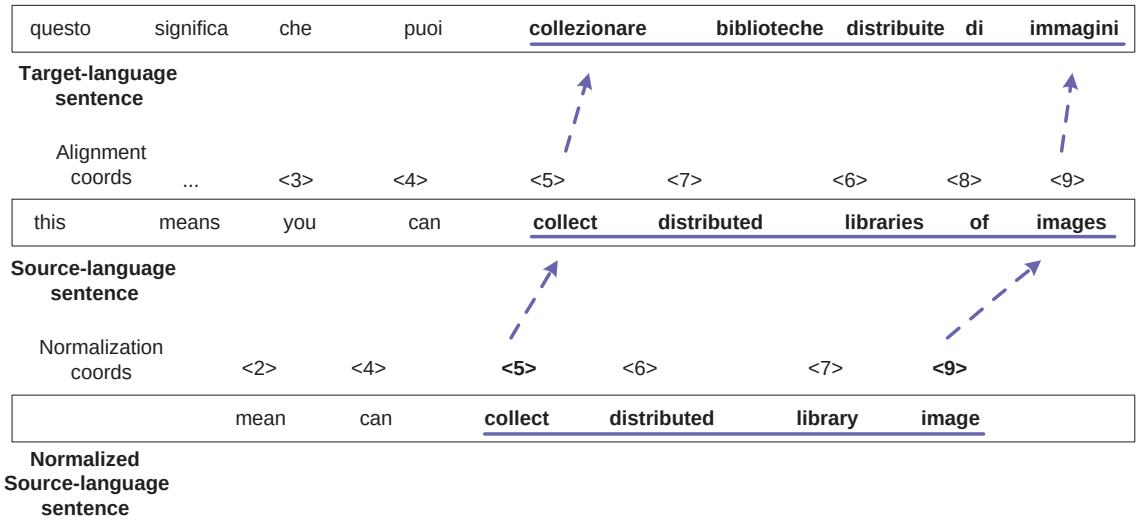


Figure 4. Using coordinates information

Query sentence: Position the 4 clips (D) as shown and at the specified dimensions.
Similar sentence in the source language: Position the 4 clips (A) as shown and at the specified distance.
Corresponding sentence in the target language: Posizionare le 4 mollette (A) come indicato e alla distanza prevista.
Query sentence: On completion of *electrical connections, fit the cooktop in place from the top* and *secure it by means of the clips* as shown.
Sentence containing a similar part: After the *electrical connection, fit the hob from the top* and hook it to the support springs, according to the illustration.
Corresponding sentence in the target language: Dopo aver eseguito il *collegamento elettrico, montare il piano cottura dall' alto* e agganciarlo alle molle di supporto come da figura.
Suggestion in the target language: collegamento elettrico, montare il piano cottura dall'alto
Sentence containing a similar part: *Secure it by means of the clips.*
Suggestion in the target language: Fissare definitivamente per mezzo dei ganci.

Figure 6. Examples of full and partial matches

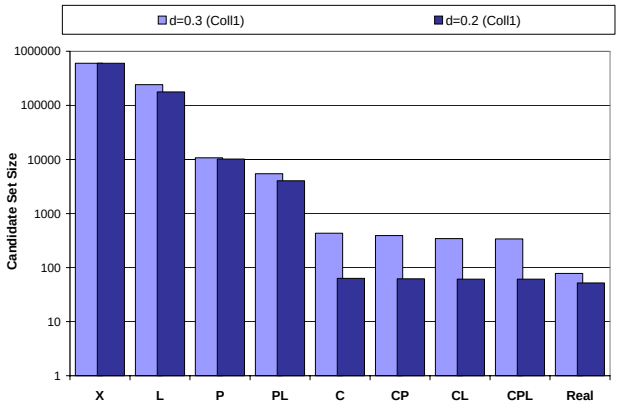
and TM sentences. Obviously, the more filters are effective the more the size of candidate answers gets near to the size of the answer set. In Figure 7-a we show the effectiveness of each combinations of filters (length, position and count) applied to Collection 1. The results for Collection 2 do not show a different behavior and therefore will not be presented. Count filter proves to be the most effective filter and is able alone to reduce the candidate set size almost to the real answer set size, particularly for smaller values of the distance threshold d. Length filter is less effective for the length distributions of our test collections (approximately inverse exponential) but its contribution is nonetheless important in reducing the overall computation time. Notice that an approximately uniform distribution of sentence lengths would significantly increase the effectiveness of the length filter since it compares the length of involved sentences and a uniform distribution increase the probability that two sentences have different lengths.

In Figure 7-b we show the results of the scalability and response time tests (executed on a Dell Optiplex NT Workstation). From our tests, the best choice is generally to turn on all the available filters: For both collections, enabling them allows the system to reduce the overall response times by a factor of at least 1:15. The scalability tests show that the times grow almost linearly with the number of sentence pairs.

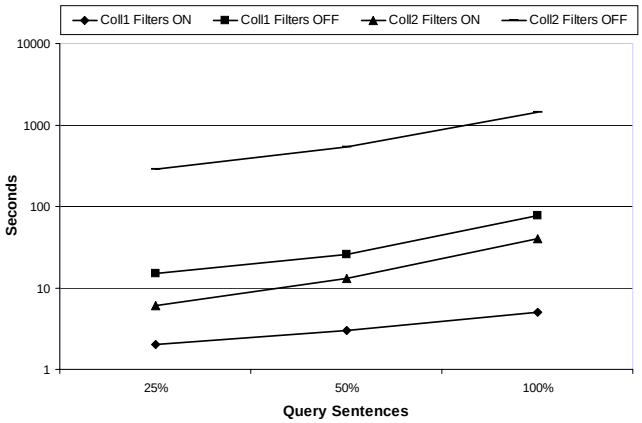
5. Related work and Conclusions

In the context of EBMT, solutions for selecting identical phrases available in the translation memory except for a similar content word have been proposed in [8, 2]. The closeness of the match is determined by a semantic distance between the two content words as measured by some metric based on a thesaurus or ontology.

The main drawback of the above approaches is twofold: they assume the availability of a particular knowledge strictly related to the involved language, thus requiring the intervention of the translator who has to annotate his/her translations before their insertion in the translation memory. Second, they never define a similarity measure thus forbidding the ranking of the results. Indeed, as far as we



(a) Filtering tests (X: Cross-product, C: Count, P: Position, L: Length)



(b) Scalability and response time tests

Figure 7. Whole matching efficiency tests

know, only few researchers tried to include the concept of approximate search in translation memories. The paper [7], for instance, introduces an example-based method relying on a very simple measurement of similarity as the degree of matching between the strings.

Commercial systems take an important role in the context of computer-aided translation. A popular set of commercial products includes Trados [11] and Déjà Vu [3]. They offer some interesting applications for document management, such as semi-automatic processes for document alignment, but they basically work in the same manner and show the drawbacks discussed above. None of such commercial systems is customizable and portable since they are based on closed architectures where information about past translations cannot be accessed from other database applications. Moreover, they do not support search of sentence parts which become essential whenever, for instance, a translator decides to merge two sentences into a single one.

In this paper, we tried to overcome some of the above cited drawbacks in the EBMT field. We presented an approach for searching useful suggestions which goes beyond the search of whole sentences while maintaining a similarity-based metric. Our experiments show the effectiveness of the underlying similarity measure and the efficiency of the SQL mapping together with filtering techniques. As a final remark, notice that the searching mechanisms are almost independent from language features, thus text corpora in any occidental language can be stored in the translation memory and exploited whenever their syntactic operations are available.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[2] R. Brown. Example-Based Machine Translation in the Pangloss Systems. In *Proc. of 16th Int'l Conf. on Computational Linguistics*, pages 169–174, 1996.

[3] Atril Deja Vu - Translation Memory and Productivity System. http://www.atril.com.

[4] L. Gravano, P. Ipeirotis, H. Jagadish, N. Koudas, S. Muthukrishnan, and D. Srivastava. Approximate String Joins in a Database (Almost) for Free. In *Proc. of 27th Int'l Conf. on Very Large DataBases (VLDB)*, 2001.

[5] F. Mandreoli, R. Martoglia, and P. Tiberio. A Syntactic Approach for Searching Similarities within Sentences. In *Proc. of the 11th Conference of Information and Knowledge Management (CIKM)*, pages 635–637, 2002.

[6] I. Melamed. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, 25(1):107–130, 1999.

[7] M. Murata, K. Uchimoto, and H. Isahara. An Example-Based Approach to Japanese-to-English Translation of Tense, Aspect, and Modality. *Journal of Japanese Society of Artificial Intelligence*, 16(1):20–28, 2001.

[8] M. Nagao. *A Framework of a Mechanical Translation between Japanese and English by Analogy Principle*. Nato Publications, 1984.

[9] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[10] E. Sutinen and J. Tarhio. Filtration with q-samples in Approximate String Matching. In *Proc. of the 7th annual Symposium on Combinatorial Pattern Matching*, 1996.

[11] Trados Team Edition - Translation Memory Technologies. http://www.trados.com.