Large-Scale Hybrid Interlingual Translation in GF: a Project Description

Aarne Ranta, Krasimir Angelov, Prasanth Kolachina, Inari Listenmaa

Department of Computer Science and Engineering, University of Gothenburg

Abstract

This paper describes an on-going project on machine translation addressing, at the moment, all language pairs for 11 languages: Bulgarian, Chinese, Dutch, English, Finnish, French, German, Hindi, Italian, Spanish, and Swedish. The translator is based on GF (Grammatical Framework), enhanced with a statistical model learned from a treebank. It runs on multiple platforms, including a web service and a mobile off-line Android application. These systems are open-domain browsing-quality translators, which supports high-quality domain adaptation via embedded controlled languages. The code is open-source free software.

1. Introduction

Interlingual translation is an old idea that has been suggested numerous times and refuted almost as many times. A typical criticism is that the very idea is utopistic: that one can never build an interlingua that faithfully **represents meaning** in all languages of the world. However, as the focus in machine translation has shifted from the perfect rendering of meaning to less modest goals, the idea of an interlingua can be reconsidered.

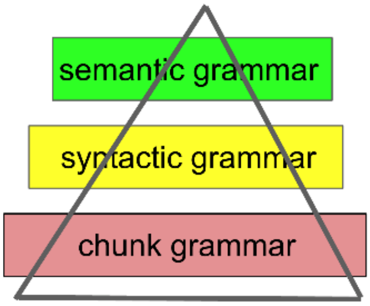
In most of the past efforts, the interlingua was thought as a precise language-independent meaning representation (Hutchins, 1986). In our view, it is this ideal rather than the interlingua idea itself that has made it impossible to build interlingual systems that scale up beyond toy examples. Therefore, if we follow the current main stream and stay content with browsing quality, another characteristic of interlinguas stands out: the **low cost** of interlingual translation systems. The interlingua is then something weaker than a precise meaning representation. Technically, it is just any formal structure that abstracts away from words, morphology, agreement, and word order, and serves as a skeleton of a content we want the translation to preserve.

An interlingual system covering all pairs of n languages only needs $2n$+1 modules, instead of $n(n$-1) as in pairwise systems: one module for the interlingua, plus mappings from each language to the interlingua and back. If we use reversible mappings, we can further reduce the number to n+1. Thus in our current system of 11 languages, we need 12 components, as opposed to 110 in a pairwise system. The grammar components and everything else needed to run the translator (parsing, generation, statistical disambiguation) fit in a binary file of 24 MB, whereas, for instance, the off-line version of Google translate takes 200 MB *per language pair*. The interlingual translator scales up to a large number of languages, and the use of grammar rules compresses the information stored for each language. The source code for the system is, for each language, around 3000 lines for the grammar rules and 20–60 thousand lines for the lexicon (one line per lemma).

Interlingua can also decrease the need of data. The interlingua in our system is an **abstract syntax** in the sense of GF (Ranta, 2011). This architecture is similar to Rosetta (Rosetta, 1994), where the interlingua is Montague's analysis trees (Montague, 1974). All languages in a system share the abstract syntax trees, but they can also share meta-information associated with the trees. For instance, our current system performs disambiguation by using tree probabilities estimated from the Penn Treebank (Marcus et al., 1993), which we have converted into GF abstract syntax (Angelov, 2011). This model is of course based on English data. But as long as we don't have corresponding data for other languages, it gives a model for those as well, which is much better than nothing.

The light weight of the system makes it easily **programmable**. The compilation of the whole system from source takes around CPU 40 minutes (10 minutes on a 4-core processor), as opposed to the several days of training often needed for statistical systems. The rule-based architecture makes it possible to fix individual bugs, such as grammar errors in a particular language. Fixing a bug in one language fixes it in 20 translation directions. As the system enjoys separate compilation of modules, updating the whole system with a bug fix in one language takes just a few minutes of compilation time.

The GF Translator is not only meant as yet another browsing-quality system on the market. GF was originally designed for high-quality systems on specific domains. The novelty in the current project is that we can combine *both* coverage and quality in one and the same system. From the point of view of domain-specific applications, this means that the system doesn't just fail with out-of-grammar input as before, but offers **robustness**. From the open-domain point of view, the system offers a clear recipe for quality improvements by **domain adaptation**. In another view, the system we have built incorporates three levels of the Vauquois triangle in one and the same system: semantic, syntactic, and chunk-based translation, each of which — and not just the highest level — is based on its own part of the interlingua:



2. How it works

2.1 The translator architecture

The GF translator pipeline has three main phases:

1. **Parsing** converts the source into a forest of **AST**s, **Abstract Syntax Trees**, i.e. interlingual representations.

2. **Disambiguation** selects the most probable AST.

3. **Linearization** converts the AST into the target language.

Disambiguation is for efficiency reasons integrated in the parser, which enumerates the results lazily in order of decreasing probability (Angelov and Ljunglöf, 2014). Unlike for most k-best parsers, there is no upper limit on how many results can be obtained.

Translation is performed by the following components:

1. A **PGF** grammar (**Portable Grammar Format**, Angelov et al. (2009)) a binary file consisting of an **abstract syntax** (defining the ASTs) and, for each language, a **concrete syntax** that defines the linearization and (by reversibility) parsing for the language.

2. A **probability model** for disambiguation.

3. The **PGF interpreter**, with generic **parser** and **linearizer** as well as **disambiguator**.

The PGF interpreter is generic, so that the PGF grammar and the probability model can be changed to produce new translation systems. Thus the success of the GF translator does not depend on a specific interlingua, but GF is a framework in which different interlinguas can be built.

2.2 The wide-coverage grammar

Traditional GF translation systems have small, domain-specific interlinguas and grammars. In this paper, we assume one large-scale generic grammar based on the **GF Resource Grammar Library** (RGL, Ranta (2009)). The complete grammar has the following components:

1. **RGL**, defining morphology and most of the syntax.

2. **Syntax extensions**, about 10% addition to RGL.

3. **Dictionary**, mapping abstract word senses to concrete words by using open resources such as linked wordnets and wiktionaries (Virk et al., 2014); morphology mostly by the RGL's "smart paradigms" (Détrez and Ranta, 2012). Abstract dictionary entries are presented as English words split into distinct **senses**. For instance, Swedish requires splitting the noun *time* into senses that linearize to *tid* and *gång*. Splitting senses is on-going work guided by translation needs rather than predefined semantics. It has shown nice convergence properties: the French translations of *time* as *temps* or *fois* result from the same split as had already been performed for Swedish. Splitting is a part of the manual checking of automatically generated dictionaries. The 11 dictionaries range in size from 16k to 66k lemmas, with the average of 25k, of which typically 2k to 5k have already been checked.

4. **Chunk grammar**, to make the translation robust for input that doesn't parse as complete sentences. It is inspired by Apertium (Forcada et al., 2011), which is a rule-based system operating only chunks rather than complete syntactic analyses. In GF, it is derived from the RGL by enabling sub-sentential categories as start categories. The result can contain local agreement and reordering: for instance, if French *dans la maison bleue* is recognized as an adverbial chunk, it can be linearized to German as *im blauen Haus*.

5. **Probabilities**, estimated from the Penn Treebank.

6. **CNL** (Controlled Natural Language), an optional part enabling domain adaptation via Embedded CNLs (Ranta, 2014). If something is parsable in the CNL, the CNL translation is given priority. CNL phrases can also appear as chunks in robust translation. The current demos use the MOLTO phrasebook (Ranta et al., 2011).

Adding a new language is a matter of a couple of days of work, provided that (1) the language has an RGL; (2) mappings to English words are available, e.g. a Wiktionary; (3) the work is done by a person who knows both GF and the target language well. We expect at least a few more of the existing 29 RGL languages to be soon available for large-scale translation.

3. First results

Wide coverage GF translation was made possible by the statistical parser in (Angelov, 2011). Building dictionaries started in 2013 (Angelov, 2014; Virk et al., 2014). Chunk-based robustness and the embedded CNL idea are from spring 2014. The first full-scale system was built as a web service[1] and as an Android application (Angelov et al., 2014). Both interfaces give additional feedback to the users: they show in colours whether the translation comes from the semantic CNL (green), the surface-syntactic RGL (yellow), or from chunks (red). The user can also see syntax trees and alternative translations of ambiguous input.

Evaluation of translation quality is work in progress. In the MOLTO project evaluations (Rautio and Koponen, 2013), BLEU and TER scores were calculated on the basis of human post-edited corrections, and were clearly above general-purpose systems (such as Google), when CNLs were considered. Translation outside CNLs is expected to be below the state of the art for many language pairs (e.g. English to Swedish), but competitive for lower-resourced unrelated languages (e.g. Bulgarian to Finnish).

4. Open questions

The probabilistic model based on individual syntax constructors is basically context-free and hence insufficient, in particular for word sense disambiguation. More sophisticated models introducing conditional probabilities to abstract syntax trees are work in progress.

In GF, translation is **compositional**, because the linearization rules operate on the ASTs in a subtree by subtree fashion. This is a serious problem for translation in general, because the syntactic structure must often be changed, e.g. when converting Swedish *jag heter NN* to English *my name is NN*. In a CNL setting, the problem can be avoided by using sufficiently abstract ASTs, for instance, a two-place "naming" predicate for the mentioned example. One direction in the on-going GF translator work is to introduce such abstractions in large-scale grammars as well, in a way similar to **multiword expressions** and **constructions** (Enache et al., 2014).

[1] http://cloud.grammaticalframework.org/wc.html

References

Krasimir Angelov and Peter Ljunglöf. 2014. Fast statistical parsing with parallel multiple context-free grammars. *European Chapter of the Association for Computational Linguistics, Gothenburg*.

Krasimir Angelov, Björn Bringert, and Aarne Ranta. 2009. PGF: A Portable Run-Time Format for Type-Theoretical Grammars. *Journal of Logic, Language and Information*.

Krasimir Angelov, Björn Bringert, and Aarne Ranta. 2014. Speech-enabled hybrid multilingual translation for mobile devices. *EACL 2014*, page 41.

Krasimir Angelov. 2011. *The Mechanics of the Grammatical Framework*. Ph.D. thesis, Chalmers University of Technology.

Krasimir Angelov. 2014. Bootstrapping open-source english-bulgarian computational dictionary. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), Reykjavik, Iceland, May 26-31, 2014.*, pages 1018–1023.

Grégoire Détrez and Aarne Ranta. 2012. Smart paradigms and the predictability and complexity of inflectional morphology. In *EACL (European Association for Computational Linguistics)*, Avignon, April. Association for Computational Linguistics.

Ramona Enache, Inari Listenmaa, and Prasanth Kolachina. 2014. Handling non-compositionality in multilingual CNLs. In *Controlled Natural Language - 4th International Workshop, CNL 2014, Galway, Ireland, August 20-22, 2014. Proceedings*.

Mikel L Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M Tyers. 2011. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.

W. J. Hutchins. 1986. *Machine translation: past, present, future*. Ellis Horwood Chichester.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330.

R. Montague. 1974. *Formal Philosophy*. Yale University Press, New Haven. Collected papers edited by Richmond Thomason.

Aarne Ranta, Ramona Enache, and Grégoire Détrez. 2011. Controlled Language for Everyday Use: the MOLTO Phrasebook. *Proceeding of CNL 2010, Zurich*.

A. Ranta. 2009. The GF Resource Grammar Library. *Linguistics in Language Technology*, 2. http://elanguage.net/journals/index.php/lilt/article/viewFile/214/158.

Aarne Ranta. 2011. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford. ISBN-10: 1-57586-626-9 (Paper), 1-57586-627-7 (Cloth).

Aarne Ranta. 2014. Embedded controlled languages. In *Controlled Natural Language - 4th International Workshop, CNL 2014, Galway, Ireland, August 20-22, 2014. Proceedings*.

Jussi Rautio and Maarit Koponen. 2013. Deliverable 9.2: Molto evaluation and assessment report.

M. T. Rosetta. 1994. *Compositional Translation*. Kluwer, Dordrecht.

Shafqat Mumtaz Virk, KVS Prasad, Aarne Ranta, and Krasimir Angelov. 2014. Developing an interlingual translation lexicon using wordnets and grammatical framework. *COLING 2014*, page 55.