Gábor Prószéky

Translating while Parsing

Abstract

Translations—unconsciously—rely on the hypothesis that sentences of a certain source language L_S can be expressed by sentences of the target language L_T. Formal theories of semantics, on the other hand, assume that semantics is computed compositionally: the meaning of an expression is a function of the meaning of its parts. A standard argument for the compositionality of meaning is productivity. Formal semantic compositionality has been formalized by the rule-to-rule hypothesis of Bach (1976). The idea is well-known from the Montague grammars. The meaning of idioms is, however, not a function of its parts: its meaning has to be learnt as a unit. Combining the basic idea of translation with the rule-to-rule hypothesis, we can express structures of language L_S in language L_T. These structures of L_T—called translations—can also be either computed compositionally or as units: each syntactic rule of L_S should have its counterpart expressed—not by semantic rules, but—by syntactic descriptions in language L_T.

1. Toward a new approach to machine translation

The meaning of a complex linguistic structure is wholly determined by its sub-structures and the meanings of them. Semantic compositionality was formalized by the rule-to-rule hypothesis of Bach (1976): it says that a tight correspondence is imposed between syntax and semantics such that every rule of syntax is also a rule of semantics. In a machine translation system (Rosetta—described by Landsbergen 1985) we can meet a rather direct application of the compositionality principle adopted from the Montague Grammars (Thomason 1974):

> The meaning of an expression is a function of the meaning of its parts and the way in which they are syntactically combined. (…) it is an important criterion of a correct translation that it is meaning-preserving, this seems to be a useful guideline in machine translation. (Appelo & al. 1987)

The rule-to-rule hypothesis really seems a useful working assumption in machine translation, but in a different way. We can claim the following: if a structure can be described syntactically in the language L_S, it can also be described by structures of another language L_T. Of course, human languages are not as exactly formalized as formal languages, so ambiguous source language sentences are permitted to produce more than one target language description. If a source structure is underspecified, target structures are minimally as underspecified as the source was. Technically, an extra layer should be introduced to a phrase structure, which we call the translation of the actual structure. So a compositional approach to translation will have a representation of the contribution of each word and sub-phrase towards the translation of the whole.

2. Pattern-based description of translation

As is well-known, three rule-based approaches to MT are traditionally distinguished: direct, interlingua~~l~~ and transfer. The direct method uses a primitive one-stage process in which words in the source language are replaced with words in the target language and then some rearrangement is done. The main idea behind the interlingua method is that the analysis of any source language should result in a language-independent representation. The target language is then generated from that language-neutral representation. The transfer method first parses the sentence of the source language. It then applies rules that map the lexical and grammatical segments of the source sentence to a representation in the target language. The three methods can be expressed as below:

Direct: $MT_{Direct}(S_S) = S_T$

Interlingua: $MT_{Interlingua}(S_S) = Generate(Analyze(S_S)) = S_T$

Transfer: $MT_{Transfer}(S_S) = Generate(Transfer(Analyze(S_S))) = S_T$

where S_S is the source sentence to be translated, S_T is the translation of the source sentence in question, that is, the target sentence.

In the eighties EBMT, i.e., example-based machine translation was suggested i.e., by Nagao (1984) as a better approach to machine translation than RBMT, i.e., rule-based machine translation. The goal of our research was to find an optimum between EBMT and RBMT in terms of practical applicability: translation quality and speed. EBMT is generally considered a statistics-based, probabilistic process, whereas RBMT is often thought of as a fixed, traditional, deterministic approach. In contrast, we believe that EBMT and RBMT are just the two extremes of a generalized model. In our model, there are many possible transitions between the two. Not all of our "examples" have been directly extracted from corpora, or produced by statistical analysis. Rather we opted to build a database of structural segments, which have been generated from various dictionaries of idioms and collocations. Besides multi-word lexemes we had to add some single-element lexical entries to the collection, as well. However, in many cases, their linguistic behavior has been described by expanding them to multi-word units by hand.

In the MetaMorpho machine translation project—led by László Tihanyi—we describe both what are traditionally called rewriting rules and lexical entries in the form of patterns. Currently we have more than 200,000 patterns, the majority of which are lexicalized items. The system uses no separate dictionary: what would traditionally be a lexical entry is integrated in the form of patterns. A simple lexical pattern is like `NX[N.lex, N.num] = N(lex='birthday')`, a bit more underspecified grammatical pattern is like `VP[TV.conj] = TV(lex='read', pass=NO) + DOBJ`, but a pattern can also be 'productive,' which means it contains little or no lexical information. Such a pattern would be, for instance, `VP=TV(vti=VT)+DOBJ`, which describes the fact that a transitive verb and an object can form a verbal phrase. Such patterns are traditionally called rules. The patterns in the grammar are much more complex than the ones shown. The grammar itself operates with pairs of patterns that consist of one source pattern used during bottom-up parsing and one or more target patterns that are applied during top-down generation. The below examples show two English patterns with their 'on-the-fly' Hungarian translations.

A multi-word lexical unit, *approach to*, is treated as below:

```
*NX=approach+to:0401311411
EN.NX[ct=COUNT] = N(lex="approach") + PPOBJ(lex="to")
HU.NX = PPOBJ[case=GEN] + N[lex="megközelítés"]
```

The treatment of the English interrogative subordinate clause with its translation looks like this:

```
*S=SUBX+SPU:0206251536-6
EN.S[SPU.shlf, SPU.shulex, SPU.sonlyowner,
SPU.msonlyowner,     SPU.mshulex, SPU.mshlf, SPU.slex,
SPU.word,   tsub=WH,  SPU.sval, sublex=SUBX.lex] =
SUBX(decide=YES) +    SPU(missing=NTG, word=DIR,
imper!=YES)
HU.S(subtype=OBJ) = SUBX[subtype=OBJ,
case=EN.SPU.mscase, postp=EN.SPU.mpostp] +
SPU[S.subtype]
HU.S(subtype=COMPL) = SPU[Subx<-EN.SUBX, subxfl=YES]
```

Now we can see how these patterns work in practice:

EN:	*This is a really nice approach to mathematics.*							
HU:	*Ez*	*a*	*matematika*	*egy*	*igazán*	*jó*	*megközelítése.*	
literally:	*this*	*the*	*mathematics*	*a*	*really*	*nice*	*approach+of*	

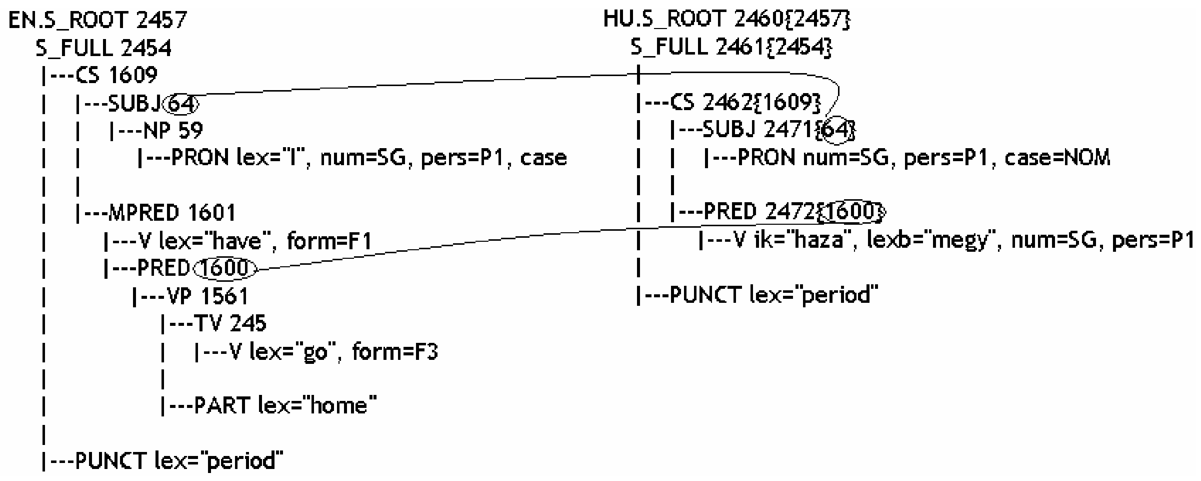
EN:	*I don't know if he came.*			
HU:	*Nem*	*tudom,*	*hogy*	*jött-e.*
literally:	*no*	*know+I*	*that*	*came-if*

Every terminal and non-terminal symbol (or what is equivalent, the corresponding node in the syntactic tree under construction) has a well-defined set of features. The number of features varies between zero and a few dozen, depending on the category. These features can either take their values from a finite set of symbolic items (e.g., values of `case` can be INS, ACC, DAT, etc.), or represent a string (e.g., `lex="approach"`, that is, the lexical form of a token). The formalism does not allow for embedded feature structures. It is important to note that no structural, semantic or lexical information is amassed in the features of symbols: the interpretation of the input is contained in the syntactic tree itself, and not in the features of the node on the topmost level. More specific patterns (e.g. *approach to*) can override the general ones (e.g. *approach*), meaning that all subtrees containing symbols that were created by the general pattern are deleted. Every symbol that is created and is not eliminated by an overriding pattern is retained even if it does not form part of a correct sentence's syntactic tree. Each pattern can state any number of overrides on other rules: if the overriding rule fires over a specific range of the input, it blocks the overwritten one over the same range. Not only is this extremely useful for debugging purposes, but it allows for 'best guesses' when no interpretation for a whole sentence is found in real-life applications.

3. Translation via immediate transfer

The analysis of the input is performed in three basic steps. First the sentence to be translated is segmented into terminal symbols or tokens. This token sequence is the actual input of the parser. The morphological analyzer determines all the needed morpho-syntactic attributes of these symbols. We use the Humor morphological system (Prószéky & Kis 1999) that is based on surface patterns. The basic strategy of Humor is inherently suited to parallel execution: search in the main dictionary, secondary dictionaries and affix dictionaries can be performed in a parallel way. In the case of agglutinative languages like Hungarian, where the number of inflected word-forms for a single word is well over hundreds, a reliable morphological generator is a crucial part of any translation tool. The advantage of Humor is that it could be used as a generator as well as an analyzer. The system accepts unknown elements: they are treated as strings to be inflected at the target side. Our syntactic parser called Moose (Prószéky, Tihanyi & Ugray 2004) analyzes this input sequence and if it is recognized as a correct sentence, comes up with one or more root symbols on the source side. When the whole input is processed and no applicable patterns remain, the target equivalent is read top-down from the root symbols by firing the target pattern corresponding to the source pattern that created the edge at parse time. This solution we can call "immediate transfer" as it uses no separate transfer steps or target transformations.

Pattern pairs can have conditions in the left-hand side, and in the case of multiple target patterns, the first one whose conditions are satisfied is fired. The right-hand side of the source pattern can state conditions for any of its symbols' values. To handle more complicated word-order changes, however, a stronger means of rearrangement is also provided: interpretation of the source structure in the target structure may need rearrangement of its elements within the scope of a single node and its children. Three subtree interpretations are allowed: (i) permutation of the node's children, (ii) deletion of one or more children from the target tree, and (iii) insertion of terminal symbols. The next example shows MetaMorpho's translation trees for the sentence *I have gone home*. Its translation is *Hazamentem*. Numbers in curly brackets show the number of the source pattern belonging to the Hungarian translation.



```
EN.S_ROOT 2457                                          HU.S_ROOT 2460{2457}
  S_FULL 2454                                             S_FULL 2461{2454}
   |---CS 1609                                            |
   |   |---SUBJ(64)                                       |---CS 2462{1609}
   |   |   |---NP 59                                      |   |---SUBJ 2471{64}
   |   |       |---PRON lex="I", num=SG, pers=P1, case    |   |   |---PRON num=SG, pers=P1, case=NOM
   |   |                                                  |   |
   |   |---MPRED 1601                                     |   |---PRED 2472{1600}
   |       |---V lex="have", form=F1                      |       |---V ik="haza", lexb="megy", num=SG, pers=P1
   |       |---PRED(1600)                                 |
   |           |---VP 1561                                |---PUNCT lex="period"
   |               |---TV 245
   |               |   |---V lex="go", form=F3
   |               |
   |               |---PART lex="home"
   |
   |---PUNCT lex="period"
```

Figure 1.

There are various differences in the source and the target structure: the Hungarian tense system is essentially simpler than English, but compounding is very productive and non-third person subject of sentences are not explicitly given in most cases. Thus, *I* is translated as a verbal suffix, the present perfect structure expressed by *have* plus the verb's third form becomes simple past in Hungarian and *go home* is expressed by a single word in Hungarian: *hazamegy*. MetaMorpho's full parsing of this English sentence needed 2,458 steps, the synthesis of the Hungarian output was made in 26 steps. This big difference between the numbers of steps in analysis and generation is of general importance: it illustrates that the output is perfectly given when the parse is over, so the only operation to be done is simplification of the target description of the root element of the analysis.

A subtree can be memorized in a feature when a unification operation takes places at parse time, and because this feature's value can percolate up the parse-tree and down the target tree, just like any other feature, a phrase swallowed at any level in the source side can be expanded at a completely different location in the target tree. The power and simplicity of subtree memorization and random insertion can be demonstrated in the following example with the translation of English possessive structures into Hungarian: *the sixtieth birthday of Fred* translates into *0*-DET/*the Fred*-N/*Fred hatvanadik*-NUM/*sixtieth születésnapja*-N+POS/*birthday*, that is, the order of the two nominal phrases is reversed and the determiner absorbed by the proper noun: *Fred hatvanadik születésnapja*.

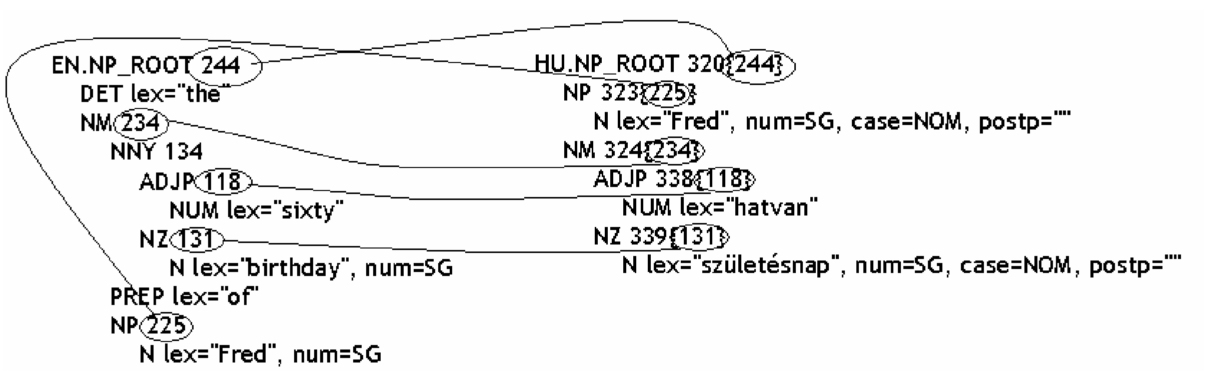


Figure 2.

Through the interplay of only two patterns (the place of memorization at N-bar level and insertion at NP-level), a possessive structure of any length is translated recursively in reverse order into Hungarian.

As a consequence, we can say that MetaMorpho's translation method is opposed to the traditional transfer approach in that there is no need to "transfer" an abstract structure at any level: we create our analysis with the final output in mind, and can produce the result in a very straightforward manner, without any need for complex independent transfer methods following syntactic analysis. However, MetaMorpho does not use interlingual representations, and it would be misleading to claim that it belongs to direct translation systems. Unlike the first primitive machine translation systems, MetaMorpho uses systematic grammatical descriptions and a mechanism that is a special variant of the rule-to-rule idea of Bach: the target equivalent of a MetaMorpho source structure is its translation and not the formal logical representation of its semantics. Summarizing the above, MetaMorpho seems to belong to a fourth machine translation paradigm. It~~s~~ shows, of course, some relation to the Rosetta machine translation system (Landsbergen 1984) which uses logical semantic representations. Rosetta really used the rule-to-rule hypothesis, but that representation was considered as an interlingua which differs basically from the MetaMorpho approach. Therefore, we introduce a fourth approach to machine translation for MetaMorpho, as it is represented here:

MetaMorpho: $MT_{MetaMorpho}(S_S) = ReadOut(Analyze(S_S)) = S_T$,

where S_S is the source sentence to be translated, S_T is the translation of the source sentence in question, that is, the target sentence, and the two functions are the bottom-up parsing (Analyze) that produces a second structure besides the parse-tree, and the ReadOut function interprets it in a top-down manner, as shown by the examples above.

4. Recent and future implementation issues

The principles and modules discussed above are used in real-life applications translating texts from English into Hungarian. MetaMorpho's knowledge consists of lexical patterns and core patterns that serve as generalized examples for the more specific ones. The number of core type patterns is in the magnitude of thousands for a language like English. The number of lexical patterns is, however, well in the hundreds of thousands. They have mainly been derived from existing lexicons and collocation databases.

The motivation for creating our robust bottom-up parser (Prószéky, Tihanyi & Ugray 2004) is that the grammar's applications invariably require access to a parse's partial results in the absence of a full parse tree. The parser is able to cope with sentences that have no full parses: it accesses all parse trees and can select a disjunctive coverage of the input tokens. Another new feature of MetaMorpho is its grammar writer's workbench called Rule Builder. This allows the controlled addition of new, lexical or even syntactic patterns into the grammar. With the help of this, the user can add and modify the rules of the grammar on-line without the need to recompile the rest, using an SQL database for user added entries. The technology used in RuleBuilder can also be applied to work out a special combination of the MetaMorpho machine translation tool and translation memories (Hodász, Grőbler & Kis 2004). Taking advantage of MetaMorpho's above mentioned ability to translate incomplete sentences, we could translate this differing part of the sentence and thus improve the efficiency of translation memories. MetaMorpho currently fetches only the first target equivalent from the lexical patterns. This could be changed by reordering the target equivalents according to the actual context. A word-sense disambiguation module providing semantically disambiguated output is under development. We are also working on a topic recognition module running the same way as language identifier programs do but identifying the sublanguage (business, medicine, sport, etc.), having a well recognizable terminology within a single language.

5. Conclusion

MetaMorpho applies Bach's rule-to-rule hypothesis for translation purposes. The system relies on a uniform description of lexical and structural information called patterns: they are basic tools for describing both standard and idiomatic behavior of sentences, clauses and phrases. If a pattern is short and fully specified, it is a lexical entry in the traditional terminology; if it is longer, but fully specified, it is an idiom, or a specific example. If no attributes of a pattern are specified, then the pattern is conventionally a rule. Our approach puts the emphasis on the transitions between the two: idioms and collocations are elements that are filled in, but which are not fully specified. The key issues of our model are how to manage these generalized patterns. MetaMorpho patterns show certain similarities to the "translationally equivalent patterns" used in the English-Japanese translation system of Kawasaki & al. (1992). The knowledge base in their model consists of patterns mainly utilized to translate idiomatic or nonstandard expressions.

MetaMorpho represents a generalization of the EBMT model. However, the parser is not statistical, and it combines source language analysis and target language interpretation in one single task. If the input is grammatically correct, the system should provide correct translation, and if the input cannot be analyzed, the system should provide the translation of all the separate correct structures it can identify. MetaMorpho, however, uses neither interlingua representations, nor transfer steps: both structural and lexical transfers are already done while parsing. It would, however, be misleading to say that our approach belongs to the paradigm of direct translation, just because it is neither interlingual, nor transfer-based. Unlike the first primitive machine translation systems, MetaMorpho uses systematic and complex grammatical descriptions with a mechanism that is close to the rule-to-rule hypothesis of Bach. MetaMorpho shows some relation to machine translation systems which use logical semantic representations (e.g. Rosetta), but the original form of the rule-to-rule hypothesis in those systems was, however, used as an interlingua which essentially differs from the MetaMorpho approach. Since MetaMorpho seems to belong to another, new machine translation paradigm.

Acknowledgement

The author is grateful to the machine translation division of MorphoLogic, but first of all to László Tihanyi who has lead the developments from the beginnings.

References

Appelo, Lisette, Carol Fellinger & Jan Landsbergen (1987) Subgrammars, rule classes and control in the Rosetta translation system. In Bente Maegaard (ed.) *Third Conference of the European Chapter of the Association for Computational Linguistics. Proceedings of the Conference, 1-3 April 1987, University of Copenhagen, Denmark*, pp. 118–133. Copenhagen: Association for Computational Linguistics.

Bach, Emmon (1976) An extension of classical transformational grammar. In Saenz (ed.) *Problems of Linguistic Metatheory: Proceedings of the 1976 Conference*, pp. 183–224. East Lansing, MI: Michigan State University.

Hodász, Gábor, Tamás Grőbler & Balázs Kis (2004) Translation memory as a robust example-based translation system. In Hutchins (ed.), pp. 82–89.

Hutchins, John (ed.) *Broadening horizons of machine translation and its applications. Proceedings of the 9th EAMT Workshop, 26-27 April 2004*. La Valletta: Foundation for International Studies.

Kawasaki, Zenshiro, Fumiyuki Yamano & Niriyuki Yamasaki (1992) Translator knowledge base for machine translation systems. *Machine Translation* 6.4: 265–278

Landsbergen, Jan (1984) Isomorphic Grammars and their use in the Rosetta Translation system. In Margaret King (ed.) *Machine Translation Today: The state of art*, pp. 351–372. Edinburgh: Edinburgh University Press.

Nagao, Makoto (1984) A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. In Alick Elithorn & Ranan Banerji (eds.) *Artificial and Human Intelligence: Edited review papers presented at the International NATO Symposium on Artificial and Human Intelligence sponsored by the Special Programme Panel held in Lyon, France October, 1981*, pp. 173–180. Amsterdam & New York, NY: North-Holland.

Prószéky, Gábor & Balázs Kis (1999) Agglutinative and other (highly) inflectional languages. In Robert Dale & Kenneth W. Church (eds.) *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp. 261–268. Morristown, NJ: Association for Computational Linguistics.

Prószéky, Gábor, László Tihanyi & Gábor Ugray (2004) Moose: A robust high-performance parser and generator. In Hutchins (ed.), pp. 138–142.

Thomason, Richmond H. (ed.) (1974) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven, CT & London: Yale University Press.

Contact information:

Gábor Prószéky
MorphoLogic
Orbánhegyi út 5,
Budapest 1126, Hungary
proszeky(at)morphologic(dot)hu
www.morphologic.hu