Open-Source Portuguese–Spanish Machine Translation

Carme Armentano-Oller, Rafael C. Carrasco, Antonio M. Corbí-Bellot, Mikel L. Forcada*, Mireia Ginestí-Rosell, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Gema Ramírez-Sánchez, Felipe Sánchez-Martínez, and Miriam A. Scalco

Transducens Group,
Departament de Llenguatges i Sistemes Informàtics,
Universitat d'Alacant, E-03071 Alacant, Spain
*mlf@ua.es

Abstract. This paper describes the current status of development of an open-source shallow-transfer machine translation (MT) system for the [European] Portuguese ↔ Spanish language pair, developed using the OpenTrad Apertium MT toolbox (www.apertium.org). Apertium uses finite-state transducers for lexical processing, hidden Markov models for part-of-speech tagging, and finite-state-based chunking for structural transfer, and is based on a simple rationale: to produce fast, reasonably intelligible and easily correctable translations between related languages, it suffices to use a MT strategy which uses shallow parsing techniques to refine *word-for-word* MT. This paper briefly describes the MT engine, the formats it uses for linguistic data, and the compilers that convert these data into an efficient format used by the engine, and then goes on to describe in more detail the pilot Portuguese↔Spanish linguistic data.

1 Introduction

This paper presents the current status of development of an open-source (OS) shallow-transfer machine translation (MT) system for the [European] Portuguese ↔ Spanish language pair, developed using the recently released OpenTrad Apertium MT toolbox (http://www.apertium.org), Apertium for short. Apertium is based on an intuitive approach: to produce fast, reasonably intelligible and easily correctable translations between related languages, it suffices to use a MT strategy which uses shallow parsing techniques to refine fixed-equivalent, *word-for-word* machine translation.

Apertium uses finite-state transducers for lexical processing (powerful enough to treat many kinds of multi-word expressions), hidden Markov models (HMM) for part-of-speech tagging (solving categorial lexical ambiguity), and finite-state-based chunking for structural transfer (local *structural* processing based on simple and well-formulated rules for some simple structural transformations such as word reordering, number and gender agreement, etc.).

R. Vieira et al. (Eds.): PROPOR 2006, LNAI 3960, pp. 50–59, 2006.

This design of Apertium is largely based on that of existing systems such as interNOSTRUM[1] (Spanish↔Catalan, [1]) and Tradutor Universia[2] (Spanish↔Brazilian Portuguese, [2]), systems that are publicly accessible through the net and used on a daily basis by thousands of users.

The Apertium toolbox has been released as OS software;[3] this means that anyone having the necessary computational and linguistic skills can adapt it to a new purpose or use it to produce a MT system for a new pair of related languages.

In addition to the toolbox, OS data are available for three language pairs: Spanish–Catalan and Spanish–Galician, developed inside the OpenTrad consortium,[4] and more recently, Spanish–European Portuguese, developed by the authors and described in this paper. Prototypes for all three pairs may also be tested on plain, RTF, and HTML texts and websites at the address `http://www.apertium.org`.

The introduction of open-source MT systems like Apertium may be expected to ease some of the problems of closed-source commercial MT systems: having different technologies for different pairs, closed-source architectures being hard to adapt to new uses, etc. It will also help shift the current business model from a licence-centered one to a services-centered one, and favor the interchange of existing linguistic data through the use of standard formats.

The Spanish↔Portuguese language pair is one of the largest related-language pairs in the world; this is one of the main reasons to release pilot OS data for this pair. We believe this may motivate researchers and groups to improve these data or adapt them to other variants of Portuguese such as Brazilian Portuguese, and collaborate to develop, in the near future, a high-quality, free, OS Portuguese↔Spanish MT system.

This paper briefly describes the MT engine (sec. 2), the formats it uses for linguistic data (sec. 3), the compilers that convert these data into an efficient format used by the engine (sec. 4), and the pilot Spanish↔Portuguese linguistic data (sec. 5). Brief concluding remarks are given in sec. 6.

2 The Apertium Architecture

The MT strategy used in the system has already been described in detail [1, 2]; a sketch (largely based on that of [3]) is given here.

The MT engine is a classical shallow-transfer or *transformer* system consisting of an 8-module assembly line (see figure 1); we have found that this strategy is sufficient to achieve a reasonable translation quality between related languages such as Spanish and Portuguese. While, for these languages, a rudimentary word-for-word MT model may give an adequate translation for 75% of the text,[5]

[1] `http://www.internostrum.com`
[2] `http://traductor.universia.net`
[3] Under the GNU General Public License, `http://www.gnu.org/licenses/gpl.html`
[4] `http://www.opentrad.org`
[5] Measured as the percentage of the words in a text that do not need correction

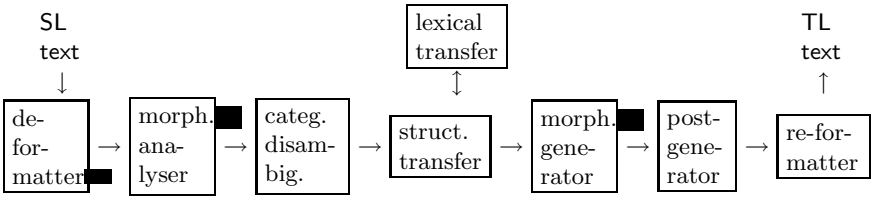

Fig. 1. The eight modules of the Apertium MT system (see section 2)

the addition of homograph disambiguation, management of contiguous multiword units, and local reordering and agreement rules may raise the fraction of adequately translated text above 90%. This is the approach used in the engine presented here.

To ease diagnosis and independent testing, modules communicate between them using text streams (see examples below). Most of the modules are capable of processing tens of thousands of words per second on current desktop workstations; only the structural transfer module lags behind at several thousands of words per second. A description of each module follows.

The de-formatter: separates the text to be translated from the format information (RTF and HTML tags, whitespace, etc.). Format information is encapsulated in brackets so that the rest of the modules treat it as blanks between words. For example, the HTML text in Portuguese "`vi <em>a bola</em>`" ("I saw the ball") would be transformed by the de-formatter into "`vi[ <em>]a bola[</em>]`".[6]

The morphological analyser: tokenizes the text in *surface forms* (lexical units as they appear in texts) and delivers, for each surface form (SF), one or more *lexical forms* (LFs) consisting of *lemma*, *lexical category* and *morphological inflection information*. For example, upon receiving the example text in the previous section, the morphological analyser would deliver

```
^vi/ver<vblex><ifi><1><sg>$[ <em>]
^a/a<pr>/o<det><def><f><sg>/o<prn><pro><3><f><sg>$
^bola/bola<n><f><sg>$[</em>]
```

where each SF has been analysed into one or more LFs: *vi* is analysed into lemma *ver*, lexical category *lexical verb* (`vblex`), indefinite indicative (`ifi`), 1st person, singular; *a* (a homograph) receives three analyses: *a*, preposition; *o*, determiner, definite, feminine singular ("the"), and *o*, proclitic pronoun, 3rd person, feminine, singular ("her"), and *bola* is analyzed into lemma *bola*, noun, feminine, singular. The characters "`^`" and "`$`" delimit the analyses for each SF; LFs for each SF are separated by "`/`"; angle brackets "`<...>`" are used to delimit grammatical symbols. The string after the "`^`" and before the first "`/`" is the SF as it appears in the source input text.[7]

[6] As usual, the escape symbol \ is used before symbols `[` and `]` if present in the text.
[7] The \ escape symbol is used before these special characters if present in the text.

Tokenization of text in SFs is not straightforward due to the existence, on the one hand, of contractions, and, on the other hand, of multi-word lexical units. For contractions, the system reads in a single SF and delivers the corresponding sequence of LFs (for instance, the Portuguese preposition-article contraction *das* would be analysed into two LFs, one for the preposition *de* and another one for the article *as*). Multi-word SFs are analysed in a left-to-right, longest-match fashion; for instance, the analysis for the Spanish preposition *a* would not be delivered when the input text is *a través de* ("through"), which is a multi-word preposition in Spanish.

Multi-word SFs may be invariable (such as multi-word prepositions or conjunctions) or inflected (for example, in Portuguese, *tinham saudades*, "they missed", is a form of the imperfect indicative tense of the verb *ter saudades*, "to miss"). Limited support for some kinds of inflected discontinuous multi-word units is also available. The module reads in a binary file compiled from a source-language (SL) morphological dictionary (see section 3).

The part-of-speech tagger: As has been shown in the previous example, some SFs (about 30% in Romance languages) are homographs, ambiguous forms for which the morphological analyser delivers more than one LF; when translating between related languages, choosing the wrong LF is one of the main sources of errors. The part-of-speech tagger tries to choose the right LF according to the possible LFs of neighboring words. The part-of-speech tagger reads in a file containing a first-order hidden Markov model (HMM, [4]) which has been trained on representative SL texts. Two training modes are possible: one can use either a larger amount (millions of words) of untagged text processed by the morphological analyser or a small amount of tagged text (tens of thousands of words) where a LF for each homograph has been manually selected. The second method usually leads to a slightly better performance (about 96% correct part-of-speech tags, considering homographs and non-homographs). The behavior of the part-of-speech tagger and the training program are both controlled by a tagger definition file (see section 3). The result of processing the example text delivered by the morphological analyser with the part-of-speech tagger would be:

```
^ver<vblex><ifi><1><sg>$[ <em>]
^a<det><def><f><sg>$
^bola<n><f><sg>$[</em>]
```

where the correct LF (determiner) has been selected for the word *a*.

The lexical transfer module: is called by the structural transfer module (described below); it reads each SL LF and delivers a corresponding target-language (TL) LF. The module reads in a binary file compiled from a bilingual dictionary (see section 3). The dictionary contains a single equivalent for each SL entry; that is, no word-sense disambiguation is performed. For some words, multi-word entries are used to safely select the correct equivalent in frequently-

occurring fixed contexts.[8] This approach has been used with very good results in Tradutor Universia and interNOSTRUM. Each of the LFs in the running example would be translated into Spanish as follows:

```
ver<vblex> : ver<vblex>
o<det> : el<det>
bola<n><f> : balón<n><m>
```

where the remaining grammatical symbols for each LF would be simply copied to the TL output. Note the gender change when translating *bola* to *balón*.

The structural transfer module: uses finite-state pattern matching to detect (in the usual left-to-right, longest-match way) fixed-length patterns of LFs (chunks or phrases) needing special processing due to grammatical divergences between the two languages (gender and number changes to ensure agreement in the TL, word reorderings, lexical changes such as changes in prepositions, etc.) and performs the corresponding transformations. This module is compiled from a transfer rule file (see below). In the running example, a *determiner–noun* rule is used to change the gender of the determiner so that it agrees with the noun; the result is

```
^ver<vblex><ifi><1><sg>$
[ <em>]^o<det><def><m><sg>$
^balón<n><m><sg>$[</em>]
```

The morphological generator: delivers a surface (inflected) form for each TL LF. The module reads in a binary file compiled from a TL morphological dictionary (see section 3). The result for the running example would be `vi[ <em>]el balón[</em>]`.

The post-generator: performs orthographic operations such as contractions and apostrophations. The module reads in a binary file compiled from a rule file expressed as a dictionary (section 3). The post-generator is usually *dormant* (just copies the input to the output) until a special *alarm* symbol contained in some TL SFs *wakes it up* to perform a particular string transformation if necessary; then it goes *back to sleep*. For example, in Portuguese, clitic pronouns in contact may contract: *me* ("to me") and *o* ("it", "him") contract into *mo*, or prepositions such as *de* ("of") may contract with determiners like *aquele* ("that") to yield contractions such as *daquele*. To signal these changes, linguists prepend an *alarm* symbol to the TL SFs *me* and *de* in TL dictionaries and write post-generation rules to effect the changes described.

The re-formatter: restores the format information encapsulated by the deformatter into the translated text and removes the encapsulation sequences used to protect certain characters in the SL text. The result for the running example would be the correct Spanish translation of the HTML text: `vi <em>el balón</em>`.

[8] For example, the Portuguese word "bola" ("ball") would be translated as "balón", but as "pelota" when it is part of the multiword unit "bola de tenis".

3 Formats for Linguistic Data

An adequate documentation of the code and auxiliary files is crucial for the success of OS software. In the case of a MT system, this implies carefully defining a systematic format for each source of linguistic data used by the system.

Apertium uses XML[9]-based formats for linguistic data for interoperability; in particular, for easier parsing, transformation, and maintenance. The XML formats for each type of linguistic data are defined through conveniently-designed XML document-type definitions (DTDs) which may be found inside the `apertium` package (available through `http://www.apertium.org`). On the one hand, the success of the OS MT engine heavily depends on the acceptance of these formats by other groups;[10] acceptance may be eased by the use of an interoperable XML-based format which simplifies the transformation of data from and towards it. But, on the other hand, acceptance of the formats also depends on the success of the translation engine itself.

Dictionaries (lexical processing): Monolingual morphological dictionaries, bilingual dictionaries and post-generation dictionaries use a common format.

Morphological dictionaries establish the correspondences between SFs and LFs and contain (a) a definition of the alphabet (used by the tokenizer), (b) a section defining the grammatical symbols used in a particular application to specify LFs (symbols representing concepts such as noun, verb, plural, present, feminine, etc.), (c) a section defining paradigms (describing reusable groups of correspondences between parts of SFs and parts of LFs), and (d) one or more labelled dictionary sections containing lists of SF–LF correspondences for whole lexical units (including contiguous multi-word units). Paradigms may be used directly in the dictionary sections or to build larger paradigms (at the conceptual level, paradigms represent the regularities in the inflective system of the corresponding language).

Bilingual dictionaries have a similar structure but establish correspondences between SL LFs and TL LFs.

Finally, *post-generation dictionaries* are used to establish correspondences between input and output strings corresponding to the orthographic transformations to be performed by the post-generator on the TL SFs generated by the generator.

Tagger definition: SL LFs delivered by the morphological analyser are defined in terms of fine part-of-speech tags (for example, the Portuguese word *cantávamos* has lemma *cantar*, category *verb*, and the following inflection information: *indicative, imperfect, 1st person, plural*), which are necessary in some parts of the MT engine (structural transfer, morphological generation); however, for the purpose of efficient disambiguation, these fine part-of-speech tags may be manually grouped in coarser part-of-speech tags (such as "verb in personal form"). In the tagger definition file (a) coarser tags are defined in terms of fine

[9] `http://www.w3.org/XML/`

[10] This is indeed the mechanism by which *de facto* standards appear.

tags, both for single-word and for multi-word units, (b) constraints may defined to forbid or enforce certain sequences of part-of-speech tags, and (c) priority lists are used to decide which fine part-of-speech tag to pass on to the structural transfer module when the coarse part-of-speech tag contains more than one fine tag. The tagger definition file is used to define the behavior of the part-of-speech tagger both when it is being trained on a SL corpus and when it is running as part of the MT system.

Structural transfer: rule files contain pattern–action rules describing what has to be done for each pattern (much like in languages such as `perl` or `lex` [5]). Patterns are defined in terms of categories which are in turn defined (in the preamble) in terms of fine morphological tags and, optionally, lemmas for lexicalized rules. For example, a commonly used pattern, *determiner–noun*, has an associated action which sets the gender and number of the determiner to those of the noun to ensure gender and number agreement.

De-formatters and re-formatters: are generated also from *format management files*. These are not linguistic data but are considered in this section for convenience. Format management files for RTF (rich text format), HTML (hypertext markup language) and plain text are provided in package `apertium`. The corresponding compilers generate C++ de-formatters and re-formatters for each format using `lex` [5] as an intermediate format.

4 Compilers

The Apertium toolbox contains compilers to convert the linguistic data into the corresponding efficient form used by the modules of the engine. Two main compilers are used in this project: one for the four lexical processing modules of the system and another one for the structural transfer.

Lexical processing: The four lexical processing modules (morphological analyser, lexical transfer, morphological generator, post-generator) are implemented as a single program which reads binary files containing a compact and efficient representation of a class of finite-state transducers (letter transducers, [6]; in particular, augmented letter transducers [7]). The lexical processor compiler [8] is very fast (it takes seconds to compile the current dictionaries in the system) which makes linguistic data development easy: the effect on the whole system of changing a rule or a lexical item may be tested almost immediately.

Structural transfer: The current structural transfer compiler (version 0.9.1 of `apertium`) reads in a structural transfer rule file and generates a C++ structural transfer module using `lex` [5] as an intermediate step. This makes it mandatory to recompile the engine each time the structural transfer data change; we are currently working on a precompiled format for transfer rules which would be read in by a general structural transfer module.

5 Portuguese↔Spanish Data

Lexical data: Currently, the Portuguese morphological dictionary contains 9700 lemmas; the Spanish morphological dictionary, 9700 lemmas, and the Spanish–Portuguese bilingual dictionary, 9100 lemma–lemma correspondences.

Lexical disambiguation: The tagset used by the Portuguese (resp. Spanish) HMM [4] lexical disambiguator consists of 128 (resp. 78) coarse tags (80 —resp. 65— single-word and 48 —resp. 13— multi-word tags for contractions, etc.) grouping the 13,684 (resp. 2,512) fine tags (412 (resp. 374) single-word and 13,272 (resp. 2,143) multi-word tags) generated by the morphological analyser.[11] The number of parameters in the HMM is drastically reduced by grouping words in ambiguity classes [4] receiving the same set of part-of-speech tags: 459 (resp. 260) ambiguity classes result. In addition, a few words such as *a* (article or preposition) or *ter* (*to have*, auxiliary verb or lexical verb) are assigned special hidden states. The Spanish lexical disambiguator has similar figures.

The current Portuguese (resp. Spanish) disambiguator has been trained as follows: initial parameters are obtained in a supervised manner from a 29,214-word (resp. 22,491-word) hand-tagged text and the resulting tagger is retrained (using Baum-Welch re-estimation as in [4]) in an unsupervised manner over a 454,197-word (resp. 520,091-word) text. Using an independent 6,487-word (resp. 24,366-word) hand-tagged text, the observed coarse-tag error rate is 4,0% (resp. 2,9%).

Before training the tagger we forbid certain impossible bigrams, such as *ter* as a lexical verb (translated into Spanish as *tener*) before any participle, so that in that case, *ter* is translated as an auxiliary verb (translated as *haber*).

Structural transfer data: The Portuguese↔Spanish structural transfer uses about 90 rules (the Spanish↔Portuguese figures are similar). The main group of rules ensures gender and number agreement for about 20 very frequent noun phrases (determiner–noun, numeral–noun, determiner–noun–adjective, determiner–adjective–noun etc.), as in *um sinal vermelho* (Portuguese, masc.) ["a red signal"]) → *una señal roja* (Spanish, fem.). In addition, we have rules to treat very frequent Portuguese–Spanish transfer problems, such as these:

- Rules to ensure the agreement of adjectives in sentences with the verb *ser* ("to be") to translate, for example, *O sinal é vermelho* (Portuguese, masculine, "The signal is red") into *La señal es roja* (Spanish, feminine).
- Rules to choose verb tenses; for example, Portuguese uses the subjunctive future (*futuro do conjuntivo*) both for temporal and hypothetical conditional expressions (*quando vieres* ["when you come"], *se vieres* ["if you came"]) whereas Spanish uses the present subjunctive in temporal expressions (*cuando vengas*) but imperfect subjunctive for conditionals (*si vinieras*).
- Rules to rearrange clitic pronouns (when enclitic in Portuguese and proclitic in Spanish or vice versa): *enviou-me* (Portuguese) → *me envió* (Spanish)

[11] The number of fine tags in Portuguese is high due to mesoclitics in verbs.

["he/she/it sent me"]; *para te dizer* (Portuguese) → *para decirte* (Spanish) ["to tell you"], etc.
- Rules to add the preposition *a* in some modal constructions (*vai comprar* (Portuguese) → *va a comprar* (Spanish) ["is going to buy"]).
- Rules for comparatives, both to deal with word order (*mais dois carros* (Portuguese) → *dos coches más* (Spanish) ["two more cars"]) and to translate *do que* (Portuguese) ["than"] as *que* (Spanish) in Portuguese comparative constructs such as *mais... do que...*
- Lexical rules, for example, to decide the correct translation of the adverb *muito* (Portuguese) → *muy/mucho* (Spanish) ["very", "much"] or that of the adjective *primeiro* (Portuguese) → *primer/primero* (Spanish) ["first"].
- A rule to translate the progressive Portuguese structure *estar a* + infinitive into the Spanish structure *estar* + gerund (`en` *to be* + *-ing*), and vice versa.
- Rules to make some syntactic changes like those needed to correctly translate the Portuguese construction "gosto de cantar" ("I like to sing") into Spanish as "me gusta cantar". Note that simple syntactic changes can be performed despite Apertium does not perform syntactic analysis.

Post-generation data: Current post-generation files for Spanish contain 26 entries using 5 paradigms grouping common post-generation operations. The most common Spanish post-generation operations include preposition–determiner contractions, or using the correct form of Spanish coordinative conjunctions *y/e*, *o/u* depending on the following vowel. On the other hand, current post-generation files for Portuguese contain 54 entries with 16 paradigms grouping common post-generation operations. Portuguese post-generation operations include clitic–clitic, preposition–determiner, and preposition–pronoun contractions.

A quick evaluation: With the described data, the text coverage of the Portuguese–Spanish (resp. Spanish–Portuguese) system is 92.8% (resp. 94,3%) as measured on a 5,294-word (resp. 5,028-word) corpus gathered from various sources. The translated word error rate (including unknown words) is 10.5% (resp. 8.3%). Speed surpasses 5000 words per second on an desktop PC equipped with a Pentium IV 3 GHz processor.

6 Concluding Remarks

We have presented the application of the OpenTrad Apertium open-source shallow-transfer machine translation toolbox to the Portuguese–Spanish language pair. Promising results are obtained with the pilot open-source linguistic data released (less than 10000 lemmas and less than 100 shallow transfer rules) which may easily improve (down to error rates around 5%, and even lower for specialized texts), mainly through lexical contributions from the linguistic communities involved. Note that the OpenTrad Apertium open-source engine itself is still being actively developed and contributions to its design may enhance it to perform more advanced lexical and structural processing tasks so that it can deal with more general language pairs.

Acknowledgements. Work funded by the Spanish Ministry of Industry, Tourism and Commerce through grants FIT-340101-2004-0003 and FIT-340001-2005-2, and partially supported by the Spanish Comisión Ministerial de Ciencia y Tecnología through grant TIC2000-1599-CO2-02. Felipe Sánchez-Martínez is supported by the Ministry of Education and Science and the European Social Fund through graduate scholarship BES-2004-4711.

References

1. Canals-Marote, R., Esteve-Guillen, A., Garrido-Alenda, A., Guardiola-Savall, M., Iturraspe-Bellver, A., Montserrat-Buendia, S., Ortiz-Rojas, S., Pastor-Pina, H., Perez-Antón, P., Forcada, M.: The Spanish-Catalan machine translation system interNOSTRUM. In: Proceedings of MT Summit VIII: Machine Translation in the Information Age. (2001) Santiago de Compostela, Spain, 18–22 July 2001.
2. Garrido-Alenda, A., Gilabert Zarco, P., Pérez-Ortiz, J.A., Pertusa-Ibáñez, A., Ramírez-Sánchez, G., Sánchez-Martínez, F., Scalco, M.A., Forcada, M.L.: Shallow parsing for Portuguese-Spanish machine translation. In Branco, A., Mendes, A., Ribeiro, R., eds.: Language technology for Portuguese: shallow processing tools and resources. Edições Colibri, Lisboa (2004) 135–144
3. Corbí-Bellot, A.M., Forcada, M.L., Ortiz-Rojas, S., Pérez-Ortiz, J.A., Ramírez-Sánchez, G., Sánchez-Martínez, F., Alegria, I., Mayor, A., Sarasola, K.: An open-source shallow-transfer machine translation engine for the romance languages of Spain. In: Proceedings of the Tenth Conference of the European Association for Machine Translation. (2005) 79–86
4. Cutting, D., Kupiec, J., Pedersen, J., Sibun, P.: A practical part-of-speech tagger. In: Third Conference on Applied Natural Language Processing. Association for Computational Linguistics. Proceedings of the Conference, Trento, Italy (1992) 133–140
5. Lesk, M.: Lex — a lexical analyzer generator. Technical Report 39, AT&T Bell Laboratories, Murray Hill, N.J. (1975)
6. Roche, E., Schabes, Y.: Introduction. In Roche, E., Schabes, Y., eds.: Finite-State Language Processing. MIT Press, Cambridge, Mass. (1997) 1–65
7. Garrido-Alenda, A., Forcada, M.L., Carrasco, R.C.: Incremental construction and maintenance of morphological analysers based on augmented letter transducers. In: Proceedings of TMI 2002 (Theoretical and Methodological Issues in Machine Translation, Keihanna/Kyoto, Japan, March 2002). (2002) 53–62
8. Ortiz-Rojas, S., Forcada, M.L., Ramírez-Sánchez, G.: Construcción y minimización eficiente de transductores de letras a partir de diccionarios con paradigmas. Procesamiento del Lenguaje Natural (35) (2005) 51–57