General Framework for Multilingual Text Processing

Krasimir Angelov

1 Purpose and aims

Computational linguistics is still oscillating between two paradigms. From one side the natural language is seen as a random process, and then the only feasible modeling framework is the usage of statistics and probability theory. On the other side traditional linguistics has developed a number of theoretical frameworks based on logic and formal language theory, which also try to describe the complex phenomena in the language. While the former is robust and scalable enough to handle free text, the later is more precise.

The advantage of each of the paradigms is the disadvantage for the other, and the scientific community is aware that both paradigms have limitations. As a result, there is a new trend where both statistical and knowledge (logic) based techniques are combined in new ways. For instance Volume 6 (2011) of the "Linguistic Issues in Language Technology" journal is entirely devoted to the position of the leading researchers about the gap between the paradigms. We propose to design a framework where both logical and statistical processing can coexist in an uniform and mutually beneficial way.

For instance whenever logical reasoning is involved, the search for a proof always includes a number of arbitrary choices. In a typical scenario, most of the time is spent on backtracking for finding the right sequence of choices, and we could speed it up by ranking the alternative choices with some statistical model. This is the trick that is used in all statistical parsers. Parsing is usually seen as a logical deduction, but if it was only that then the number of independent choices would be enormous. Instead all statistical parsers still do deduction but they take first choices that are more likely to contribute to the final result. Similarly if there are more than one ways to prove a statement, then we often want the simplest proof. In parsing, we do not want

all possible analyses of the sentence but only the one that gets the highest score. Both the notion of proof simplicity and the ranking of the syntactic analyses are instances of a problem that can be modeled statistically.

The introduction of logical aspects in a statistical model is beneficial as well since the logical dependencies naturally define conditional probabilities in the model. More concretely, we are interested in using logical definitions as a way to program rich statistical models tailored to the specific application.

Our aim is to build a scalable framework which can process unrestricted text while still being able to encode complex linguistic knowledge in a logical framework. Statistics will be used to guide the logical reasoning and the syntactic parsing in the most prominent direction. Conversely, the logical definitions will define statistical models where logical relations will naturally lead to conditional probabilities.

Furthermore, we aim at architecture with sufficient level of abstraction between the concrete syntactic features of a language and its logical representation. In the modern statistical systems, the syntactic analysis of a sentence differs significantly from the analysis of its translation, even if the target language is closely related with the original language. On the contrary our framework will produce analyses which are as close as possible even if the two languages are far apart. This will immediately boost the development of new cross lingual applications. In particular, we are targeting machine translation and multilingual information extraction.

2 Survey of the field

Parsing free text has a long history but perhaps the most pioneering work on parsing with probabilistic context-free grammars is due to Charniak et al. [1998], Charniak [2000], Collins [1997] and Collins [2003]. In parallel, formalisms like Tree Adjoining Grammar (TAG) and Combinatory Categorial Grammar (CCG) have become increasingly popular since they are more expressive than context-free grammars but still they are efficient enough and their statistical models are learnable from the available data [Chiang, 2000][Clark and Curran, 2007]. In this proposal, we are particularly interested in using **Parallel Multiple Context-Free Grammars** (PMCFG) [Seki et al., 1991] since this is the assembly language for the high-level language **Grammatical Framework** (GF) [Ranta, 2004]. Data-driven statistical processing with Linear Context-Free Rewriting Systems (a formalism very

close to PMCFG) has been shown useful in Kato et al. [2006] and Kallmeyer and Maier [2010].

GF is an expressive linguistic framework where both the syntax and the semantics of a natural language can be described in a clear and human understandable way. However, GF is more in the spirit of the traditional frameworks like Head-Driven Phrase Structure Grammar (HPSG) and Lexical Functional Grammar (LFG), and as such it has been limited so far to processing only restricted controlled language. Similarly, large scale statistical processing with HPSG and LFG is still difficult although there is a recent work [Miyao and Tsujii, 2005] [Riezler et al., 2002] in this direction. Fortunately, the reducibility of GF to the low-level PMCFG formalism makes it possible to do language processing comparable in efficiency with the best statistical parsers [Angelov, 2011], while retaining the high-level nature of the grammatical description.

In addition to the PMCFG based syntactic framework, GF has also a logical framework based on Martin-Löf [1984] type theory, where each GF grammar defines a set of type signatures and function definitions. This is a Turing complete functional language which is used in a number of ways.

First of all the embedded lambda calculus and the constraints from the dependent types in the logical framework make it easy to do different kinds of semantic processing. In combination with an efficient parser, this lets us to do wide-coverage semantic analysis in the style of Bos et al. [2004].

The type signatures in the functional language can be seen by the Curry-Howard correspondence as a logic program which GF uses for reasoning. This reasoning capability has been used both for knowledge representation and for sentence planning when the framework is used for natural language generation. The current implementation of the reasoner [Angelov, 2011] is inspired by λ**Prolog** [Nadathur and Miller, 1988] but our implementation is still considerably slower than λProlog. We plan to improve this by directly linking to the latest implementation of the λProlog interpreter [Qi, 2009]. In addition we will need weighted search which will add statistical aspects to the framework.

As a logic language composed of type signatures, GF is very similar to Twelf [Pfenning and Schurmann, 1999], and as a probabilistic logic language it is similar to Dyna [Eisner and Filardo, 2011] and ProbLog [Kimmig et al., 2011]. None of these languages, however, has an associated syntactic framework like GF, and if they are to be used for natural language processing then the parsing algorithm has to be implemented in the language itself. Dyna

and ProbLog are still in their early stage but they are already a useful source of ideas. The main feature that we miss from a logical perspective is the lack of embedded lambda calculus. On the contrary, it is an essential part of both λProlog and Twelf.

Another distinctive feature of GF is its ability to build multilingual abstractions. While the concrete syntax of a language is defined in the syntactic framework, the analysis that is produced from the parser is an abstract expression in the logical framework. Thanks to the high expressivity of PMCFG and the mapping between concrete and abstract terms, it is possible to have grammars where the analysis of a sentence is kept the same or almost the same when it is translated from one language to another. Following this idea, the GF community has developed a library of wide-coverage **resource grammars** for currently 21 complete and 5 still incomplete languages with shared abstract representations. In combination with a statistical disambiguation model, this will give us an unique multilingual system with wide-coverage.

3 Project description

The project has two phases. In the first phase, we will improve the efficiency of our parser by guiding the search with statistics. Although all algorithms will be language independent, we will start with processing English and Swedish, since we already have the training data for these languages. We also want to show the applicability of the approach for other not so closely related languages, so we will choose a third language in the later stages. We will also offer student projects in the department for adding more languages.

The application of statistics for guided parsing will build on ideas from Kallmeyer and Maier [2010] but we will generalize it in two ways. First of all we will need to move from using Linear Context-Free Rewriting Systems to the slightly more general PMCFG formalism. Second, the model in Kallmeyer and Maier [2010] does not take into account statistical dependencies between the different parts of the sentence. Essentially this means that the model does not try to solve the famous PP-attachement problem.

The introduction of dependencies is the link to the second phase of the project where we want to improve the scalability of the logical reasoner. Here the first step it to integrate the GF runtime with the λProlog runtime. This will also require an extension to the GF compiler which will have to generate λProlog bytecode. In addition, we will have to add three features to the

λProlog runtime which are needed for GF. The major difference is that while in λProlog it is only possible to define logical predicates, in GF it is possible to have both predicates (type signatures) and functions. The result is a hybrid logic-functional language. The other two differences are that we need a bounded search which ensures the termination and a randomized search where the clauses are chosen at random. Esentially these two features taken together will turn λProlog from programming language to an automated theorem prover.

The dependencies in the logical framework naturally lead to conditional probabilities in the statistical model. For instance, let say that we have a rule for syntactic predication. In the logical framework it can be defined as:

```
PredVP : NP -> VP -> S
```

This simply says that if there is a noun phrase (NP), i.e. a subject, and a verb phrase (VP), then we can build a sentence. Note that this definition does not say anything about the word order or the gender/number agreement because this is handled in the syntactic framework. The abstract logical definition specifies only the general schema which is more language independent. The probabilities are also computed in the logical framework and they must take into account the dependencies in the type signature. In this case there are no dependencies at all which means that the probability for the whole term will be equal to the product of the probability of the function `PredVP` with the probabilities for the sub-terms of the NP and VP arguments. If we want to introduce conditional probabilities, then we must use **dependent types**:

```
PredVP : (v : V) -> NP v -> VP v -> S
UseV   : (v : V) -> VP v
UseN   : (n : N) -> (v : V) -> HasArg n v -> NP v
```

Here we added the function `UseV` which builds a verb phrase from an intransitive verb, but we also added the verb `v` as an index to the categories VP and NP. Furthermore the new type for `PredVP` specifies that the indices for the subject and the verb phrase must be the same. In this way the constructors for the noun phrase have access to the value of the verb. This is used in the type of function `UseN` which is applicable only if the type `HasArg n v` is inhabited. We can ensure this by adding one constructor for each combination of a subject and a verb that is encountered in the training data. In addition we can have the function:

```
any : (n : N) -> (v : V) -> HasArg n v
```

which guarantees that the processing is robust, i.e. the type is always inhabited. Now if we define a statistical distribution over the abstract representations of a sentence, then the probabilities for the noun phrase must be conditional on the choice of the verb. In addition, the existence of the function `any` will introduce a smoothing parameter. This schema is a realization of Model 2 from Collins [2003] in type theory. The exact algorithm for computing probability distributions over dependently typed terms is still a research question but we see some relation in Martin-Löf [1988] where the terms are represented as sequences of choices. The addition of probabilities for the choices will define a distribution.

We want the statistical model to be used for guiding both the parser and the theorem prover. Finding the most probable analysis of a sentence, i.e. the Viterbi tree, is important for parsing and similarly finding the Viterbi proof is important for natural language generation when the theorem prover is used as a sentence planer. The most probable proof is likely to generate a more natural sentence. The guided proof search will be another extension to the λProlog runtime.

4 Significance

The project involves both challenging scalability issues and deep theoretical questions about the interaction between logic, syntax and probabilities. The hybrid approach to natural language processing is a new trend and we believe that a systematic integration will be highly influential in the community. In order to encourage the exchange of ideas and to reduce the effort by other researchers we will release our software as open source.

5 Preliminary results

An important step towards wide-scale text processing with GF was the development of an efficient parsing algorithm [Angelov, 2011]. For instance, for the German resource grammar, we observed about 400 times improvement. During the MOLTO project (www.molto-project.eu) the parser was reimplemented from Haskell to C which gave us another efficiency boost. The latest implementation is only 2-3 times slower than the Stanford Parser

[Klein and Manning, 2003] without using any statistics for guiding the search. Furthermore, although the theoretical complexity for PMCFG is polynomial, in practice, for the resource grammars we measured linear complexity [Angelov, 2011]. The parser also offers an interface for plugging external tools like named entity recognizers and n-gram models which can complement the rule based parser. The new C implementation is the starting point for the integration with the λProlog runtime which is also written in C.

As part of MOLTO, we translated 92% of the Penn Treebank, and we are using the data for building statistical model for the English resource grammar. A parallel project creates a model from Talbanken for Swedish. The models are used to extract the most probable result from the chart of all analyses that the rule based parser found. Again within MOLTO, we made the GF parser robust. Now even if a sentence is not parseable by the grammar, the parser still produces partial output.

We will get both the parser and the statistical model from MOLTO but their systematic integration is the topic of the current proposal.

6 International and national collaboration

The research will be in collaboration with leading experts from the different fields affecting the proposal. In our home department, we have experts in type theory (Thierry Coquand, Peter Dybjer, Bengt Nordström, Aarne Ranta), algorithms and machine learning (Devdatt Dubhashi), syntax and morphology (Aarne Ranta). In addition there are the networks of the Center of Language Technology in Gothenburg and the Swedish National Graduate School of Language Technology where we are in contact with Lars Borin, Robin Cooper, Elisabet Engdahl and Joakim Nivre. Our research group will continue its collaboration with the group in machine translation in UPC Barcelona (Lluís Màrquez) which is currently involved in MOLTO.

There are three companies that showed interested in GF and are using it for different purposes: Be Informed (The Netherlands), Galois (USA) and Ontotext (Bulgaria).

7 Independent line of research

The project builds on top of the applicant's doctoral thesis where the efficiency of the GF interpreter was radically improved. The combination of statistics and linguistic grammars is a new initiative and it has not been tried before the MOLTO project. However in MOLTO the main goals are more modest and focus on a hybrid translation where GF is used as a rule based engine which complements a statistical translation system.

8 Form of employment

I am currently employed as a PostDoc at the department.

References

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

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. Wide-coverage semantic representations from a CCG parser. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. doi: 10.3115/1220355.1220535.

Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, NAACL 2000, pages 132–139, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

Eugene Charniak, Sharon Goldwater, and Mark Johnson. Edge-based best-first chart parsing. In *Sixth Workshop on Very Large Corpora*, 1998.

David Chiang. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 456–463, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. doi: 10.3115/1075218.1075276.

Stephen Clark and James R. Curran. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Comput. Linguist.*, 33(4):493–552, December 2007. ISSN 0891-2017. doi: 10.1162/coli.2007.33.4.493.

Michael Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, ACL '98, pages 16–23, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics. doi: 10.3115/976909.979620.

Michael Collins. Head-driven statistical models for natural language parsing. *Comput. Linguist.*, 29(4):589–637, December 2003. ISSN 0891-2017. doi: 10.1162/089120103322753356.

Jason Eisner and Nathaniel W. Filardo. Dyna: Extending Datalog for modern AI. In Tim Furche, Georg Gottlob, Giovanni Grasso, Oege de Moor, and Andrew Sellers, editors, *Datalog 2.0*, Lecture Notes in Computer Science. Springer, 2011. 40 pages.

Laura Kallmeyer and Wolfgang Maier. Data-driven parsing with probabilistic linear context-free rewriting systems. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 537–545, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Yuki Kato, Hiroyuki Seki, and Tadao Kasami. Stochastic multiple context-free grammar for RNA pseudoknot modeling. In *Proceedings of the Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms*, TAGRF '06, pages 57–64, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. ISBN 1-932432-85-X.

Angelika Kimmig, Bart Demoen, Luc De Raedt, Vítor Santos Costa, and Ricardo Rocha. On the implementation of the probabilistic logic programming language ProbLog. *Theory and Practice of Logic Programming*, 11 (2-3):235–262, 2011. ISSN 1471-0684. doi: 10.1017/S1471068410000566.

Dan Klein and Christopher D. Manning. Fast exact inference with a factored model for natural language parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS*, pages 3–10. MIT Press, 2003.

Per Martin-Löf. *Intuitionistic Type Theory*. Napoli: Bibliopolis, 1984.

Per Martin-Löf. Mathematics of infinity. In Per Martin-Löf and Grigori Mints, editors, *Conference on Computer Logic*, volume 417 of *Lecture Notes in Computer Science*, pages 146–197. Springer, 1988. ISBN 3-540-52335-9.

Yusuke Miyao and Jun'ichi Tsujii. Probabilistic disambiguation models for wide-coverage hpsg parsing. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 83–90, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219851.

Gopalan Nadathur and Dale Miller. An overview of λProlog. In *Fifth International Logic Programming Conference*, pages 810–827. MIT Press, 1988.

Frank Pfenning and Carsten Schurmann. System description: Twelf — a meta-logical framework for deductive systems. In *Proceedings of the 16th International Conference on Automated Deduction (CADE-16*, pages 202–206. Springer-Verlag LNAI, 1999.

Xiaochu Qi. An implementation of the language Lambda Prolog organized around higher-order pattern unification. *CoRR*, abs/0911.5203, 2009.

Aarne Ranta. Grammatical Framework: A Type-Theoretical Grammar Formalism. *Journal of Functional Programming*, 14(2):145–189, March 2004. ISSN 0956-7968. doi: http://dx.doi.org/10.1017/S0956796803004738.

Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Iii Mark Johnson. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th meeting of the ACL*, pages 271–278, 2002.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, October 1991. ISSN 0304-3975.