Intelligent Information Systems 2008
ISBN 978-83-60434-44-4, pages 269–278

A Method of Hybrid MT for Related Languages

Petr Homola and Vladislav Kuboň

Institute of Formal and Applied Linguistics, Charles University, Prague, Czech Republic

Abstract

The paper introduces a hybrid approach to a very specific field in machine translation — the translation of closely related languages. It mentions previous experiments performed for closely related Scandinavian, Slavic, Turkic and Romanic languages and describes a novel method, a combination of a simple shallow parser of the source language (Czech) combined with a stochastic ranker of (parts of) sentences in the target language (Slovak, Russian). The ranker exploits a simple stochastic model of the target language and its main task is to choose the best translation among those provided by the system. The last section of the paper presents results indicating better translation quality compared to the existing MT system for Czech and Slovak and compares them top the results obtained by the translation from Czech to Russian using the same system.

1 Introduction

The problem of a combination of rule-based and stochastic methods in some kind of a hybrid machine translation architecture is becoming more and more popular recently (Vandeghinste *et al.* (2006)). The hybrid architecture may help to combine the best methods of both approaches in order to obtain better translation quality. The combination is not easy, primarily because the translation task itself is very complex and difficult. One field where the complexity of the problem is definitely lower, is the field of machine translation between closely related languages, therefore it seems to be a good testing ground for experiments with hybrid MT methods. There are numerous experiments which have been performed recently for various language groups — for Slavic langauges in Marinov (2003) and Homola and Kuboň (2004), for Scandinavian languages in Dyvik (1995), for Turkic languages in Altintas and Cicekli (2002) and for languages of Spain in Corbi-Bellot *et al.* (2005)

The close relatedness of natural languages from one typological group (and sometimes even across the group borders, cf. Czech-to-Lithuanian experiment described in Hajič *et al.* (2003)) makes the translation task easier thus allowing for the application of methods which would not be good enough for the translation of unrelated language pairs. Using simpler methods does not mean a lower translation quality — many of the translation errors result from the imperfect attempts to parse a source language fully, in some cases even to the deep syntactic level of representation. The accummulation of errors in parsing, transfer and generation in the systems using the classical transfer-based architecture substantially decreases the translation quality.

The stochastic methods, on the other hand, suffer from the lack of parallel data for less frequently translated language pairs. It is therefore no surprise that the results reported by simple systems for the machine translation (MT) of closely related languages (presented for example in Hajič *et al.* (2003) or in Corbi-Bellot *et al.* (2005), with a correct translation of about 90% of the text in both cases) are clearly better than any results published for real MT systems (although the results reported for closely related languages usually do not use the same evaluation metrics as other systems).

1.1 Česílko — a test case for Slavic languages

The main advantage of translating between related languages is the possibility to use much simpler means, in most cases some kind of "shallow" methods, most prominently in parsing or in transfer. The influence of a shallow syntactic parser on the quality of MT for closely related languages can be demonstrated on the example of the system Česílko (Hajič *et al.* (2003)). In the first version of the system, which was able to translate from Czech to Slovak exploiting only a morphological analyzer of Czech accompanied by a stochastic tagger (Hajič and Vidová-Hladká (1998)) and performing the lexical transfer and the translation of morphological tags, we have achieved slightly more than 90% accuracy (measured by a metrics roughly reflecting the complexity of the post-editing task).

The system has been extended later for additional target languages — Polish and Lithuanian. As reported in Hajič *et al.* (2003), the performance of the system for Czech-to-Polish translation using an identical architecture as the original Czech-to-Slovak system reached only 71.4% (due to a much lesser degree of similarity between Czech and Polish compared to Czech and Slovak), while for the Czech-to-Lithuanian translation the result was only slightly lower, 69%. This surprising result (Lithuanian belongs to a different language group, namely to Baltic languages, and its similarity to Czech is lower than the similarity of Polish and Czech) can be attributed to the involvement of a module of simple shallow syntactic analysis of Czech coping with some phenomena which cannot be directly translated from the source language to the target language.

2 A description of the system

Our MT system, originally designed for the Czech-to-Slovak MT, has a very simple architecture. It exploits the close similarity of both languages at all linguistic levels. In particular, no full-fledged analysis is needed. A full syntactic analysis cannot be done with sufficient precision as for now, and the errors we would introduce by trying to create a full syntactic tree for the sentence would lower the quality of the translation significantly (as reported in Oliva (1989) for a Czech-to-Russian MT system). Thus we have adopted the simplistic and rather naïve approach of ignoring syntactic differences and focusing on morphology and lexics. Nevertheless, since Czech is a language with rich inflection, which implies a very high degree of morphological ambiguity, it seemed helpful to integrate a partial ('shallow') parser of Czech into the system. Compared to the system Česílko briefly mentioned in the previous section, the new system does not use the stochastic tagger because due to a relatively low precision (hardly extending 95%) it caused too many errors that had a negative impact on the translation quality.

The main goal of the partial parser is to restrict the ambiguity of morphological annotation by using local context. For example, the soft adjectives inflected according to the pattern *jarní* "spring" have 27 different tags in Czech which would result in many different forms in Slovak (*jarný, jarná, jarné, jarní* etc.) because the target language lacks the high syncretism of soft adjectives. Fortunately, this immense ambiguity can often be constrained if the adjective is followed by a noun that governs it. Due to the agreement, only the intersection of possible tags of both words is valid in the given local context.

The partial analysis is followed by lexical transfer. In this phase, lemmas of all words are translated to their Slovak equivalents according to a bilingual dictionary. This dictionary contains no additional morphological information. Since the partial parser produces complex syntactic structures, they have to be linearized so that we obtain the target sentence. The linearization is a reverse process of parsing which means that the complex structures are decomposed while preserving the original word order in the source language. Finally, all words are morphologically processed (word forms are generated from lemmas and tags).

In the following subsections, we describe in detail the components and data structures they use.

2.1 Feature structures

In the system, the basic data structure for representing linguistic data is a feature structure. It is an attribute-value-matrix (AVM); the values of its attributes are atoms, strings or complex values (sets or embedded feature structures). All feature structures in the system are typed, i.e., there is a global type hierarchy and each feature structure is assigned a type, for instance:

(1) $\begin{bmatrix} \textit{adv} & \\ \text{LEMMA} & \text{`quickly'} \\ \text{POS} & \text{adv} \end{bmatrix}$

Each linguistically significant entity has a set of relevant features. The value of a feature may be underspecified, i.e., its value may not be fully known until a more specific context is given (e.g., the morphological analyzer classifies the word *IBM* as a noun but specifies neither number nor case for it). Ambiguous feature values usually get resolved after having taken the context into account.

The most typical operation on feature structures is unification which is a combination of mutually compatible attribute values. What is often used in the rules is a partial unification, i.e., only specified attributes are unified (e.g. *case, gender, number*), which reflects the linguistic notion of agreement between a head and some of its dependents.

2.2 Chain graphs

The basic data structure that represents text segments and their local contexts is a chain graph. A chain graph is a continuous graph with designated initial and end nodes. It represents all hypotheses that are valid up to a certain point in the parsing process. The application of syntactic rules is implemented by adding new edge to the graph. For example, implementing the rule of an adjective that depends on a following noun and agrees with it in gender, number and case, means finding two adjacent edges in the chain graph (for the adjective and its governing noun, respectively) and adding a new edge that spans the found edges, if both words agree in the required attributes. The original edges are marked as used by a rule which means that they will be removed from the graph after the application of all possible rules. At the end of the parsing process, the remaining edges represent a partial parse of the source segment (the parsing process is described in detail in Colmerauer (1969)).

There are some simple workarounds that allow for a more effective processing of chain graphs like reducing morphological ambiguity by means of shackles, removing of falsified hypotheses etc.

The use of chain graphs bears one specific problem. Since we do not aim to parse whole sentences, the result of the parser usually are not long edges from the initial node to the end node, but rather edges that cover simple noun and prepositional phrases. If such edges overlap and there is no other edge that would cover both of them, there is no path in the graph from the initial graph to the end graph, resulting in an empty graph. This is an inherent property of the formalism, described in more detail in Colmerauer (1969). We do not solve this problem explicitly, suggesting to translate such sentences once more with the shallow parser switched off.

2.3 Rules

2.3.1 The structure of the rules

The grammar for analysis and synthesis consists of declarative rules that prescribe how to combine phrases to complex structures or how to decompose them. In our system, all rules are context-free.

A rule can be applied if its right-hand side matches the categories of a subchain in the chain graph and all conditions associated with the rule are met. The conditions are defined by means of unification over the associated feature structures and/or their attributes (which can be atomic values or recursively embedded features structures). In such a case, a new edge or a subchain of new edges is added to the chain graph which spans the edges that are covered by the right-hand side of the rule. The feature structure the new edge is labelled with is usually based on one of the feature structures of the covered edges and extended by means of unification according to the conditions associated with the rule (an exception may be, for example, a feature structure for coordination).

2.3.2 NP/PP rules

NP/PP rules are used to identify simple noun and prepositional phrases and their internal syntactic structure. A simple but very frequent example may be the combination of an adjective and a noun that are adjacent and agree in gender, case and number.

Rules used for partial ('shallow') analysis do not usually reflect the relationship (mainly dependencies) between the main verb and its complements. Such techniques are used for instance for named entity recognition. Here is an example of a simple NP/PP rule (the agreement in case is expressed by the first equation):[1]

(2) PP → P **NP**, ↑ CASE = ↓ CASE & ↑ PREP = ↓

The rule attaches a preposition to a noun phrase. The first part (before the comma) declares the categories of the subchain the rule will be tentatively applied to. The bold font denotes that the feature structure of the right element will be propagated as the head (the core of the feature structure) of the phrase. It takes a preposition and a noun phrase to the right of it that agree in case which is declared in the other part of the rule — the conditions. Thus the resulting feature structure is the feature structure of the noun phrase extended with a new attribute — *prep* — which is unified with the feature structure of the preposition.

3 Shallow parsing module

In this section, we describe the parser from a more general perspective than it is used in the presented system. The power of the parser component has also been tested in our recent experiments with the Czech-to-German language pair and it turned out that it is also capable of performing deep syntatic analysis (including valence).

The main task of the shallow parser in an MT system is to deliver an information about the sentence structure to the transfer module so that language specific structural properties could be handled and transferred properly. Without the parser, morphological differences may only be considered. That would, of course, be insufficient for most language pairs. Hence the parser may provide an add-on value which is supposed to improve the target sentence. If the source sentence is left untouched by the parser (because it is too short or too complex), the system translates it as if there were no parsing module.

The output of the parser is a set of *c*-trees. It is important to mention that a *c*-tree does not represent the structure of the sentence as such but a concrete rule application sequence. What is passed to the transfer module are *f*-structures that are assigned to constituent phrases during the parsing process.

We would like to underline once again that the shallow Czech grammar is not supposed to parse whole sentences. Of course, if the syntactic structure of the sentence is simple enough, the result will be one tree (or a set of trees) covering the whole sentence. Nevertheless in most cases, the result is a set of trees which

[1]We use the LFG notation (Bresnan (2002)) although the rules are interpreted in a slightly different way (see below).

only represent fragments of the sentence. One reason for such behavior may be non-projectivity which is very frequent in languages with free word order. But projective sentences also may be parsed only partially since the grammar focuses on the level of noun and prepositional phrases. The coverage of verbal phrases is rather small, the rules on this level are meant to capture only the syntactic construction which may cause serious problems in the target sequence.

The formalism is based on a chunk parser. What is very important is the fact that the derivational process is context-free (in the sense of Chomsky's hierarchy) which has the crucial consequence for Slavonic languages that it is not able to handle non-projective constructions (at least not directly).

In the following subsections, we give a brief overview of what should be coped with within the grammar.

3.1 Ambiguous input

The input of the parser can be morphologically ambiguous. In such a case, the parser tries to use all available data to construct a complete tree. If it succeeds, all complete trees create the result set whereas all input items which are not contained in a complete tree are discarded.

It is necessary to parse the whole sentence in order to disambiguate it morphologically. Even then, some words may keep more than one morphological tag (due to case syncretism). In case of shallow parsing only, the morphological ambiguity seems to be one of the most serious problems. The best case scenario would be to get an disambiguated input. Unfortunately, at the moment the only possibility is to use a stochastic tagger which introduces too many errors that make it impossible for the parser to recognize important dependencies. It is a general problem of highly inflected languages that their taggers work with lower precision and at the same time it is impossible to disambiguate the input text morphologically by means of shallow rules only (as shown, e.g., for Czech in Žáčková (2002)).

3.2 Agreement

One of the essential rule principles is the agreement of morphological categories between the governor and its dependent. For example, an adjective which depends on a noun, has to agree with it in gender, case and number. We understand the term *agreement* in broader sence, i.e., a dependent agrees with its governor if a set of conditions which are defined for the particular type of syntactic construcion, is satisfied. In most cases, the conditions are simply equivalences of category values, as in the following phrase:

(3) *mladší* *sestře*
younger-FEM,SG,DAT sister-FEM,SG,DAT
"to the younger sister"

Nevertheless, the conditions may be more complicated sometimes, for instance, in Polish noun phrases if the governor is in dual form[2]:

[2] There are rests of dual in Polish for pairwise nouns.

(4) *czarnymi* *oczyma*
black-NEUT,PL,INS eyes-NEUT,DU,INS
"with black eyes"

(5) *w* *swoim* *ręku*
in REFL-POSS,SG,LOC hands-FEM,DU,LOC
"in his/her hands"

Another example can be found in Russian:[3]

(6) два больших города
two big-MASC,PL,GEN towns-MASC,SG,GEN
"two big towns"

3.3 Implementation of the formalism

Let us make a couple of short notes on the underlying implementation of the formalism. Both feature structures and rules are written in form of (Lisp-like) *s*-expressions that are automatically trans-compiled to LFG-like rules. The implementation has been inspired mostly by LFG (Bresnan (2002)) and the SProUT framework (Becker *et al.* (2002)).

Apart from rules used to build syntactic trees, we use in our grammar some rules the aim of which is to modify the chain graph or to control the parsing process. Since the formalism is declarative, the *control rules* use a workaround to achieve a particular modification of the graph. Let us briefly explain at least the most important one of these rules.

As has been already described above, the input of the parser is typically morphologically highly ambiguous and the main task of the parser is to disambiguate the sentence (or at least to reduce its ambiguity). Let us consider the sentence *Starý hrad stojí na kopci* "There is an old castle on the hill". The phrase *starý hrad* is morphologically ambiguous (nominative and accusative). After having recognized this phrase as the subject of the main verb, we know that the case is nominative in this context. And since there is no other reading where it would be accusative, we want to remove this wrong reading. In fact, it is removed automatically by the algorithm of the parser. But what would happen if we had the bare phrase *staré hrady*? There are two possible readings (nominative and accusative) which cannot be resolved due to lack of context. Nevertheless, there are still other meanings for each of the words independently (unregarding the dependence between them). In this case, the contextually incorrect edges would not be removed although the parser has analyzed the phrase. This is one negative property of the parser framework which has to be solved explicitly. We use a workaround: we insert a dummy edge (*shackle*) between adjacent edges. If there is at least one analysis which connects two words from adjacent edge bunches, the parser marks the shackle as used, i.e., it will be removed by the system later. As a side effect of

[3]Originally, the form of the noun was nominative dual which has been re-analyzed and extended to the numbers 3 and 4 (Trunte (2005)). In today's grammars, the form is considered to be genitive singular (Townsend and Janda (2003)).

this, the ‘wrong’ edges do not belong to a valid path from the initial node to the end node any more and will be deleted, too.

4 Statistical postprocessing and evaluation

An essential part of the whole MT system is the statistic postprocessor. The main problem with our simple MT process described in the previous sections is that both morphological analyzer and transfer introduce a huge number of ambiguities into the translation. It would be very complicated (if possible at all) to resolve this kind of ambiguity by hand-written rules. That is why we have implemented a stochastich post-processor which aims to select one particular sentence that is best in the given context.

We use a simple language model based on trigrams (trained on word forms without any morphological annotation) which is intended to sort out “wrong” target sentences (these include grammatically ill-formed sentences as well as inappropriate lexical mapping). The current model has been trained on a corpus of approx. 18 million words which have been randomly chosen from the Wikipedia for the target language[4].

Let us present an example of how this component of the system works. In the source text we had the following Czech segment (matrix sentence):

(7)	*Společnost*	*ve*	*zprávě*	*uvedla*
	company-FEM,SG,NOM	in	report-FEM,SG,LOC	inform-LPART,FEM,SG

“The company stated in the report, … ”

The rule-based part of the system has generated two target segments: 1) *Spoločnosť vo zpráve uviedli*, 2) *Spoločnosť vo zpráve uviedla*.

The word *uvedla* is ambiguous (fem.sg and neu.pl). According to the language model, the ranker has (correctly) chosen the second sentence as the most probable result.

There are also many homonymic word forms that result in different lemmas in the target languages. For example, the word *pak* means both “then” and “fool-pl.gen”, the word *tři* means “three” and the imperative of “to scrub”, *ženu* means “wife-sg.acc” and “(I’m) hurrying out” etc. The ranker is supposed to sort out the contextually wrong meaning in all these cases.

We have evaluated the system on approx. 300 text segments from technical and news domain. We use smaller text segments than whole sentences, i.e., we translate matrix and embedded sentences separately for higher efficiency (the ranker has less sequences to evaluate). The metrics we are using is the Levenshtein edit distance between the automatic translation and a reference translation. There are three basic possibilities of the outcome of translation of a segment.

1. The rule-based part of the system has generated a ‘perfect’[5] translation (among other hypotheses) and the ranker has chosen this one.

[4] http://sk.wikipedia.org, http://ru.wikipedia.org

[5] By ‘perfect’ we mean that the result does not need any human post-processing.

2. The rule-based part of the system has generated a ‘perfect’ translation but the ranker has chosen a different one.
3. All translations generated by the rule-based part of the system need post-processing.

In the first case, the edit distance is zero, resulting in accuracy equal to 1. In the second case, the accuracy is $1 - d$ with d meaning the edit distance between the segment chosen by the ranker and the correct translation divided by the length of the segment. In the third case, the accuracy is calculated as for (2) except that we use the reference translation to obtain the edit distance.

Given the accuracies for all sentences, we use the arithmetic mean as the translation accuracy of the whole text. The accuracy is negatively influenced by several aspects. If a word is not known to the morphological analyzer, it does not get any morphological information which means that it is practically unusable in the parser. Another possible problem is that a lemma is not found in the dictionary. In such a case, the original source form appears in the translation, which penalizes the score of course. Finally, sometimes the morphological synthesis component is not able to generate the proper word form in the target language (due to partial incompatibility of tagsets for both languages). In such a case, the target lemma appers in the translation.

In our test data, approx. 35% of segments have been translated perfectly. For approx. 12% of segments, the system has generated a perfect translation but the ranker has chosen a different one. Approx. 20% of the segments could not be parsed properly because there was no path consisting of unused edges from the initial node to the end node. For these segments, we have considered as the result all paths with the lowest number of used edges.

In general, the accuracy of the translation into Slovak is 96.45%. With the original architecture (i.e., using a tagger only), the accuracy is 93.92%. When we left out the parser, the result was 96.10%. For Russian, the results are 74.67%, 69.87% and 72.85%, respectively.

5 Conlusions

Although the hybrid MT technique presented in the paper has not been tested by means of standard metrics, the results achieved for the language pair Czech-Slovak encourage further experiments for other language pairs. The experiment with Russian has been only a first step in this direction, the results of the Czech-to-Russian experiment indicate that for less closely related languages it might be necessary to add more modules into the system.

The second interesting result has been obtained in the experiment measuring the individual influence of the stochastic ranker and a shallow parser on the overall quality of the translation. Although the ranker is responsible for a higher increase of translation quality, the positive role of the shallow parser of Czech is also apparent. Also in this case it will be a matter of further research to find out whether other language pairs will be in accordance with the results presented in this paper.

Acknowledgments

The research presented in this paper has been supported by the grant No. 1ET100-300517 of the GAAV ČR.

References

Kemal ALTINTAS and Ilyas CICEKLI (2002), A Machine Translation System between a Pair of Closely Related Languages, in *Proceedings of the 17th International Symposium on Computer and Information Sciences (ISCIS 2002)*, pp. 192–196, Orlando, Florida.

M. BECKER, W. DROŻDŻYŃSKI, H.U. KRIEGER, J. PISKORSKI, U. SCHAEFER, and F. XU (2002), SProUT — Shallow Processing with Typed Feature Structures and Unification, *Proceedings of ICON 2002*.

Joan BRESNAN (2002), *Lexical-functional syntax*, New York.

Alain COLMERAUER (1969), Les systèmes Q ou un formalisme pour analyser et synthétiser des phrases sur ordinateur, Technical report, Mimeo, Montréal.

Antonio CORBI-BELLOT, Mikel FORCADA, Sergio PRTIZ-ROJAS, Juan Antonie PEREZ/ORTIZ, Gema REMIREZ-SANCHEZ, Felipe Sanchez MARTINEZ, Inaki ALEGRIA, Aingeru MAYOR, and Kepa SARASOLA (2005), An Open-Source Shallow-Transfer Machine Translation Engine for the Romance Languages of Spain, in *Proceedings of the 10th Conference of the European Association for Machine Translation*, Budapest.

Helge DYVIK (1995), Exploiting Structural Similarities in Machine Translation, *Computers and Humanities*, 28:225–245.

Jan HAJIČ and Barbora VIDOVÁ-HLADKÁ (1998), Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset, in *Proceedings of the Conference COLING - ACL '98*, Montreal, Canada.

J. HAJIČ, P. HOMOLA, and V. KUBOŇ (2003), The MT system Česílko, *In: Proceedings of the VIII MT Summit, New Orleans.*

Petr HOMOLA and Vladislav KUBOŇ (2004), A translation model for languages of acceding countries, in *Proceedings of the IX EAMT Workshop*, University of Malta, La Valetta.

Svetoslav MARINOV (2003), Structural Similarities in MT: A Bulgarian-Polish case, http://www.gslt.hum.gu.se/ svet/courses/mt/termp.pdf.

Karel OLIVA (1989), A parser for Czech implemented in Systems Q, Technical report, MFF UK, Prague.

Charles E. TOWNSEND and Laura A. JANDA (2003), *Gemeinslavisch und Slavisch im Vergleich. Einführung in die Entwicklung von Phonologie und Flexion*, Verlag Otto Sagner, München.

Nikolaos H. TRUNTE (2005), *Altkirchenslavisch*, volume 1 of Словенскъи языкъ. *Ein praktisches Lehrbuch des Kirchenslavischen in 30 Lektionen. Zugleich eine Einführung in die slavische Philologie*, Verlag Otto Sagner, München.

Vincent VANDEGHINSTE, Ineke SCHUURMAN, Michael CARL, Stella MARKANTONATOU, and Toni BADIA (2006), METIS-II: Machine Translation for Low Resource Languages, in *Proceedings of the 5th international conference on Languages Resources and Evaluation (LREC)*, Genoa, Italy.

Eva ŽÁČKOVÁ (2002), *Parciální syntaktická analýza (češtiny)*, Ph.D. thesis, Fakulta informatiky Masarykovy univerzity, Brno.