Formalizing Translation Memories

Emmanuel Planas, Osamu Furuse

NTT Cyber Solutions Laboratories
2-4 Hikaridai Seika-cho Soraku-gun,
Kyoto 619-0237, Japan
{planas, furuse} @soy.keel.ntt.co.jp

Abstract

The TELA structure, a set of layered and linked lattices, and the notion of Similarity between TELA structures, based on the Edit Distance, are introduced in order to formalize Translation Memories (TM). We show how this approach leads to a real gain in recall and precision, and allows extending TM towards rudimentary, yet useful Example-Based Machine Translation that we call Shallow Translation.

1 Introduction

This paper describes a pioneer step towards a theory for Translation Memories (TM). We do not pretend to give "the" theoretical approach, but giving at least one arises interesting issues we propose to address now.

The use of TM involves two phases. The first consists in accumulating translation units (TU). A TU is simply a source sentence (source TU) and its corresponding translation, also called the target sentence (target TU), as the human translator has translated it. In the second phase, an input source sentence (INPUT) being given, the TM retrieves the *more similar* source TU and proposes the corresponding target TU as a close translation of the input sentence INPUT.

The first issue we want to address is how to represent sentences in the TM. Does a linear string of characters suffice? By the way, are only characters involved in the representation of sentences extracted from current editing tools like Adobe Frame Maker, Interleaf of Microsoft Word? The first section, *Structures,* proposes answers to this issue.

Another important question is: what does the *more similar* sentence mean? Is it the number of different characters between INPUT and source TU? In this case, in the example below, is sentence (0) closest to sentence (1) that has only four different letters, or to sentence (2) that has nine different letters?

(0) "The wild child is destroying his new toy."

(1) "The wild chief is destroying his new tool."

(2) "The wild children are destroying their new toy."

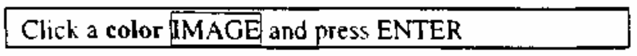
Furthermore, how to cope with non literal data like index marks and layout attributes that are inevitably found in common nowadays edited documents? The second section, *Similarity,* deals with these points.

We have built a prototype implementing our proposals for the structures and the similarity issues. Section 4 shows that this leads to a real improvement of TM efficiency. This section also introduces the notion of "Shallow Translation" as a natural extension of TM, and we give some examples of our prototype responses.

2 Structures

2.1 Real documents code

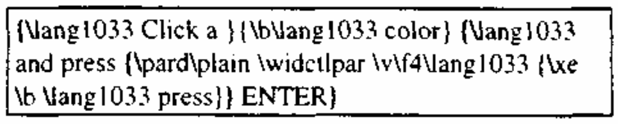
Here is the display of a typical sentence, as it can be seen in classical word editors:



```
Click a color IMAGE and press ENTER
```

Sentence (3a): as displayed word editors.

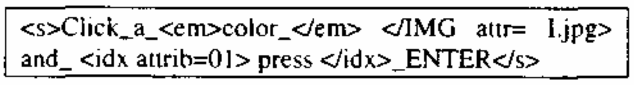
Please suppose also that there is an index mark on "color". In the typical internal code used in word processors formats (Microsoft RTF™ in the example below), it would look like this:



```
{\lang1033 Click a }{\b\lang1033 color} {\lang1033
and press {\pard\plain \widctlpar \v\f4\lang1033 {\xe
\b \lang1033 press}} ENTER}
```

Sentence (3b): as coded in word editors.

Current TM tools1 often represent internally this sentence in a SGML-like way with some variations according to each tool. Here is an XML-like representation:



```
<s>Click_a_<em>color_</em>  </IMG  attr=  I.jpg>
and_ <idx attrib=01> press </idx>_ENTER</s>
```

Sentence (3c): as coded in TM.

[1] Trades Workbench, IBM Translation Manager™, and Star Transit™ are the leaders of the TM tools market. Among today challengers there are Xerox XMS™ , Atril DejaVu™, and SDL XSDL™.

This is a necessary step for encoding in TM, in a standard way. documents whose formats are as different as Adobe Frame Maker™ and Corel WordPerfect™. Otherwise, a memory built from such a kind of formatted document could not be immediately applicable to another kind of formatted document. One has to realize that this involves filtering problems that are not so easy because the approaches to text representations word processors adopt are different, and also because some format definitions change rapidly, so already existing filters become obsolete within half the year.

2.2 Using XML as the exchange standard

We looked for such a basic level standard representation that could cope with the following conditions:

- This standard should be able to handle commercial editing file formats
- It should also be able to cope with first generation TM memory formats
- It should be able to integrate linguistic data
- It should be exchangeable on the Internet
- We did not want to re-implement a new standard

We came to the conclusion that the XML standard [XML 1997] is the basic representation we were looking for. The basic document unity is the "segment". A segment corresponds to the source or target part of the TU. It is generally a sentence, but not always ("Recommendations for the authors" for example can be such a segment). We use beginning and ending tags (we call such a set a "content tag") for delimiting groups of elements like emphasis groups, indexed words or revision marks and empty tags for isolated elements like hyperlinks buttons or images[2].

2.3 The TELA structure: Separating the data into different layers

When trying to match a new input sentence with a similar sentence from their memory, first generation TM apply their matching algorithms to a heterogeneous flow of linear data such as shown previously (Sentence (3c)). This approach leads to errors:

- A modification in the layout will have an influence on the retrieval process.
- The kind of phenomena shown in the introduction with sentences (0), (1) and (2) can not be handled.
- Transferring the non literal data to the target sentence is not possible, unless it has been previously recorded in the memory (and then only old recorded layout can be transferred).
- The application of external modules or tools to this representation is difficult: try to imagine for example how a morphological analyzer, or simply a spelling checker would behave on such an heterogeneous segment !!

If the first use of such a heterogeneous kind of representation is necessary for standardization and exchange matters, and we recommend the use of the XML standard, it does not suffice as the above points show.

We propose two improvement directions:

- The separation of the document data into a "layered" structure
- The inclusion of linguistic data in supplementary layers

Hence, rather then a flat heterogeneous structure, we propose a multilevel structure, homogeneous by level. We call the levels "layers" and the whole structure TELA[3].

This structure can have as many layers as necessary. Each layer is a lattice whose bottom is inferior to all elements of the layer, and top superior to all these elements'. The natural monotony induced by the sequence of elements in the XML segment flow of data can be kept through the partial order the lattices bear. Other orders can be specified by what we call a "linear measure scale", that is a mapping M from each element of the lattice to the Cartesian square of a numeric set. We propose eight basic layers:

1. Text Characters: this layer contains all relevant characters involved in the real text. The XML notation makes it easy to distinguish between attributes and real text. The first "path" (from bottom to top) of the lattice contains all the sequences of such characters, as found in the XML segment. These characters can be converted to a relevant coding for next processes. Each time a rewriting rule on characters is applied, the result constitutes a new path, and a "link relation" indicates the relation between old and new nodes. Imagine for example that a French extended ASCII "é" is represented by the "&eacute" string. If we use a French parser, it is more likely that the input sentence should be written in extended ASCII. Figure 1 shows how TELA can express such a rewriting operation. Note that similar operations can be applied at all layers.
2. Words: this is simply the sequence of the surface forms of the words of the sentence. Tokenizing the string of characters into separated words needs the use of an analyzer for languages for which the

[2] see [Planas 1998] for an extended description of the reasons for this choice, and [Thurmair 1997] for a overview of existing exchange formats.

[3] TELA is a French acronym for "Treillis Etagés et Liés pour le traitement Automatique", meaning "web" in Spanish, and standing for "Floored and Linked Lattices for Automatic Processing".

[4] In the following schemas, the lattices are represented horizontally: the bottom is the first node on the left, the top on the last on the right.

words are not separated in the sentence (Japanese, Chinese, Thai, etc.).

3. Lemmas (Basic forms): For a precise process of the sentences, there is a need for at least a light linguistic analysis. The lemmas, are part of the a result of this shallow parsing.
4. Parts of Speech (POS): This is also comes from the shallow parsing. The "O" part of speech qualifies terms found in the glossary (not in the parse dictionary), like "press on the ENTER key".
5. XML content tags: these tags represent for example where to apply layout attributes in the original XML segment.
6. XML empty tags: these tags cope with objects (like images) inserted in the flow of text of the XML segment. They are represented on a separate layer.
7. Glossary entries
8. Linguistic analysis structures: This level depends on how far the available linguistic analyzer can go. This layer could for example hold EBMT light structural approaches like the Constituent Boundary Analysis of [Furuse 1994], or templates like in [Takeda 1996] or [Argamon et al. 1998]. We propose the notion of "pivot schemata" as a light structure analysis. A pivot schemata is simply a pattern composed of variables and pivot keywords like "and", plus links. Such a pivot schemata is shown on Figure 3. A target pivot pattern corresponds to the source one, and is linked like on Figure 3.

Note: in Figure 2, most of the links are not represented, for the sake of the clarity of the schema.

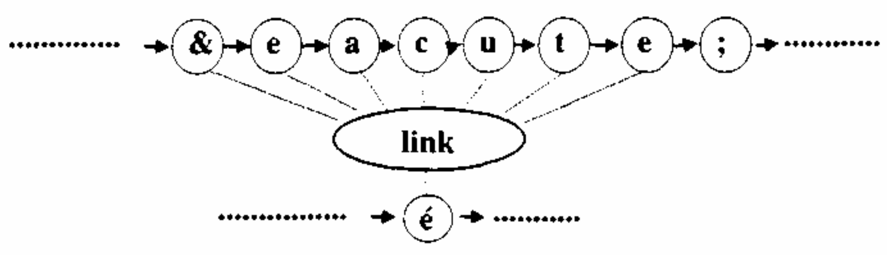


Figure 1: Creating a new node for a different coding of the French "é"character.

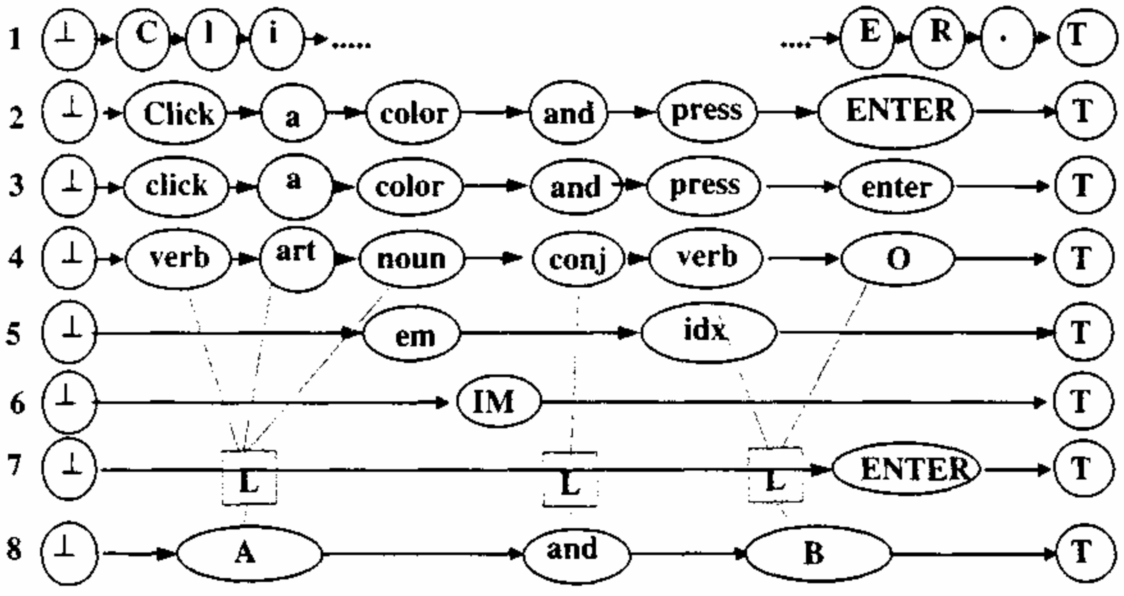


Figure 2: A TELA Structure representing sentence (3)

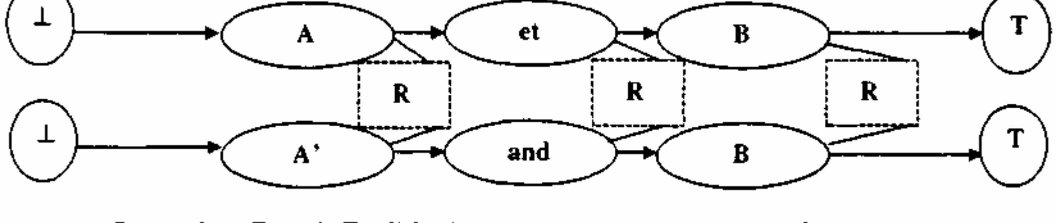


Figure 3: A French-English pivot expression with source and target parts.

3 Similarity

3.1 Using key words plus a matching algorithm

First generation TM tools generally use a key word index based engine for retrieving similar source TU from the memory. This is a fast and efficient technology, allowing to manipulate memories containing a large number of TU. This is indeed almost sufficient for the sake of getting a short list of sentences the human translator will be able to judge their reusability, but this is not optimal in terms of recall and precision, like Section 4 experiments confirm.

Furthermore, when we want to process automatically some operations like format transfer on the retrieved TU, we have no clue where to start from, because the best this kind of retrieval can offer is a general score of how many keywords have been found in the source part candidate TU. We claim that in order to perform such operations, we need to know something about the structures of the INPUT, the source TU, and its target TU.

A first option could be to perform a deep analysis of these sentences that would give us the syntactic and perhaps the logical structure of the sentence. This requires for each language to be taken into consideration a classical analyzer that takes years to be build, whose accuracy is never satisfactory, and which application would take long seconds, sometimes minutes for each sentence. We think it is not suitable for TM. We rather propose to use a shallow analysis that provides us with the lemma and the POS of every word, and for which a lot of taggers or morphological parsers are available and reasonably accurate for many languages. We propose then to look for a first short-list of similar sentences in the examples database through a index-based technique, and to refine the selection with a precise matching algorithm to be applied between the TELA representation of each sentence of the short-list including this analysis, and of the INPUT. We show in the next paragraph how, the similarity between the structure of INPUT and the source TU is interpreted as a edition operation involving the different available layers of the TELA structure.

3.2 Matching two sentences: the principle

For the sake of simplicity, we will now represent TELA structures as stacked sequences of items in an array. In our example, we will consider five layers: words, lemmas. parts of speech, content tags, (no empty tags) and glossary entries. Let us consider sentence (3) as the input sentence. (3) is represented by its related TELA structure T3. and suppose that we want to estimate its similarity to sentence (4), represented by T4. Because the TELA structure is a set of layered lattices, for each layer, there always exists a path from the bottom (left) to the top (right) node. We only take the path resulting from the last rewriting process. Having only a sequence, we reduce the comparison of two TELA structures to the comparison of a set of linear sequences of items that are presented in above Arrays 1 and 2.

We consider the matching of two sentences as a global process involving all layers. We use five layers in our example, but we could also include more layers like a semantic one for example, or less, for example when no layout or non textual objects would not be considered.

Starting from the first represented layer (layer 2: the words), and ending with the fifth (layer 7: the words from a glossary), we try to match each item of T4. the retrieved example, onto corresponding items of T3. the searched pattern. In our example, "He" does not correspond to item "click", so we try to match it at a higher level. As it does not match any item, at any level, it is then considered as a

```
<s>Click_a_<em>color_</em></IMG attr= 1.jpg>and_<idx attrib = 01> press </idx>_ENTER</s>
```

Sentence (3)

	1	2	3	4	5	6
2	~~click~~	a	color	~~and~~	~~press~~	~~enter~~
3	click	a	color	~~and~~	press	~~enter~~
4	verb	art	noun	~~conj~~	verb	O
5			em		idx	
7						~~ENTER~~

Array 1: T3, Simplified TELA structure for sentence (3)

```
<s>He_clicks_on_a_<em>color_</em></img attr=1.jpg>,_then_<idx attrib=01>presses</idx>_OK</s>
```

Sentence (4)

	1	2	3	4	5	6	7	8	9
2	~~he~~	~~clicks~~	~~on~~	a	color	:	~~then~~	~~presses~~	~~OK~~
3	~~he~~	click	~~on~~	a	color	:	~~then~~	press	~~OK~~
4	~~pp~~	verb	~~prep~~	art	noun	;	~~conj~~	verb	O
5					em			idx	
7									~~OK~~

Array 2: T4, Simplified TELA structure for sentence (4)

deletion when editing (4) into (3). Next word "clicks" does not correspond to surface form "click". So we look up a layer to see if lemma "click" matches with lemma "click". It does, so we keep this match, and try to match sentence (4) following word "on" with sentence (3) words following "a". We keep on this process for every word of (4). Finally the matching words are indicated in bold (Arrays 1 and 2). Word "and" is in (3), but not in (4), this is considered as an insertion.

Two sentences are considered as matching if and only if it exists a matching word for each of the words of the INPUT. We do not accept strict substitutions so that we are sure that the two sentences really match. A match that includes an insertion in INPUT, is useful for the human translator, but it is useless for the machine because it do not have enough linguistic knowledge for knowing where to insert the word in the target segment.

3.3 The match as an edition operation

3.3.1 Edit Distance

The above matching is in fact based on an Edit Distance. Edit Distances have been developed for matching the characters of two words, so that corrections could be performed on spelling errors (see [Wagner & Fischer 1974]). A level f and two sentences (1) and (2) being given, we consider the layers of TELA structures T1 and T2 as sequences of items $s^f_1 = (s^f_{1i})_{1<i<nf1}$, $s^f_2 = (s^f_{2j})_{1<j<nf2}$, where $n^f_1 <= n^f_2$, here n^f_1 and n^f_2 represent the number of items of the layer f of T1 and T2. The edit distance between layers s^f_1 and s^f_2 is the total cost of the sequence of elementary operations transforming s^f_1 into s^f_2 that minimizes this total cost. Here are the classical elementary operations:

- Equality (cost 0)
- Deletion (cost 1)
- Insertion (cost 1)

As an example, the distance between layers 2 of sentences T3 and T4 (say: s^2_3 and s^2_4) is 6 because, apart from equalities, one insertion ("and"), and five deletions ("he"; "on"; ","; "then", and "OK") are involved. We do not consider strict substitutions as Wagner & Fischer did because we are looking for a complete match, not a "fuzzy" match. This is essential so that we should be able to use this match for later transfer operations.

3.3.2 Similarity between two TELA structures

For matching TELA structures, we consider the following edition operations between the layers of T1 and T2:

- Layer 1 equality (score 1)
- ...
- Layer F equality (score 1)
- Deletion for all layers (score 1)
- Insertion for all layers (score 1)

Let $sum^{12}_1 ..., sum^{12}_F$; sum^{12}_- sum^{12}_+ be the sequence of the number of these elementary operations for editing S_2 into S_1. For example the edition of T4 into T3 has the following sequence; 2, 4, 5, 2, 1: 4, 1.

We have inserted a semicolon between the equality scores and the deletion and insertion scores, because they play a (geometrically) opposite paper in the comparison between S_1 and S_2 than equality scores. The similarity between S_1 and S_2 is defined as the following vector:

$$\boxed{\sigma^{12} = (sum^{12}_1/n^1_1, ..., sum^{12}_F/n^F_1,\ 1-sum^{12}_-/n^2_2,\ 1-sum^{12}_+/n^2_1)}$$

For example n^2_1 is the number of surface words in S_1, and n^5_2 is the number of content tags in S_2. Sentences (3) and (4) have then the following similarity:

σ^{34} = (2/6, 4/6, 5/6, 2/2, 1-4/9, 1-1/6), say:

σ^{34} = (2/6. 4/6, 5/6, 2/2, 5/9, 5/6)

Let S_0 be an input sentence, and S_1 and S_2 two candidates. Let σ^{01} and σ^{02} be the corresponding similarities between T_0 and T_1, and T_0 and T_2. S_0 and S_1 are said to be closest than S_0 and S_2, if $\sigma^{01} > \sigma^{02}$. The ">" notation stands for the classical partial order defined on vectors:

(2, 4, 5) > (1, 4, 5) and (2, 4, 5) > (2, 4, 4).

Considering sentences (3) and (4) and the following sentence (5) (see Array 3 below), we have:

σ^{35} = (5/6, 5/6, 6/6, 1, 6/7, 1), then $\sigma^{35} > \sigma^{34}$, and (5) is considered closest to (3) than (4).

Note that we choose to give priority to the less analyzed level, that is the surface word level. This could be redefined according to the availability of data and the linguistic analyzers, and according to the strategy too. This current choice *is not linguistically exact.* There could be for example a match for two sentences bearing some surface

<s>Click_on_a_<em>color_</em>and_<idx attrib = 01> press </idx>_OK</s>

Sentence (5)

	1	2	3	4	5	6	7
2	**click**	~~on~~	**a**	**color**	**and**	**press**	~~OK~~
3	**click**	~~on~~	**a**	**color**	**and**	**press**	~~OK~~
4	**verb**	~~prep~~	**art**	**noun**	**and**	**verb**	**O**
5				**em**		**idx**	
7							~~OK~~

Array 3: T5, Simplified TELA structure for sentence (5)

form belonging to two different base forms (like the surface form "tables" verb "to table" and noun "table"). There are however two reasons for this choice. The first is that we want our matching operation to be robust: as we do not require from the parser to know the lemma and part of speech of every word, it may happen that we do know the surface form but neither lemma nor POS. In this case, the match would be possible at the surface level layer, but not at other ones. Beginning at the surface words layer allows a response even when the analyzer can not parse the word: the recall is increased. The second reason is that we mean to use TM on similar sentences form the short-list (see 3.1), and there are good reasons to hope (and the experiments confirm it) that the sentences to be compared are close and use related vocabulary. The precision is then only slightly altered.

3.3.3 Matching algorithm

Our matching procedure gives the similarity between two multi-sequences of items. This notion of similarity is precise because it involves a *constructive* procedure based on *located* correspondences between the items of the two compared sentences, and because it does not loose information mixing-up it in a magic formula. This is a new approach, different from the classical one that relies on the retrieval of a bag of common key words, indexed on their characters and gives a *global* similarity called "fuzzy matching". Our approach is near from symbolic matching in that sense that the operands of the basic edition operations are any kind of item that match or not in a Boolean manner.

The advantage of this procedure is that not only it provides us with the similarity between the two chains, but it also gives the matching words, in a "trace". This concept has been introduced by [Wagner & Fisher 1974]. The matching trace for sentences (3) and (4). is:

(1 0 -) (2 1 3)(3 0 -)(4 2 2)(5 3 2)(0 4 +)(6 0 -)(7 0 -) (8 5 3)(9 6 4)

This indicates that for example word #4 of sentence (3) has to be inserted (0 4 +), word #7 of sentence (4) has to be deleted (7 0 -), and word #8 of sentence (4) matches with word #5 of sentence (3), on the 3rd layer (8 5 3).

We have constructed a specific matching Dynamic Programming algorithm for this purpose that we call the "Multi-level Similar String Matching" (MSSM) algorithm. It is fast since it has a linear behavior for sentences of comparative lengths, sub-quadratic in any case. This algorithm is an adaptation of ideas that can be found in [Wagner & Fischer 1974] and [Ukkonen 1985] algorithms. A related approach can be found in [Cranias et al. 1997]. without the notion of trace. Details will be given in further publications.

3.4 Layout transfer

We have now a structure able to interrelate lemmas, surface forms, and layout containers. It "suffices" to add a dictionary to be able to transfer the layout attributes from the source to the target sentence. In our example (see Figure 4). the "B" links represent bilingual correspondence found in a dictionary. "G" links come from a correspondence found in a glossary, "L" links are inherited from the XML segment structure. The layout recipient node "em" ("emphasis") being linked to "color" on the source TELA structure, and the bilingual dictionary showing that English "color" corresponds to French "couleur", we deduce that an "em" node can be assigned to the French word "couleur" of the TELA structure for the French sentence. The same operation can be performed if the link between "ENTER" and "ENTRÉE" is found in a glossary. Note that as the English word "enter" is a verb, and French "entrée" is a noun, a dictionary would not give the correspondence between the two words. Yet. it is a classical translation in software products. Note also that the classical ambiguity problem that prevents us from such an easy transfer in machine translation does not apply here. The reason is that we do not look for the target word: we have it. We only look for a bilingual correspondence between a word of the source segment and a word of the target segment. A brute force method looking for all possible corresponding translations for the source word in a dictionary and a glossary would do the job: we would only seldom find different corresponding target words in the same sentence.

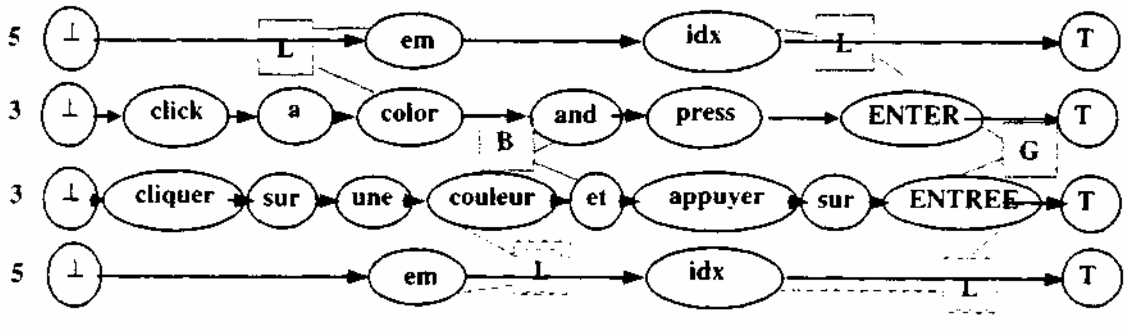


Figure 4: Transferring layout attributes

4 Experiments

We are presenting now two experiments based on the use of the TELA structure and the notion of similarity that we have described above.

4.1 Retrieving similar translations

The aim of this experiment is to compare our prototype (Vintage) retrieval performances against the leader of the TM tools market: Trados Workbench. The source language is English, the target languages are French and Japanese. The nature of the target language does not influence the results of this first experiment since all operations are conducted on the source side. The source language analyzer we used to generate the lemmas and the parts of speech is the Apple Pie parser, freely downloadable from the New York city University [Apple Pie parser]. We used two corpora. The first one is a list of strings used in an online help software (English-French). It has 7192 sentences that has been stored in the memory, and we used the first 50 sentences as a test set. The second corpus we used is a collection of brief online economical news. Originally, it has 31.526 source Japanese sentences totaling 447.772 words, aligned with a similar number of English sentences. We have used the English sentences as the source, and the Japanese as the target. We took the first 75 sentences of this second corpus as a test set. For both corpora, we looked for similar sentences in the original corpus. We collected only non identical sentences. The number of total sentences that where proposed for both tools is illustrated on the first graphic of each of Figures 5 and 6. The similarity threshold stands for the desired percentage of matching items (words at different levels) in the sentence. For example, in the first schemata of Figure 6, if we want 90% of the words to match between the retrieved examples and the input sentences, we get 20 proposals for the all of the 75 input sentences, with Vintage, and 1 with Workbench.

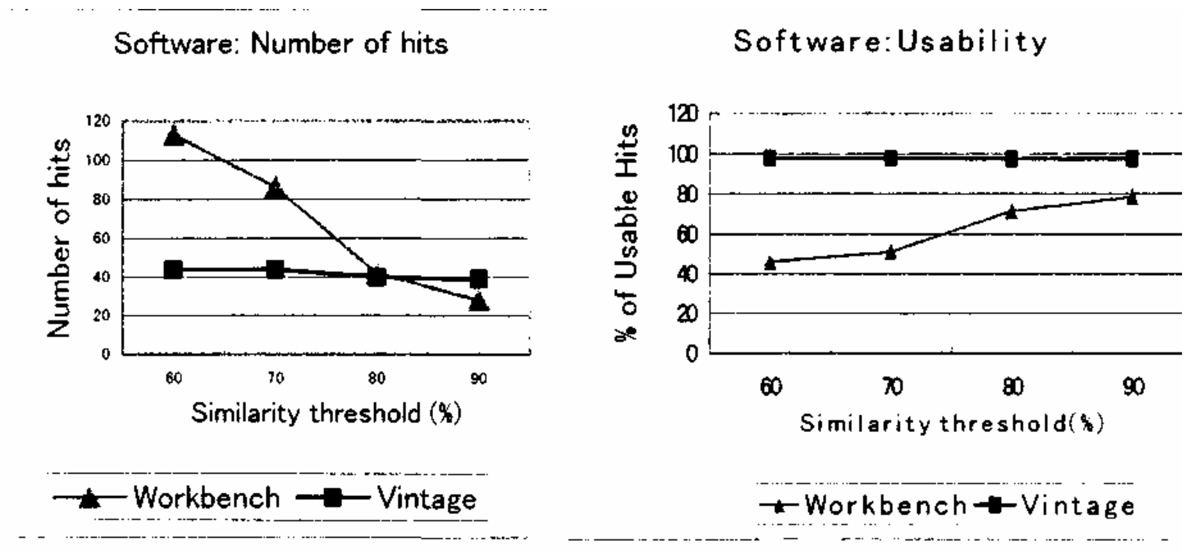


Figure 5: Compared performances on a software corpus

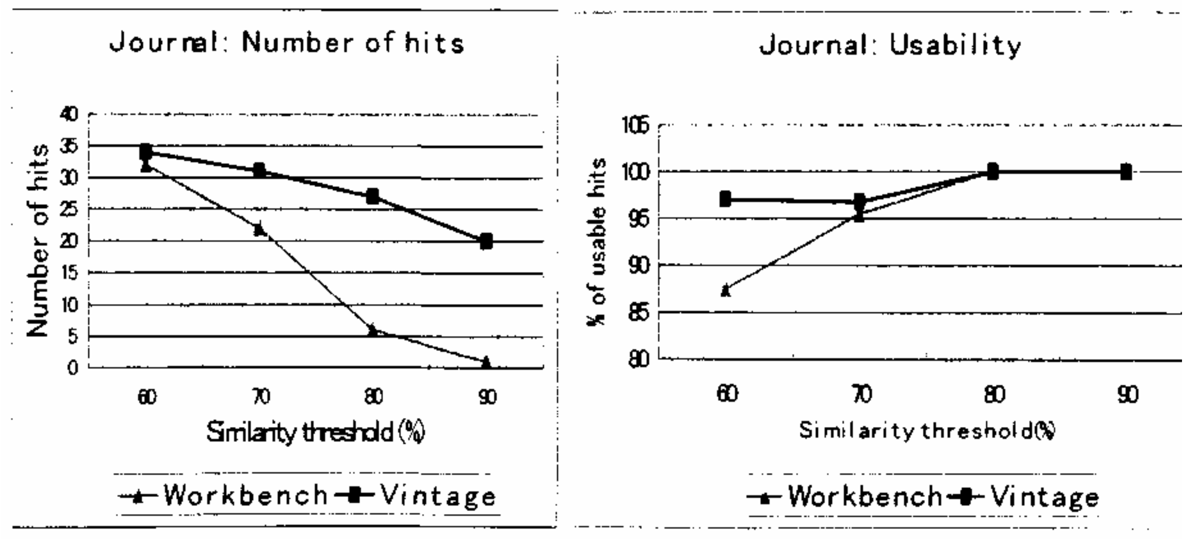


Figure 6: Compared performances on a journal corpus

We then estimated how many of these sentences could be used by a translator according to the following criteria: if for a given hit, less than half the words should be edited to get the input sentence, we considered the sentence usable. The second graphic of both Figures 5 and 6 show the percentage of usable hits.

It appears clearly from the first graphics that our system offers a better recall for higher similarity thresholds, which are the ones used for retrieving reasonable proposals. When the threshold decreases, the quality of the sentences our prototype proposes is slightly constant, while with Workbench numerous proposals diverge in precision. Note that the notion of similarity is different in Trados Workbench (based on the number of similar characters), so the comparison should be considered as a relative one.

For lower thresholds, a translator using our prototype would be shown less noisy sentences, more *really* similar ones: he will loose less time evaluating which is the best to be adapted or used. For very higher thresholds, he will be given more sentences. It is worth noticing than the precision we obtained never goes below 97% (for these tests), where classic TM precision would be irremediably degraded when the threshold similarity decreases, because of the almost exclusive index-based retrieval process they use.

In [Planas 1999a.b], we have checked that other classical TM like IBM Translation Manager and Star Transit show similar behaviors, while Xerox XMS Translation Manager that uses linguistic data is closest to our results. It is worth noting that these TM software are well designed, and allow everyday translation companies to save time and money. Our prototype is only a research program that has been built for testing the core behavior of TM. It does not include all the useful feature commercial TM software offer.

4.2 Shallow translating

Previous research like [Collins & Cunningham 1997] or [Lange et al. 1997], and [Veale & Way 1998] have shown that approaching translation as a transfer problem using TU as "surface patterns" could lead to interesting results for some specific situations: around 30% of "adaptable" Translation Units in software manuals for the first reference, and 63% for sentences that did have correspondences, for the second. We thought it could be interesting to try to get similar evidences in the case TU are used as surface patterns with substitutable words for translating *similar* sentences, without the use of any syntactic structures in the case of English to Japanese translation.

We have developed a specific algorithm that matches only equalities and deletions (we do not know where to insert words, only longer example sentences are usable for translating a shorter input sentence), and have applied it after a retrieval phase, like in the previous experiment. The target language parser was the NTT ALTJAWS parser for the Japanese language, provided by our colleagues from the Translation Group of NTT Communication Science labs [ALTJAWS 1999]. The French parser was an ad-hoc built parser. Using a dictionary look-up, we then replaced the corresponding target words in the target part of the TU. and eliminated superfluous words for getting a translation for the input sentence. This approach is really basic and does not take into consideration important problems like the correct choice of the target word (lexical and semantic ambiguity), or of the correct syntactic structure (syntactic ambiguity). Far more, in the examples below, we do not give the final inflected sentence, but only the sequence of lemmas. Those not negligible issues should be covered by further research. Yet, used for technical manuals or software translation (localization), it seemed to us that these results could be of some help. Here are some input English sentences (I), the corresponding closest TU retrieved from the database (TU has a source part (S) and a target part (T), and the French translation solution (R) proposed by the prototype.

I: Press on the *red button*.

S: Press on the *green window*
T: Appuyez sur la *fenêtre verte*

R: (Appuyer) (sur) (le) *(bouton) (rouge)*

I: Select "*Open*" and click "*OK*"

S: Select "*Now*" and click "*Done*"
T: Choisissez "*Maintenant*", et cliquez sur "*Fini*".

R: Choisissez "*Ouvrir*", et cliquez sur "*OK*".

This last example requires that a preference translation glossary has been filled into the system, to apply the company conventions. According to these conventions, "Open" should be translated into "Ouvrir", and "Done" into "Fini". We show now some English sentences and their Japanese translation. The sentences have been borrowed from our online economical news corpus:

I: *Sept.* Nikkei *100* options did not change hands.

S: *Aug.* Nikkei *300* options did not change hands.
T: 日経*300* の*8*月物は商いが成立しなかった。
(Nikkei 300 no 8 gatsu mono ha akinai ga seiritsushinakatta)
(Nikkei 300 of August thing -part- sell -part- conclude not)

R: 日経*１００*の *９*月物は商いが成立しなかった。

I: *KDD* and Japan Telecom fell back

S: *DDI* and Japan Telecom fell back, ...
T: *DDI*、テレコムが反落し、...
(DDI, telecom ga hanrakushi, ...)
(DDI, telecom -part- fell back, ...)

R: *KDD*、テレコムが反落し、...

In this first example, "８月"(Aug.) is changed to " ９月 " (Sept.) and " ３００" to " １００". In this second example "KDD" replaces "DDI".

5 Conclusion and Perspectives

We have designed a structure (TELA) for representing data to be used in Translation Memory tools. Using this multi-level structure, we have proposed a notion of similarity based on the Edit Distance, and we have shown with experiments that this approach gives better results than techniques used in current Translation Memories. Notably, we have increased the recall for higher similarity thresholds, and reduced the worst precision to 97%. in the tested corpora. Finally we have faked a step towards a possible extension of TM towards Machine Translation through the Shallow Translation paradigm, and shown some first results.

Yet a lot of work has to be done. Concerning the benefits of the TELA Structure as far as document structuring and layout is concerned, for example:

- We would want to test on a wide scale the layout transfer
- TELA has also been built to authorize easy connection with edition and linguistic tools, and we would also want to try to test this point.

The Shallow Translation paradigm should be studied more precisely, for example:

- Evaluating which syntactical construction would not fit into this framework
- Evaluating until which proportion of non similar items, a example could be used for translating a sentence
- We have planned the use of a semantic layer that would qualify each word by a sequence of the NTT semantic hierarchy concepts (3000 items) concepts. We would want to use this data for choosing the correct entry in a dictionary with semantic features, like NTT's GoiTake.

Acknowledgements

The whole NTT team should be thanked here for its constant support. A particularly thanks goes to Mr. Ikeda who provided the support for the ALT-JAWS and Apple Pie parsers, Mr. Adachi who provided the test files, and Mr. Bond who kindly accepted to review this article. Thanks also to Pr. Christian Boitet of Université Joseph Fourier (Grenoble, France) who supervised the Ph.D. thesis that leaded to most of the results presented here.

References

"Apple Pie parser". http://cs.nyu.edu/cs/projects/proteus/app/.

"ALT-JAWS". http:// www.kecl.ntt.co.jp/icl/mtg/resources/altjaws.html.

Argamon. S., Dagan I., Krymolowski Y. (1998). "A Memory-Based Approach to Learning Shallow Natural Language Patterns". Coling '98. Vol I. Montreal. Canada, pp 67-73.

Collins, B., Cunningham, P. (1997). "Adaptation Guided Retrieval: Approaching EBMT with Caution". 7th International Conference on Theoretical and Methodological issues in MT. pp 119-126.

Cranias, L., Papageorgiou, H., Piperidis, S. (1997). "Example retrieval from a Translation Memory". Natural Language Engineering 3(4), Cambridge University Press, pp 255-277.

Furuse, O., Iida, H. (1994). "Constituent Boundary Parsing for Example-Based Machine Translation". COLING'94. Kyoto, Japan, pp 105-111.

Lange J.-M., Gaussier E., Daille B. (1997). "Bricks and Skeletons: Some Ideas for the Near Future of MAHT", Machine Translation. #12, pp 40-51.

Planas, E. (1998). "TELA: Structures and Algorithms for Memory-Based Machine Translation". Ph.D. thesis, University Joseph Fourier, Grenoble, France (in French).

Planas. E. (1999a). "Towards Second Generation Translation Memories", in LISA Newsletter 1999, vol. VIII, n.2(June 1999). ISDN 1420-3693, pp 7-11.

Planas. E. (1999b). "Translation memory Tools: Comparative studies of Star, Trados. IBM TM2 and Shallow Translation", in LISA Forum-Asia Shanghai, Workshop Session proceedings, May 28.

Takeda, K. (1996). "Pattern-Based Machine Translation"., Coling'96, Copenhagen, Denmark, pp 1155-1158.

Thurmair, Gr. (1997). "Exchange Interfaces for Translation Tools". MT Summit VI, 29 Nov., San Diego, USA, pp 74-93.

Ukkonen, E. (1985). "Algorithms for Approximate String Matching". Information and Control, Academic Press Inc., pp 100-118.

Veale, T., Way, A. (1997). "Gaijin; A bootstrapping, Template-Driven Approach to Example-Based MT". http:// www.compapp/dcu.ie/tonyv/papers/ gaijin.html/, Dublin City University, Ireland.

"XML", http://www.xml/co/axml/axml.htmil/.

"W3C". http://www.w3.org/.

Wagner, A. R., Fisher M. (1974). "The String-to-String Correction Problem". Journal of the ACM, Vol 21. #1. January 1974, pp 168-173.