Proceedings of the International Multiconference on Computer Science and Information Technology, pp. 87 – 92

ISSN 1896-7094
© 2007 PIPS

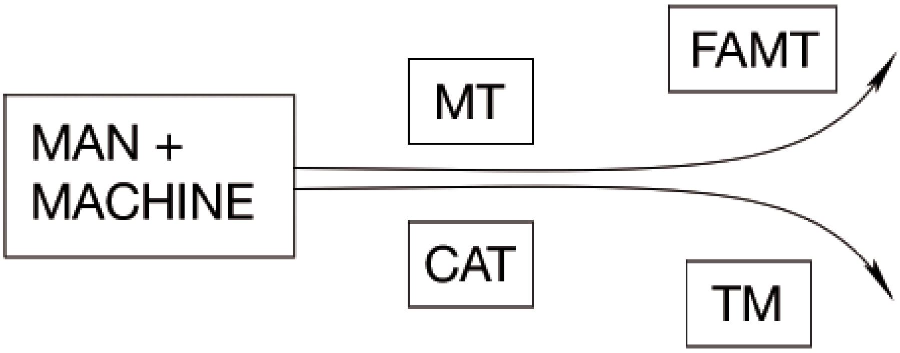
Using Regular Expressions in Translation Memories

Jacek Gintrowicz, Krzysztof Jassem

Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań
gintowicz@gmail.com, jassem@amu.edu.pl

Abstract. The paper presents the idea of using regular expressions in translation systems based on translation memories. Regular expressions are used in: search for a match in the instance sentence, search for an appropriate example in the translation memory and in the transfer of the instance sentence into its equivalent. The application of transfer rules to translation memories support the thesis, put forward in the paper, that Machine Translation and Computer-Aided Translation converge into the same direction.

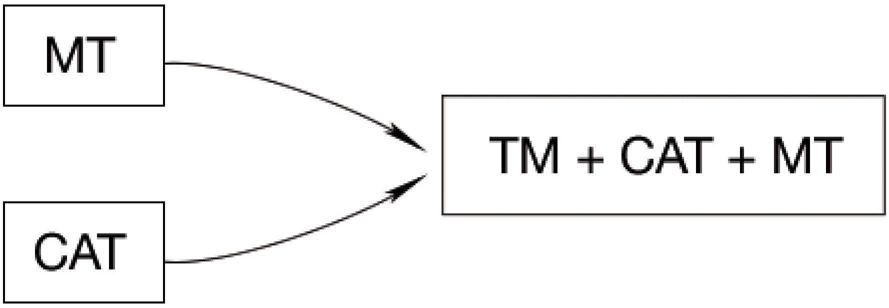
“While the original aim of the MT pioneers was fully automatic MT systems, there has also been, at least since the 1966 ALPAC report and possibly even earlier, the view that computers could be used to help humans in their translation task rather than replace them”. This citation comes from Harold Somers (Somers, 2003). The below figure may illustrate this view:



The idea of co-operation between man and machine goes in two directions: one direction is via Machine Translation (MT), possibly pre-edited or/and post-edited, towards Fully Automatic Machine Translation (FAMT) where there is nothing to do for a human. The other direction is via Computer-Aided Translation (CAT), where the user – human translator – is handily equipped with necessary glossaries and dictionaries and has the possibility to re-use his/her own translation, towards the use of large Translation Memories where there is almost nothing to do for a computer (except for just matching an instance text to an example that has once been translated).

It appears that the idea of Computer-Aided Translation is nowadays beating the idea of Machine Translation. The reason is mainly economical: The creation of a good-quality MT system is much more costly than that of a good-quality CAT system and the demand for the two types is comparable. It does not mean that there is no future for MT. Firstly, modern approaches, based on large corpora, such as: Statistical Machine Translation (SMT) [Brown, 1990, Ney, 2005] and Example-Based Machine Translation (EBMT) [Nagao, 1984, Hutchins, 2005] aim at building MT systems at minimum cost. Secondly, once the quality of MT becomes reliable, the demand for it will rapidly grow.

Still, the near future belongs to CAT. But we believe in the following scenario:



The producers of MT systems will incorporate more and more ideas and tools from CAT systems. On the other hand, CAT systems will include more and more machine work (some CAT systems can already be linked to MT procedures) .Eventually, there will be no difference between CAT and MT systems. There will be no translator's workbench or machine translation system – just one translation tool.

This paper shows a step in this direction. Given a translation memory and a CAT system that uses the memory, can we enhance the process of matching a sentence under translation to a memory unit? The enhancement should be executed fully automatically.

The idea presented in the paper is to enhance the procedure of matching by means of regular expressions.

The idea of using regular expressions for natural language processing is widely known. Regular expressions are most frequently applied to searching for regularly structured fragments, like numbers or dates. Kartunnen (1996) suggests applying finite automata and transducers that represent regular expressions, for natural language texts. Oflazer (2004) shows the use of regexps for tokenization, shallow parsing or morphology analysis. Hasan (2005) describes how regexps may be applied in SMT for sentence clustering.

Here, we present the use of regular expression in a translation system based on the translation memory. Regular expressions are used in: search for a match in the instance sentence, search for an appropriate example in the translation memory and finally in the transfer of the instance sentence into its equivalent.

Below we present the formalism that describes search-and-transfer rules developed for a CAT system, called TranslAide.

Each rule consists of:

A) Regular expressions defining search patterns in: Instance, Source TM and Target TM.

B) Transfer of the instance expression into an output expression.

Below, we show an example of a rule for translation from Polish into English. The rule converts one of Polish date formats into its English equivalent.

Suppose the instance sentence is:

Instance: *Egzamin jest zaplanowany na 18.02.2008.*

Suppose the translation memory contains the following example and its translation:

TM Source: *Egzamin jest zaplanowany na 15.01.2007.*

TM Target: *The exam is planned on 15/01/2007.*

We expect the algorithm to generate the following output sentence based on TM Target:

Output: *The exam is planned on 18/02/2008.*

The appropriate rule looks like this (only selected tags are listed for clarity):

```
Rule                                                                      1:
1.    <instance>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</instance>
2.    <source>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</source>
3.    <target>([0-9]{1,2})[\/]([0-9]{1,2})[\/]([0-9]{2,4})</target>
4.    <orders>
5.          <order        sourceGroup=”1”        suffix=”/”        />
6.          <order        sourceGroup=”2”        suffix=”/”        />
7.          <order        sourceGroup=”3”        suffix=””         />
8.    </orders>
```

Line 1 defines the search pattern for an input sentence (here, the pattern matches one of Polish date notations). The Instance must contain a matching string to become applicable to the rule.

Line 2 defines the search pattern for the source part of an example. (Here, the pattern is identical to that for the Instance). The source part of an example must contain a matching string to be considered as the source for translation

Line 3 defines the search pattern for the target part of an example. The target part of an example must contain a matching string to be considered as the target for translation

Lines 4. to 8. define the transfer of the regular expression: Line 5 refers to the first group of the instance surrounded by braces, i.e. *[0-9]{1,2}*. It says that in the output sentence the matching string should be added a suffix “/”.

Lines 6 and 7 refer to the second and third group of the instance respectively.

The <orders> tag makes it possible to permutate the order of the groups. The groups are listed inside the tag in the order expected in the target sentence (here, the order of the target is identical to that of the source).

Algorithm

The consecutive steps of the algorithm will be illustrated by applying Rule 2 and Rule 3 to the below sentences:

```
Rule 2:
<instance>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</instance>
<source>([0-9]{1,2})[\.]([0-9]{1,2})[\.]([0-9]{2,4})</source>
<target>[0-9{1,2})(st|nd|rd|th)?\s ((January)|(February)|(March)|(April)|(May)|
(June)|(July)|(August)|(September)|(October)|(November)|(December))\s([0-
9]{2,4})</target>
<orders>
        <order sourceGroup="1" sufix=" " />
        <order sourceGroup="2" sufix=" " />
        <order sourceGroup="3" sufix="" />
</orders>
<rules className="ToEnglishDateRules" />
```

Rule 2 is another rule for date conversion.

```
Rule                                                                        3:
<instance>([0-2]?[0-9])(?:\:([0-5]?[0-9])(?:[\:\.]([0-5]?[0-9]))?)?</instance>
<source>([0-2]?[0-9])(\:([0-5]?[0-9])([\:\.]([0-5]?[0-9]))?)?</source>
<target>((([0-1]?[0-9])([\:\.]([0-5]?[0-9])([\:\.]([0-5]?[0-
9]))?)?\s?[AaPp]\.?[Mm]\.?)|(([0-2]?[0-9])(\:([0-5]?[0-9])([\:\.]([0-5]?[0-
9]))?)))</target>
<orders>
        <order          sourceGroup="1"          sufix=":"          />
        <order          sourceGroup="2"          sufix=":"          />
        <order          sourceGroup="3"          sufix=""           />
</orders>
<rules className="ToEnglishTimeRules" />
```

Rule 3 searches for a Polish time format and converts it into an English time format while translating.

Instance: Egzamin jest zaplanowany na 12.05.2008, na godzinę 15:15.
TM Source : Egzamin jest zaplanowany na 02.03.2007, na godzinę 13:00.
TM Target: The exam is planned on 2 nd March 2007, at 1 PM.
Output: The exam is planned on 12 May 2008, at 3:15 p.m.

*0**Input:** Instance.

*1

```
For each rule in turn
 if regexp described in <instance> tag matches a fragment of Instance
extract the fragment, and name it RegExpInstance
 //Here, Rule 1 extracts the RegExpInstance: 12.05.2008

 For each TM Unit in turn //TM Unit is a pair <TMSource; TM Target>
        If regexp described in <target> matches a fragment of TM Target
        extract the fragment, and name it RegExpTarget
         //Here, the above TM Target is found and 2 nd March 2007 is
extracted
```

Separate RegExpInstance into groups according to positions of braces in the <instance> pattern
*//Here the **12.05.2008** is divided into 3 groups: **12** , **05** , **2008** .*

For each group use the conversion declared in <rules className>
*//Here, the rule named ToEnglishDateRules is invoked. The rule converts the 3 groups into **12, May, 2008** respectively*

Merge the converted groups according to the <orders> element.
//Here, the merged expression is: 12 May 2008, as the order of the groups is preserved

Place the merged expression into the TM Target surrounded by the tag **antra-changed**

//The following is obtained:
The exam is planned on <antra-changed> 12 May 2008 </antra-changed> , at 1 PM.
Still, the hour needs adjustment

Go back to the beginning. In the next loop do not touch texts surrounded by the **antra-changed** tag.
//Another loop yields:
*The exam is planned on **<antra-changed> 12 May 2008 </antra-changed> ,** at **<antra-changed>** 3:15 p.m **</antra-changed> .***

Clear up all <antra-changed> tags

So far, transfer rules have been developed for:

- various formats of date
- various formats of time
- currency expressions
- metric expressions
- numbers
- e-mail addresses.

Future Work

One direction of the development is to expand the set of rules. The sample ideas for the Polish language are: zip-codes, telephone numbers and URL addresses.

Another aspect is the transfer of formatting tags. The most obvious fields of interest seem to be XML and HTML formatting languages. The structure of tags is strictly defined, which allows for using the mechanism of regular expressions to recognize and transfer tags between texts in different languages.

Another possible direction is to expand the formalism beyond regular expressions. The idea is to replace regular expressions in the rules by lexical expressions based on bilingual dictionaries prepared beforehand. The authors have the access to multiple bilingual dictionaries developed for the MT system, Translatica (`www.translatica.pl`, `www.poleng.pl`). The first experiments with this idea look promising.

Summary

The transfer approach in Machine Translation has been widely criticized for high cost of rule preparation. The idea of using transfer rules in a CAT system, presented here, overcomes this problem: On the one hand a rule is not required to cover the whole sentence – only a part must match the pattern – which expands the application of a rule. On the other, thanks to using a general formalism of regular expressions, one rule may be matched to many fragments of texts.

A successful application of an MT idea (transfer rules) to a CAT system supports the thesis of the convergence between MT and CAT systems.

Bibliography

1. Brown P. F., Cocke J., Della Pietra S. A., Della Pietra V. J., Jelinek F., Lafferty J. D., Mercer R. L., Roossin P. S. (1990), *A statistical approach to machine translation* . Computational Linguistics 16 (2), pp. 79-85.
2. Hasan S., Ney H.(2005), *Clustered language models based on regular expressions for SMT*, 10th EAMT conference *Practical applications of machine translation*, 30-31 May 2005, Budapest; pp. 119-125.
3. Hodász G., Gröbler T., Kis B. (2004), *Translation memory as a robust example-based translation system* , 9th EAMT Workshop, Broadening horizons of machine translation and its applications, 26-27 April 2004, Malta; pp. 82-89.
4. Hutchins J. (2005), *Towards a definition of example-based machine translation* , MT Summit X, Phuket, Thailand, September 16, 2005, Proceedings of Second Workshop on Example-Based Machine Translation; pp.63-70.
5. Karttunen L., Chanod J-P., Grefenstette G., Schiller A.(1996), *Regular Expressions for Language Engineering*, Natural Language Engineering, Cambridge University Press, pp. 305-328.
6. Nagao M. (1984), *A framework of a mechanical translation between Japanese and English by analogy principle* , Artificial and human intelligence: edited review papers presented at the international NATO Symposium, October 1981, Lyons, France; ed. A. Elithorn and R. Banerji. Amsterdam: North Holland, pp. 173-180.
7. Ney H. (2005), *One decade of statistical machine translation: 1996-2005*, MT Summit X, Phuket, Thailand, September 13-15, 2005, Conference Proceedings: the tenth Machine Translation Summit: invited paper; pp. 12-17.
8. Oflazer K., Yilmaz Y, *Vi-xfst* (2004), *A Visual Regular Expression Development Environment for Xerox Finite State Tool*, In Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology, Association for Computational Linguistics, Barcelona, Spain, July 2004, pp 86-93.
9. Somers H. (ed.) (2003), *Computer and translation: a translator's guide* , John Benjamins Publishing, Amsterdam, page 11.