Robust Parsing with a Large HPSG Grammar

Yi Zhang, Valia Kordoni

German Research Center for Artificial Intelligence (DFKI GmbH)
Department of Computational Linguistics, Saarland University
P.O.Box 15 11 50, D-66041
Saarbrücken, Germany
{*yzhang,kordoni*}*@coli.uni-sb.de*

Abstract

In this paper we propose a partial parsing model which achieves robust parsing with a large HPSG grammar. Constraint-based precision grammars, like the HPSG grammar we are using for the experiments reported in this paper, typically lack robustness, especially when applied to real world texts. To maximally recover the linguistic knowledge from an unsuccessful parse, a proper selection model must be used. Also, the efficiency challenges usually presented by the selection model must be answered. Building on the work reported in Zhang et al. (2007a), we further propose a new partial parsing model that splits the parsing process into two stages, both of which use the bottom-up chart-based parsing algorithm. The algorithm is implemented and a preliminary experiment shows promising results.

1. Introduction

Linguistically motivated precision grammars are highly valuable language resources which provide in-depth modeling of complex language phenomena. Based on sound linguistic theoretical backgrounds and rigid mathematical formalisations, such approaches to natural language processing are capable of delivering highly accurate analyses when compared to shallower NLP systems. As pivotal central parts of continuous efforts on grammar engineering over the last decade, several of such grammars have achieved broad coverage on various linguistic phenomena in recent years, and have been successfully integrated in several NLP applications including information extraction, question answering, grammar checking, machine translation, and intelligent information retrieval, among others.

However, being highly restricted rule systems, these grammars are typically vulnerable to noisy inputs, and perform badly in terms of robustness. This is one of the major reasons why, despite being highly valuable language resources, precision grammars have been very much underused in real world applications in the past decades. Baldwin et al. (2004) reported that the *jun-04* version of the English Resource Grammar (ERG; Flickinger (2002)) achieves full lexical span[1] over a mere 32% of a random sample of 20K BNC strings. Among these inputs, 57% receive at least one analysis. Through a series of parsing coverage tests, Zhang and Kordoni (2006) also showed that, at least for grammars similar to the ERG, incomplete lexicon is one of the major sources of parsing failures, with the other major source being missing grammar constructions.

Targeting the missing lexical coverage in hand-crafted lexica of manually developed linguistically motivated precision grammars, like the ones mentioned above, several deep lexical acquisition approaches have been proposed (cf., Baldwin (2005), Zhang and Kordoni (2006)). The general idea shared among such approaches is to use available language resources (either derived from the grammar outputs themselvers – the so called *in vivo* deep lexical acquisition approaches –, or from external language resources – the so called *in vitro* lexical acquisition approaches) in order to automatically acquire the required linguistic knowledge and extend the lexicon. While the lexical coverage has been proven to largely improve with statistical lexical type prediction models like the one proposed in Zhang and Kordoni (2006), for instance, the missing constructions present a more serious coverage gap, as also briefly mentioned above. More specifically, in (Zhang, 2007), a coverage test run with chronologically different versions of the ERG has shown that, with the increased efforts invested into grammar engineering, the coverage of the specific grammar has shown a very promising improvement over the years. However, it is still unlikely for the specific precision large-scale grammar to achieve full coverage on unseen data without extra robust processing techniques. Also, the cost of manually extending the grammar would be too high to be easily acceptable for other precision grammar-based parsing systems.

In (Zhang et al., 2007a), we have pointed out that most applications are only interested in certain aspects of parsing results. Full analyses are preferable, but not always necessary. In fact, most of the contemporary deep parsing systems provide as outputs either semantic representations that reflect the "meaning" of the input, or rather abstract syntactic structures. Full representations with all detailed linguistic features (e.g., typed feature structures in HPSG) are almost never used either as output format or in real applications. Take the DELPH-IN HPSG grammars, for instance: Minimal Recursion Semantics (MRS, Copestake et al. (2005)) is used as the semantic representation in these grammars. For recording syntactic structures, derivation trees are usually used. Based on this fact, (Zhang et al., 2007a) have proposed to use partial parsing models to recover the most useful fragment analyses from the intermediate parsing results in cases of unsuccessful parses. To this effect, two statistical partial parse selection models are formulated, implemented, and evaluated.

Along the lines of the analysis presented in (Zhang et al., 2007a), in this paper we propose a more elaborated par-

[1] A sentence which has a full lexical span from a grammar contains only words already licensed by the lexicon of the aforementioned grammar.

tial parsing model, in order to further simplify the training procedure, so that full parse disambiguation models can be reused in partial parsing. Moreover, this new model enables us to obtain complete derivation trees, instead of a set of subtrees. Furthermore, with robust semantic composition rules, the fragment semantic representations can be put together in a robust, yet informative way.

The rest of the paper is structured as follows. Section 2. provides background knowledge about the DELPH-IN HPSG grammars, the semantic and syntactic representations, and the partial parsing model presented in Kasper et al. (1999) and Zhang et al. (2007a). Section 3. presents the new proposed two-stage robust parsing model. Section 4. further elaborates on the implementation details of the two-stage parsing model, including a detailed presentation of the efficient processing techniques. Section 5. presents a preliminary evaluation with the ERG using the PARC 700 Dependency Treebank (King et al. (2003)) sentences. In Section 6. we discuss the advantages of our model, as well as the remaining questions for future work. Section 7. concludes the paper.

2. Background

Head-driven Phrase Structure Grammar (HPSG, Pollard and Sag (1994)) is a well-known constraint-based grammar formalism. Being a highly consistent grammar framework, HPSG is a linguistic theory formulated purely with Typed Feature Structures (TFSes, cf., Carpenter (1992)). Due to its rigid mathematical foundation, HPSG has been widely adopted in the development of linguistically motivated large-scale precision grammars for different languages.

Head-driven Phrase Structure Grammar is also at the heart of DELPH-IN, a community effort on deep linguistic processing with HPSG, which has delivered the most promising multilingual parallel grammar development with HPSG to date. With a complete software tool-chain, ranging from a grammar engineering platform, the LKB system (cf., Copestake (2002)), to performance profiling and teebanking systems, the [incr tsdb()] platform (cf., Oepen (2001)), an efficient parser, PET (Callmeier, 2001), and a hybrid processing middle-ware, the HoG architecture (Callmeier et al., 2004), linguists and computer scientists are able to work together to develop language resources and applications with profound linguistic knowledge.

One of the most well-developed grammars in DELPH-IN (also among hand-crafted grammars in any other framework) is the English Resource Grammar (ERG, Flickinger (2002)). The grammar achieves broad coverage on various linguistic phenomena, but still remains rather restricted. Therefore, it is not only used for parsing, but also for text generation tasks.

On the semantic level, the English Resource Grammar outputs representations in the form of Minimal Recursion Semantics (MRS, Copestake et al. (2005)), a representation framework for computational semantics. The main assumption behind MRS is that the interesting linguistic units for computational semantics are the elementary predications (EPs), which are single relations with associated arguments. The flat (non-recursive) structure of MRS is especially suitable for situations where semantic composition is desired. Moreover, it can be easily integrated with the HPSG grammar by embedding the MRS structure into the typed feature structures.

On the syntactic level, on the other hand, a complete typed feature structure should be used, in principle. However, this is not necessary, for most of the features in the TFS are considered internal to the grammar, and not suitable as output format.[2] In practice, the derivation trees are used. For the DELPH-IN grammars, a derivation tree is composed of leaf notes, each of which corresponds to a lexical entry, and intermediate nodes, each of which corresponds to a grammar rule. Given an input and a grammar, a derivation tree records how an analysis is derived. By applying grammar rules on the lexical entries in the way indicated by a derivation tree, one can easily recreate the whole typed feature structure. For this reason, the DELPH-IN treebanks (Oepen et al., 2002; Bond et al., 2004) only record derivation trees.

Theoretically, the computational complexity in unification-based parsing is exponential to the length of the input. Given large-scale grammars like the ERG, it is crucial to have an efficient parser that can discover analyses licensed by the grammar. With continuous development in recent years, the PET (Callmeier, 2001) parser has grown to be one of the central components in the DELPH-IN software tool-chain. PET is based on a bottom-up chart-based algorithm, equipped with various efficient processing techniques, including quick-check, ambiguity packing and selective unpacking, among others. The robust parsing model proposed in this paper has been implemented as an extension to the PET parser.

We should point out that this is not the first work to propose a partial parsing model in order to improve the robustness of a hand-crafted grammar. Although the idea is usually to construct meaningful output structures from intermediate unsuccessful parsing results, the definition of a partial parse is not consentaneous. It is largely dependent on the paradigm of the parsing model. For instance, with bottom-up chart-based parsing, Kasper et al. (1999) proposed to define a partial parse as a set of consecutive non-overlapping passive parsing edges that together cover the entire input. In cases where a multiple partial parse exists, a selection criterion is required to decide which one is more preferable. In other words, a partial parse selection model is required. One of the simplest and most commonly used criterion is to prefer the partial parses which contain an edge that covers the largest fragment of the input. However, there is no strong motivation that makes this a good selection model.

An alternative selection model proposed by Kasper et al. (1999) is to consider the parsing chart as a directed graph, with vertex being all the positions between input tokens, and arcs being passive parsing edges on the chart. Then a best partial parse (as a set of arcs in the graph) connects

[2] We do not attempt to define here which representation is more suitable as a parser output in a cross-framework context. In fact, it is especially difficult to determine how syntactic information should be presented as parser output. Therefore, the choice of representation here is specific to the grammar in question, i.e., the ERG.

the shortest path from the beginning to the end of the input. Kasper et al. (1999) pointed out that the weights of the arcs can be assigned by an estimation function in order to indicate the preference over different fragment analyses. The discovery of such a path can be done in linear time ($O(|V|+|E|)$) with the DAG-shortest-path algorithm (Cormen et al., 1990). However, it is not clear (apart from some simple heuristics) how the estimation function can be acquired. Moreover, by its additive nature, the *shortest-path*, such a model makes an implicit independence assumption of the estimation function in different edge contexts.

Based on a similar definition of partial parse, Zhang et al. (2007a) formulated the following statistical model:

$$\log P(\Phi|w) \approx \log P(\Omega|w) + \sum_{i=1}^{k} \log P(t_i|w_i) \quad (1)$$

The above model contains two probabilistic components: i) $P(\Omega|w)$ is the conditional probability of a segmentation Ω given the input sequence w; and ii) $P(t_i|w_i)$ is the conditional probability of an analysis t_i for a given subsequence w_i in the segmentation. The empirical results have shown that this selection model significantly outperforms the shortest-path based baseline selection model proposed by Kasper et al. (1999). The evaluation was done using multiple metrics. While there is no gold-standard corpus for the purpose of partial parse evaluation, Zhang et al. (2007a) manually compared the parser's partial derivation trees with the Penn Treebank annotation for syntactic similarity. Furthermore, Zhang et al. (2007a) evaluated the fragment semantic outputs based on a practical estimation of RMRS similarities described by Dridan and Bond (2006). The semantic outputs of different partial parse selection models were compared to the RMRS outputs from the RASP system (Briscoe et al., 2006). If taken comparatively, all the results suggested that the model in (2.) performed much better than the baseline. But they failed to tell a clear story about the quality of the partial parse selection model.

Unfortunately, the model is approximate because of the independence assumption between the two components (for simplification). Also, due to the lack of training data, the parameters of the two components were estimated over different data sets in the experiment, which has added further doubt on the consistency of the resulting model. Moreover, it is generally not desirable to have different statistical models for full and partial parse selection. Ideally, a uniform disambiguation model should be used in both cases.

3. A Two-stage Robust Parsing Model

One common shortcoming of the partial parsing models proposed in both (Kasper et al., 1999) and (Zhang et al., 2007a) is that the results of partial parsing are sets of disjoint sub-analyses, either in the form of derivation subtrees, or in the form of MRS fragments. It is not informative enough to show the interconnection across the fragment boundaries. It is not enough, either, to tell why a full analysis is not derived for the given input.

Ideally, the partial parsing model should not only tell us which are good sub-analyses, but also predict what the missing parts from a full analysis are, should the input be licensed by the grammar. In a bottom-up chart-based parser, when a full analysis is not derived, the parser stops at a stage where no more grammar rule can be applied to either combine or create new edges on the chart. At this stage, all the passive edges on the parsing chart represent a licensed local analysis for the tokens within its span. Typically, for a broad coverage precision grammar with a well-formed input, certain rules fail to apply because some constraints are too strict. By relaxing the constraints in grammar rules, more robustness can be achieved.

The basic idea of the robust parsing model we propose in this paper is to use a set of less restrictive grammar rules to continue parsing with the passive parsing edges created with HPSG rules and lexical entries during the unsuccessful parse. To differentiate these less restrictive grammar rules from the original HPSG rules, we call them *robust rules*.

Several different ways of acquiring robust rules exist. In this paper, we use a context-free backbone grammar to simulate the behaviour of original HPSG rules. By choosing the CFG backbone, we will ignore the constraints encoded as typed feature structures. This allow us to generalise the approach beyond the specific grammar. Also, the robust parsing model we are concerned with in this paper focuses on improving constructional coverage. Therefore, only syntactic phrase structure rules are extracted. The missing lexical entries, together with the lexical rules should be captured through the lexical acquisition process. Figure 1 gives an example HPSG derivation tree and the corresponding CFG backbone.

Using these rules, together with the passive parsing edges create with HPSG rules in the first parsing stage, we are likely to be able to build larger analysis trees during the second parsing stage when the TFS unification-based parsing is substituted by CFG parsing. All the TFSes created are ignored (but still kept along with the passive edges created during the fist stage). Only the rules symbols are used as the category of the edge. Since the CFG backbone grammar uses the HPSG grammar rules names for its non-terminal nodes, the resulting parse trees are very similar to the HPSG derivation trees. The only difference is that a valid TFS cannot be recreated for those nodes constructed with CFG rules. We call such trees *pseudo-derivation trees.*

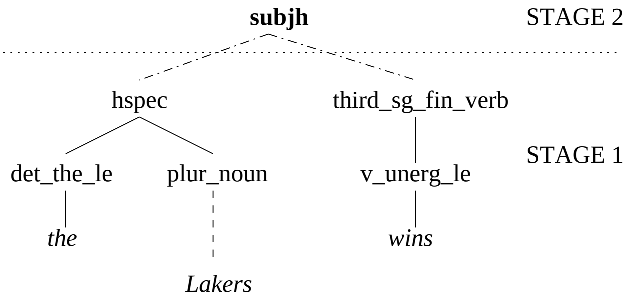
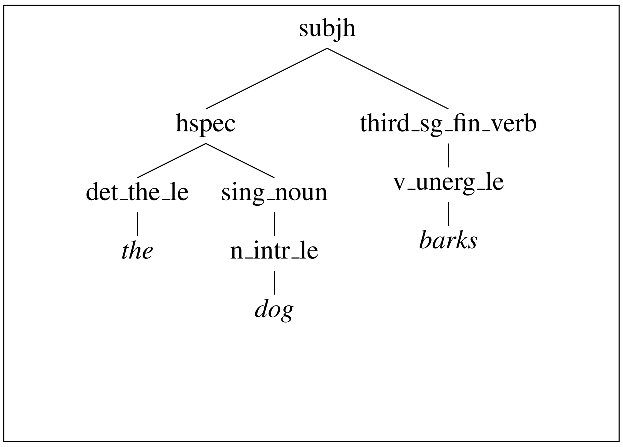


Figure 2: An example of pseudo-derivation tree in a two-stage robust parsing model

Figure 2 gives an example of a pseudo-derivation tree for the input *the Lakers wins*. Suppose the HPSG lexicon does



```
subjh  →  hspec  third_sg_fin_verb
hspec  →  det_the_le  sing_noun
```

Figure 1: An example HPSG derivation tree and its corresponding CFG backbone. Note that the lexical rules (unary projections from pre-terminal nodes) are not included in the CFG backbone.

not have a proper noun entry for *Lakers*, this will be falsely analysed as a *plur_noun* during the first parsing stage. The first parsing stage stalls at the point where the HPSG *head-subject* fails to apply because of the disagreement on the number of the subject and the head phrase. With the CFG rule:

```
subjh  →  hspec  third_sg_fin_verb
```

a CFG passive edge *subj* is constructed during the second parsing stage; this covers the entire input, and completes the pseudo-derivation tree.

Constructing pseudo-derivation trees does not only predict the structure of full analyses, but it also helps simplify the partial parse disambiguation process. In recent years, the log-linear model shown in (3.) has been widely used in many parsing systems. Toutanova et al. (2002) proposed an inventory of features that perform well in HPSG parse selection.

$$P(t|w) = \frac{\exp \sum_{j=1}^{n} \lambda_j f_j(t,w)}{\sum_{t' \in T} \exp \sum_{j=1}^{n} \lambda_j f_j(t',w)} \qquad (2)$$

For the DELPH-IN grammars, the best performing features comprise the depth-one sub-trees (or portions of these) with grammar rule names as node labels, plus optionally a chain of one or more dominating nodes (i.e., levels of grandparents). All these feature can be gathered from the derivation trees without consulting the TFSes. Therefore, the same discriminative model can be also applied to rank pseudo-derivation trees.

One potential risk of reusing the full parse disambiguation model is that the model $P(t|w)$ is conditional. Depending on the difference on the possible analyses (T) licensed by the grammar, the model is not guaranteed to be consistent when trained on a HPSG treebank and applied on CFG-based pseudo-derivation trees (a similar issue pointed out by Abney (1997)). A potential solution for this is discussed in Section 6.. However, we find that the full parse disambiguation model works very well in practice, for the CFG backbone extracted from the HPSG treebank closely mimics the behaviour of HPSG rules. In the experiment of this paper, a full parse disambiguation model trained on HPSG treebanks is directly used for partial parse ranking.

4. Some Notes on Implementation

The two-stage robust parsing model is implemented as an extension to the PET parser working with the *jul-07* version of the ERG.

The modified parser starts parsing with HPSG rules and TFS unification as usual. The second parsing starts when there is no full analysis found during the first stage. At the beginning of the second parsing stage, a new parsing chart is initiated with all passive parsing edges copied from the chart in the first stage. CFG rules are used to combine the passive edges and create new ones using an agenda-driven bottom-up algorithm. Extra checking must guarantee that new edges will not duplicate the existing passive edges (with same daughters and rule name) in the old chart.

For efficiency considerations, the PET parser uses subsumption-based ambiguity packing to effectively represent the local ambiguities. During the second parsing stage, there is no TFS for CFG passive edges; we use equivalence-based packing (i.e., two edges are packed together if they have the same span and share the same rule name). During unpacking, we use the selective unpacking algorithm proposed by Carroll and Oepen (2005) and Zhang et al. (2007b) to efficiently extract the most probable pseudo-derivation trees. The unpacking algorithm is slightly modified so that it will not try to instantiate the TFS for CFG edges. The rest parts of the unpacking algorithm remain the same, and extraction of exact n-best readings is guaranteed.

The CFG backbone grammar for ERG is extracted from the LOGON treebank (Oepen et al., 2004). We only extract syntactic rules that occur at least 5 times in the treebank. This gives us a CFG backbone grammar with about 2.5K unary and binary rules. For unary rules, we further filter out those that may lead to infinite recursion. We should point out that the decision of which CFG rules to extract is still an open question. Currently we only extract frequent rules, for they are more likely to be used in the ERG derivation trees. Moreover, by reducing the number of CFG rules, the second parsing stage becomes much more efficient. For parse disambiguation, we use the model trained on the LOGON treebank with depth-one tree features with up to 3 levels of grandparents, which has so far worked reasonably well in different application scenarios.

5. Evaluation

As Zhang et al. (2007a) have also pointed out, the evaluation of a partial parser is a very difficult task as such, due to the lack of gold-standard annotation for sentences that are not fully analysed by the grammar. For the purpose of evaluation, Zhang et al. (2007a) compared the partial derivation tree to the Penn Treebank bracketing, and partial RMRS fragments to the RASP RMRS outputs. Although the results have shown that the proposed partial parsing model performs comparatively better than the baseline model, it is not convincing in relation i) to how informative it is to compare HPSG derivations with Penn Treebank bracketings; and ii) to whether RASP RMRS output should be considered for evaluation comparison in the first place at all.

For these reasons, a manual evaluation has been carried out for the new proposed partial parsing model in this paper. For the experiment, we selected a subset of 267 sentences from the PARC 700 Dependency Bank (King et al., 2003), which have full lexical span licensed by the ERG. Among these sentences, 213 are parsed out of the box. For the remaining 54 sentences, the two-stage partial parsing model built pseudo-derivation trees for 41 of them. The remaining sentences are either not well-formed (exhibiting among them, for instance, garbage strings, incomplete utterances, etc.), or the parser is missing appropriate lexical entries.

Among those sentences for which pseudo-derivation trees could be constructed, 13 of them are completely correct, and another 18 have no more than 2 cross-bracketings. In about half of the cases where the pseudo-derivation tree is wrong, there is a key lexical entry missing in the grammar lexicon. This indicates that an automatic lexical acquisition model should be used in combination with the partial parsing model. Some errors in the pseudo-derivation trees indicate that the rule names symbols (as used in the derivation trees) are not informative enough for the CFG parser in the second stage in order for good predictions to be made.

6. Discussion

Although the evaluation shows promising improvement on the grammar coverage, it is noticed that the type of the robust rules in use plays a significant role in our robust parsing model. As pointed out in Section 3., the choice of robust rules is not limited to context-free grammars directly extracted from derivation trees. The flexibility allows us to achieve different levels of robustness, while maintaining the desired accuracy. In extreme cases, the robust rule may allow any sub-structures to be combined. But then it merely has any prediction power, and is practically equivalent to the shortest-path model. A context-free backbone grammar seems to be a reasonable choice, for it can be easily acquired from parser outputs, and can be used for efficient parsing. With rule symbols as CFG non-terminals, it appears to be too abstracted in some cases, and may lead to overgeneration. One solution to this would be to modify the CFG rules symbols with phrase categories (i.e., NP, VP, AP, PP, etc).

In Section 3. we have also mentioned that the parse disambiguation model trained on HPSG treebanks is not guaranteed to be consistent when used for pseudo-derivation tree disambiguation. The main reason is that some of the pseudo-derivation trees produced by the CFG are not licensed by the HPSG rules. It can be expected that with a set of relative strict robust rules the discrepancy would be relatively small. For rule sets which are much more relaxed than the HPSG rules, one could update the disambiguation model by extending the training HPSG treebank with the extra trees licensed by the robust rules.

Another interesting topic that we have not discussed so far is that the two-stage parsing model opens the possibility of achieving robust semantic composition. In HPSG, the semantic compositions are carried out simultaneously with the syntactic analyses. However, most of the composition can be done without the lexicalised syntactic information. By encoding the general semantic composition rules into the robust parsing rules, the fragment semantic representations can be connected.

Although this paper focuses on the robustness issue in relation to constructions, the fact that HPSG is a highly lexicalised framework entails that the lack of robustness in the lexicon may also lead to parsing failures (cf., Figure 2). If we think of the two-stage parsing model as a top-down approach to predict the upper part of a parse tree, then the automatic lexical acquisition model will serve as a bottom-up predictor that fills in the knowledge gaps about words. Exploring the interconnection between the two prediction models would be another interesting topic for our future work.

7. Conclusion

In this paper, we have proposed a two-stage model for robust parsing with a large HPSG grammar. The model uses a less restrictive grammar derived from the HPSG parser outputs to continue parsing based on the fragment analyses produced by the HPSG rules. With the pseudo-derivation trees constructed by the partial parsing model, the full parse disambiguation model is applied in partial parse selection. The approach also opens the possibility of achieving robust semantic composition which remains to be explored in the future work.

8. References

Steven Abney. 1997. Stochastic attribute-value grammars. *Computational Linguistics*, 23:597–618.

Timothy Baldwin, Emily M. Bender, Dan Flickinger, Ara Kim, and Stephan Oepen. 2004. Road-testing the English Resource Grammar over the British National Corpus. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal.

Timothy Baldwin. 2005. Bootstrapping deep lexical resources: Resources for courses. In *Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*, pages 67–76, Michigan , USA.

Francis Bond, Sanae Fujita, Chikara Hashimoto, Kaname Kasahara, Shigeko Nariyama, Eric Nichols, Akira Ohtani, Takaaki Tanaka, and Shigeaki Amano. 2004. The Hinoki Treebank: a treebank for text understanding. In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP 2004)*, pages 554–562, Hainan Island, China.

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 77–80, Sydney, Australia.

Ulrich Callmeier, Andreas Eisele, Ulrich Schäfer, and Melanie Siegel. 2004. The DeepThought core architecture framework. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal.

Ulrich Callmeier. 2001. Efficient parsing with large-scale unification grammars. Master's thesis, Universität des Saarlandes, Saarbrücken, Germany.

Bob Carpenter. 1992. *The Logic of Typed Feature Structures*. Cambridge University Press, Cambridge, UK.

John Carroll and Stephan Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP 2005)*, pages 165–176, Jeju Island, Korea.

Ann Copestake, Dan Flickinger, Carl J. Pollard, and Ivan A. Sag. 2005. Minimal recursion semantics: an introduction. *Research on Language and Computation*, 3(4):281–332.

Ann Copestake. 2002. *Implementing Typed Feature Structure Grammars*. CSLI, Stanford, USA.

Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. 1990. *Introduction to Algorithms*. MIT Press.

Rebecca Dridan and Francis Bond. 2006. Sentence comparison using Robust Minimal Recursion Semantics and an ontology. In *Proceedings of the ACL Workshop on Linguistic Distances*, pages 35–42, Sydney, Australia.

Dan Flickinger. 2002. On building a more efficient grammar by exploiting types. In Stephan Oepen, Dan Flickinger, Jun'ichi Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering*, pages 1–17. CSLI Publications.

Walter Kasper, Bernd Kiefer, Hans-Ulrich Krieger, C.J. Rupp, and Karsten Worm. 1999. Charting the depths of robust speech processing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL 1999)*, pages 405–412, Maryland, USA.

Tracy H. King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. 2003. The PARC 700 Dependency Bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora, held at the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, Budapest, Hungary.

Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning, Dan Flickinger, and Thorsten Brants. 2002. The LinGO Redwoods treebank: motivation and preliminary applications. In *Proceedings of COLING 2002: The 17th International Conference on Computational Linguistics: Project Notes*, Taipei, Taiwan.

Stephan Oepen, Helge Dyvik, Jan Tore Lønning, Erik Velldal, Dorothee Beermann, John Carroll, Dan Flickinger, Lars Hellan, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgård, and Victoria Rosén. 2004. Som å kapp-ete med trollet? Towards MRS-Based Norwegian–English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, USA.

Stephan Oepen. 2001. [incr tsdb()] — competence and performance laboratory. User manual. Technical report, Computational Linguistics, Saarland University, Saarbrücken, Germany.

Carl J. Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, USA.

Kristina Toutanova, Christoper D. Manning, Stuart M. Shieber, Dan Flickinger, and Stephan Oepen. 2002. Parse ranking for a rich HPSG grammar. In *Proceedings of the 1st Workshop on Treebanks and Linguistic Theories (TLT 2002)*, pages 253–263, Sozopol, Bulgaria.

Yi Zhang and Valia Kordoni. 2006. Automated deep lexical acquisition for robust open texts processing. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 275–280, Genoa, Italy.

Yi Zhang, Valia Kordoni, and Erin Fitzgerald. 2007a. Partial parse selection for robust deep processing. In *Proceedings of ACL 2007 Workshop on Deep Linguistic Processing*, pages 128–135, Prague, Czech.

Yi Zhang, Stephan Oepen, and John Carroll. 2007b. Efficiency in unification-based N-best parsing. In *Proceedings of the 10th International Conference on Parsing Technologies (IWPT 2007)*, pages 48–59, Prague, Czech.

Yi Zhang. 2007. *Robust Deep Linguistic Processing*. Ph.D. thesis, Saarland University.