Automating MT post-editing using regular expressions

by Rafael Guzmán

This article was originally published in Multilingual
#90, Volume 18, Issue 6, pp. 49-52 (September 2007)

At the beginning of this century, Yves Champollion wrote, "Will the future of human translation be. . . *proofreading computer output*? The bad news is yes." Then he went on to clarify that "we may see, sooner than expected, an MT solution that provides decent translation."

Unfortunately, although machine translation (MT) systems have been improved quite significantly, linguistic quality still remains the biggest challenge for MT, as stated by Carnegie Mellon University researchers Ariadna Font-Llitjós and Jaime G. Carbonell. In the end, post-editors always end up having to manually post-edit the linguistic mess left behind by MT. Whether we like it or not, it certainly looks as if MT post-editing is here to stay for a good while.

In this context, MT would be more beneficial if post-editing could be highly automated. This means not just allowing users to change vocabulary and linguistic rules in the MT system, but automatically searching for and replacing linguistic patterns in the actual MT output.

In order to quickly achieve this, translators and linguists — not just engineers — need to rediscover and master the power of regular expressions. By using them, the most complex and repetitive linguistic errors can be identified and replaced with the right text in the MT output. It is interesting to notice that search-and-replace in text using regular expressions is a feature that is at least partially available in a wide range of tools used to localize software (such as CATALYST and Pasolo) and edit TMs (Olifant), as well as in many text editors (such as EditPad Pro, Notepad ++ and UltraEdit). This feature is, however, still non-existent in widely used MT systems.

Regular expressions and linguistic patterns

By *linguistic patterns*, I will refer to linguistic errors that tend to recur in sentences translated by a MT system. Suppose, for instance, that a MT system always translates trademarks such as Windows and Linux into Spanish adding *el* (*el Windows*, *el Linux*). This would be a linguistic pattern consisting of a redundant article. This mistake would have to be post-edited as many times as it occurs in each MT output.

Regarding regular expressions (also known as *regex* or *regexp*), they are used to search and extract or manipulate bodies of text based on regular patterns — in this case, linguistic patterns. The syntax of regular expressions can be simple or highly complex, depending on the pattern. This article will not explain how to create regular expressions — many books and online tutorials are available for that purpose — but will show their potential to automate MT post-editing.

The example mentioned above will help to illustrate the concept behind regular expressions. Based on that pattern, the following regular expression could be created to search for all the occurrences of *el* followed by *Windows* or *Linux* in the MT output:

```
Search=\bel (Windows|Linux)\b
```

\b are just word boundaries. The group (Windows|Linux) targets two possible trademarks, although more could be added to this group, as long as they are separated with |. Then another regular expression could be used to replace this pattern with the right text:

```
Replace=$1
```

$1 represents everything included in the first group between brackets (in this case, Windows and Linux). Since *el* is not included in the Replace field, all instances of *el Windows* and *el Linux* will be replaced with just *Windows* and *Linux* as shown in Figure 1.

The key thing to remember is that as long as a linguistic pattern is identified, a regular expression can be created to search for and fix the pattern.

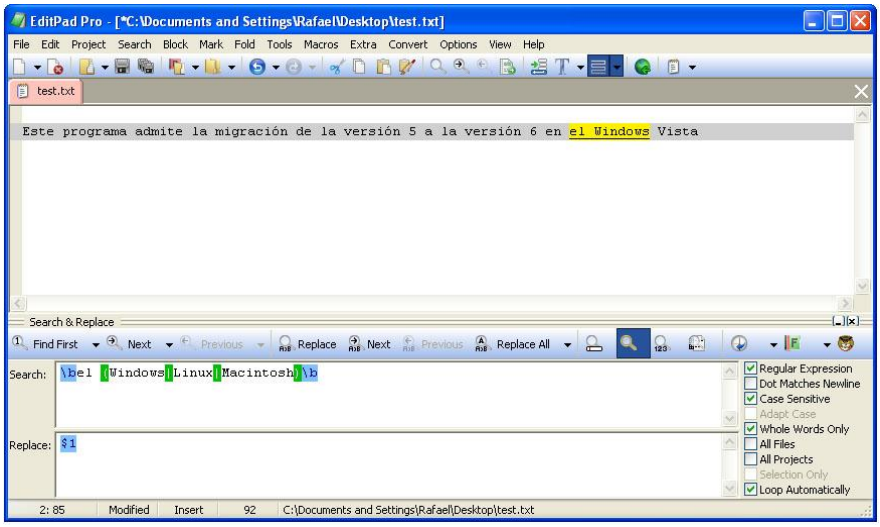

Fig. 1. Because regular expressions provide great flexibility to fine-tune search/replace criteria, they can be used to automatically post-edit MT outputs. Text editors such as EditPad Pro and UltraEdit support this type of functionality.

Examples of linguistic patterns in Spanish

The accompanying tables show a variety of typical linguistic patterns (bold text) in Spanish MT raw outputs and how they can be automatically post-edited if regular expressions are used to identify the wrong text and replace it with the right one. The principle behind these examples remains valid, of course, for other languages. The regular expressions used in these examples are Perl-based, and the translations were generated with a well-known rule-based MT system.

1. Misspellings

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
The verbs *desconectese* and *conectese* are missing an accent.	**search=**\b(D|d)(esconectese)\b **replace=**$1esconéctese **search=**\b(C|c)(onectese)\b **replace=**$1onéctese	When prompted, **disconnect** from the server.	Cuando se le pida, **desconectese** del servidor.	Cuando se le pida, **desconéctese** del servidor.
y should not precede a word beginning with *i*. Instead, it should be replaced with *e*.	**search=**(\by) ([iI]) **replace=**e $2	You need to obtain a license file **and import** it through the server.	Usted necesita obtener un archivo de licencia **y importarlo** a través del servidor.	Usted necesita obtener un archivo de licencia **e importarlo** a través del servidor.

2. Punctuation

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
If a comma appears before *y*, it should be deleted.	search=,(y\b) replace=$1	View details about all active sessions**, and** end a session when necessary.	Vea los detalles sobre todas las sesiones activas**, y** termine una sesión cuando sea necesario.	Ver los detalles sobre todas las sesiones activas **y** termine una sesión cuando sea necesario.

3. Articles

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
The article *El, el* should not precede trademarks, such as Windows, Macintosh, Linux.	**search=**\bel (Windows|Linux|Macintosh)\b **replace=**$1	This program supports migration from version 5 to version 6 **on Windows** Vista.	Este programa admite la migración de la versión 5 a la versión 6 **en el Windows** Vista	Este programa admite la migración de la versión 5 a la versión 6 **en Windows** Vista

4. Prepositions

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
The *para* preposition in front of infinitives should be deleted when following *en el cual*.	search=\b(en el cual) para\b replace=$1	Directory **into which to** place the client installation software.	Directorio **en el cual para poner** el software de instalación del cliente.	Directorio **en el cual poner** el software de instalación del cliente.

5. Grammatical agreement (sing vs plural)

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
Singular adjectives/ adverbs should be made plural by adding *-s* if preceded by *son*/*están*/*estén.*	**search=**\b(est[áé]n|son) ([\w-]+[aeiou])\b **replace=**$1 $2s	It automatically detects the connection devices **that are available** on your computer.	Detecta automáticamente los dispositivos de conexión **que están disponible** en su equipo.	Detecta automáticamente los dispositivos de conexión **que están disponibles** en su equipo.

6. Word order

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
Wrong word order produced by *that* relative clause in the source. *se* + verb + *que* needs to be replaced with *que se* + verb.	**search=**(\bse) (\w+?\b) (que) **replace=**$3 $1 $2	Name of the database user account **that is created.**	Nombre de la cuenta de usuario de base de datos **se crea que**.	Nombre de la cuenta de usuario de base de datos **que se crea**.
Word + *-based* is translated with the wrong word order: *word-basado* or *word-basada*. Instead, it should be *basado(a) en* + word, making it plural by adding *-s* to *basado(a)* if the preceding word is plural (for example, if it ends in *s*).	**search=**\b(\w+?s) (\w+?)-(basad[ao]s?)\b **replace=**$1 $3s en $2 **search=**\b(\w+?)-(basad[ao]s?)\b **replace=**$2 en $1	These are **Java-based** applets that run in your Web browser.	Estos son programas **Java-basado** que ejecutan en su navegador Web.	Estos son programas **basados en Java** que ejecutan en su navegador Web.

7. Reflexive pronoun: "-se"

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
ejecutando should be a pronominal verb with a reflexive meaning (that is, ends in *-se*) when followed by *en* and not preceded by *se*.	**search=**\b(.*[^se]) (est[áé]n?) (ejecutando) (en)\b **replace=**$1 $2 ejecutándose $4	The program is **running** in the server.	El programa está **ejecutando** en el servidor.	El programa está **ejecutándose** en el servidor.

8. Style: "usted debe, necesita, tiene que"

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
Stylistic issue: *Usted + debe/necesita/tiene que* at the beginning of a sentence should normally be replaced with a more impersonal phrase such as *Es necesario*.	**search=**\bUsted (debe|necesita|tiene que)\b **replace=**Es necesario	**You mus**t install the update to configure the connection to the server.	**Usted debe** instalar la actualización para configurar la conexión al servidor.	**Es necesario** instalar la actualización para configurar la conexión al servidor.

9. Redundancies

Linguistic pattern	Regular expression	Source	MT raw output	Automatically post-edited MT output
Most of the time, que followed by es/son is redundant and can be deleted.	**search=**\bque (es|son)\b **replace=**	If applicable, in the main window, configure the settings **that are** needed to connect to the server.	Si corresponde, en la ventana principal, configure las configuraciones **que son** necesarias para conectarse al servidor.	Si corresponde, en la ventana Detalles, configure las configuraciones necesarias para conectarse al servidor.

Practical recommendations

Trying to fix every single linguistic problem would obviously be unrealistic. The focus should be put instead on fixing patterns, but the lack of time will often make this unfeasible, too. As a result, frequency is a decisive factor to consider. Relevance should, however, also be taken into account. Less frequent patterns involving ambiguity and bad syntax may sometimes require far more time during manual post-editing than other linguistic patterns with a higher frequency.

Searching, organizing and fixing linguistic patterns should, therefore, be carried out in a systematic way.

Although searching for linguistic patterns during manual post-editing for future reference is possible, it is more practical to do this separately. This requires spending time doing tests and trying to understand how the MT system "behaves" when translating different types of content. Doing only occasional analysis — just before a translation project kicks off, for example — would be a fire-fighting approach. This approach brings little benefit in the medium-long term. There is always room for further analysis and refinement after every project that will bring benefits to future projects.

As linguistic patterns are identified, they can be organized in different categories such as "word order," "concordance," "spelling," "missing word in the MT dictionary" and so on. Metrics can be generated based on this categorization in order to get a clearer idea of what the main problems are. Using a spreadsheet template can be useful. One of the advantages of having patterns categorized is that they can be compared and fixed at a later stage if there is not enough time available at the present moment.

It must also be clarified that regular expressions are not meant to replace MT dictionaries. Instead, they are meant to be their complement (Fig. 1). This means that if a linguistic problem can be fixed by just updating the MT dictionary, a regular expression would be unnecessary.

Whenever a regular expression is created, the more flexible it is, the better. The same regular expression should fix as many potential linguistic scenarios as possible. The better defined a linguistic pattern is, of course, the more easily a regular expression can be implemented. Creating an appropriate regular expression may require some degree of creativity from the linguists. In this regard, identifying "anchor points" in sentences (such as commas, periods, and surrounding tags) will be useful. The implementation of regular expressions should in any case always be the final step in the analysis process.

Once the regular expressions are created, a process must be in place to automatically search and replace each individual pattern in the right sequence in the MT output file (Figure 2). This sequential search for patterns can be compared to the way *ezParse* uses a list of regular expressions created by the user in CATALYST to find and extract text within files. This type of functionality is, as I mentioned earlier, not currently available in well-known MT systems. While this is the case, in-house solutions will need to be developed with a little bit of creativity. In the worse of scenarios, post-editors can always avail themselves of the Find/Replace feature in text editors that support regular expressions.

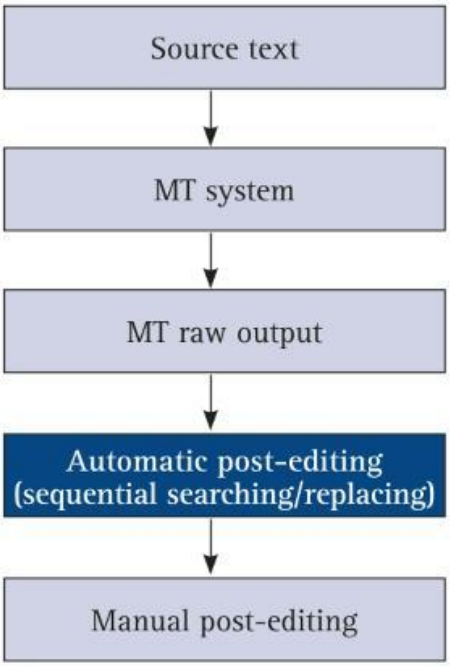

Fig. 2: The result of automating MT post-editing is a significant reduction of manual post-editing and cost savings.

Finally, it is advisable to always check that regular expressions are used properly and implemented in the right sequence. Unexpected linguistic side effects may otherwise occur. If Spanish infinitives, for instance, are assumed to be just words ending in *-ar*, *-er* and *-ir* and, based on this assumption, a regular expression is created to delete articles preceding infinitives, patterns such as *el lugar*, *el primer* and *el router* will be damaged unintentionally.

Conclusion

MT post-editing can be highly automated if regular expressions are used to search and replace linguistic patterns in MT raw outputs. This approach significantly reduces manual post-editing and the costs involved. On the other hand, translators and linguists need to invest time mastering regular expressions and analyzing MT raw outputs. Appropriate training ideally should be part of translation curricula. While MT systems do not provide an appropriate post-editing environment, in-house solutions need to be developed to carry out sequential searches and replacements using regular expressions.

Finally, the type of automatic post-editing described in this article has been tested using a ruled-based MT system (as opposed to a statistical MT system), where linguistic patterns are more likely to occur in a consistent way. It is unlikely that this approach would be as useful in outputs generated by statistical MT systems. This will in any case require further research.

References

Butt, J. & Benjamin, C., 2004. *A New Reference Grammar of Modern Spanish*. Hodder Arnold, London.

Champollion, Y., 2001. "Machine translation (MT), and the future of the translation industry," *Translation Journal*, vol. 5, No. 1. Retrieved: May 10, 2007, from: http://accurapid.com/journal/15mt.htm

Font-Llitjós, A. & Carbonell, J. G., 2006, "Automating Post-Editing to Improve MT Systems." Paper presented at the AMTA Automated Post-Editing Techniques and Applications Workshop, Cambridge, 2006.

Friedl, J., 2002. *Mastering Regular Expressions*. O'Reilly Media, Sebastopol.
www.regular-expressions.info http://en.wikipedia.org/wiki/ Regular_expressions

Acknowledgements

The author thanks Dr. Johann Roturier for introducing him to the use of regular expressions for the purposes explained in this article.

About the author

Rafael Guzmán's personal website is at www.rafaelguzman.ie