Natural Language Engineering **1** (1): 000–000. Printed in the United Kingdom

Robust Parsing, Meaning Composition, and Evaluation

Integrating Grammar Approximation, Default Unification, and Elementary Semantic Dependencies[*]

Yi Zhang[†], Stephan Oepen[‡], Rebecca Dridan[‡], Dan Flickinger[♯], and Hans-Ulrich Krieger[†]

† *Language Technology Lab, DFKI GmbH*
Campus D3_2, 66123 Saarbrücken, Germany
‡ *Department of Informatics, University of Oslo*
Postboks 1080 Blindern, 0316 Oslo, Norway
♯ *Center for the Study of Language and Information, Stanford University*
Cordura Hall, Stanford, CA 94305, USA
{*yizhang,krieger*}*@dfki.de,* {*oe,rdridan*}*@ifi.uio.no, danf@stanford.edu*

(*Received 19 May 2014*)

Abstract

In the larger context of parsing for semantic interpretation, we present and evaluate a novel approach to corpus-driven approximation of linguistically rich, constraint-based grammars. We obtain an unlexicalized probabilistic context-free grammar (PCFG) from a very large corpus that is automatically annotated with the fine-grained syntacto-semantic analyses of a broad-coverage Head-Driven Phrase Structure Grammar (HPSG). PCFG parsing with the resulting approximation greatly improves robustness, while also increasing formal and computational simplicity. Different ways of encoding relevant syntactic context in the approximating PCFG are proposed and compared empirically, and a comparison to state-of-the-art latent-variable PCFG techniques suggests that our approach is more suitable for grammar approximation with massive amounts of training data. To recover logical-form meaning representations from the analyses of the robust PCFG parser, we develop a technique for robust semantic composition based on default unification. For evaluation of the semantic accuracy of the resulting robust analysis system, we propose a method of reducing the logical forms in the representation language predominant in computational HPSG to semantic dependency triples. Under various evaluation perspectives, parsing with the approximated grammar delivers competitive analysis quality.

[*] We thank Bernd Kiefer, Dan Flickinger, Johan Benum Evensberget, and our colleagues in the DELPH-IN network for many enlightening discussions related to the work presented in this article. We also thank the anonymous reviewers and audience at the 2011 International Conference on Parsing Technologies for their comments. The first author thanks the *Deependance* project funded by BMBF (01IW11003) for its support of the work. The second and third authors are in part supported by the Norwegian Research Council, through the *WeSearch* grant. The fifth author would like to thank the EU-funded project *TrendMiner* (FP7-ICT 287863) for its support.

1 Introduction & Motivation

The terms 'deep' and 'shallow' have frequently been used to characterize or contrast different approaches to parsing. Inevitably, such informal notions lack a clear definition, and there are few signs of community consensus on the relevant dimension(s) of depth, let alone agreement on applicable metrics. In this work, we suggest more of a gradual continuum rather than a stark opposition of approaches, and conjecture as a key desideratum for many parsing applications the goal of making explicit structural relations at the level of meaning, i.e. abstract dependencies that facilitate semantic interpretation: *Who did what to whom?* Although there is no conclusive evidence that the syntax of English (and most other natural languages) transcends the mathematical complexity of context-free grammar (CFG), traditionally parsing with CFGs (and equivalent formalisms) falls short of our semantics-centered ideal—a context-free phrase structure tree, in itself, can only provide purely syntactic dependencies.

Conversely, so-called 'deep' linguistic processing has co-evolved with mathematically richer formalisms (typically introducing at least mild context sensitivity). Examples of such frameworks include various types of Tree Adjoining Grammar (TAG), Combinatory Categorial Grammar (CCG), Lexical Functional Grammar (LFG) and Head-Driven Phrase Structure Grammar (HPSG). Albeit somewhat diverse in underlying linguistic assumptions, common to all of these is their use of complex, internally structured categories (in some cases, giving rise to infinite category inventories) and an information-combining operation like unification. While increased formal power facilitates a tight integration of syntactic analysis and semantic composition, the greater computational complexity of the above frameworks can hinder deployment in practical language technology applications.

The linguistic framework of HPSG, for example, centrally builds on the logic of typed feature structures, TFS for short, (e.g. a mathematical formalism or designer logic like the one of Carpenter, 1992). The monostratal representation in HPSG integrates a broad range of syntactic and semantic information concerning a linguistic object (and all its subcomponents) in a single typed feature structure. Compatibility checking and integration of information from multiple sources is accomplished by *graph unification*. Such a formalism is very suitable for the design and implementation of a detailed linguistic theory, but the lack of a polynomial upper bound on time complexity in unification-based parsing raises concerns about its computational tractability.

From a computational grammar engineering perspective, grammarians constantly need to juggle two somewhat conflicting goals: on the one hand, (a) to describe linguistic phenomena in a precise and adequately constrained manner; and on the other hand, (b) to provide broad grammatical coverage of unseen real-world text. As a consequence, large-scale computational grammars are often forced to choose (whether consciously or not) to either compromise linguistic precision, or to accept limitations in parsing coverage. In this article, we propose PCFG approximation as a way to alleviate some of these issues. While the HPSG framework is well-suited for linguistic description, we demonstrate that, with careful design, much simpler approximating probabilistic context-free grammars can

be extracted automatically and can achieve comparable parsing accuracy while offering increased robustness and the potential of greater computational efficiency.[1]

Moving from parsing with the original HPSG to parsing with an approximating PCFG comes at the expense of integrated semantic analysis: there cannot be meaning composition during parsing through unification of feature structures, although primitive forms are conceivable here through, e.g. attribute grammars (Knuth, 1968). More importantly, the robustness and computational simplicity of the PCFG parser are owed to its inherent ability to violate—or maybe more appropriately: ignore—parts of the grammatical constraints in the original HPSG. The CFG categories in all our PCFG approximations include the unique identifiers of lexical types (i.e. very fine-grained parts of speech) and all the syntactic constructions comprising a full HPSG derivation, which is enough information to unambiguously reconstruct corresponding HPSG analyses. However, there is no guarantee that such derivations will actually be consistent with respect to the full HPSG. Naïve post-processing of a CFG derivation in the HPSG universe will frequently give rise to unification failures. To work around these issues, we couple our PCFG parser with a robust procedure for meaning composition, grounded in *default unification*, that seeks to maximize the amount of semantic information available from the robust context-free analyses. In order to evaluate our results at the level of semantics, we propose a technique for reducing a full logical form into elementary, variable-free semantic dependencies and sketch (by example) how different types of information in parser outputs are encoded (or not) across frameworks.

In summary, the main contributions in this work are: (a) the automated derivation of robust, polynomial-complexity PCFG parsers from a computationally expensive and less robust HPSG; (b) an empirical investigation into the effects of contextual category refinements (i.e. 'node annotation') and variations in training data size; (c) a head-to-head comparison to latent-variable PCFG refinement, on large amounts of training data; (d) a proposal for the reduction of logical-form meaning representations as common in the HPSG universe into semantic dependencies for granular semantic parser evaluation; and (e) the application of robust meaning composition, using HPSG principles and rules, to PCFG derivations that are potentially inconsistent or out-of-scope in the original 'deep' grammar. While the resulting system is a complete end-to-end semantic parser, cross-framework and cross-domain parser comparison remain an open challenge. As we argue in Section 6 below, there are stark limitations to both intrinsic (i.e. representation-driven) and extrinsic (i.e. task-driven) parser evaluation, and the empirical validation of our work, thus, inevitably remains limited to framework-internal comparison. In terms of both syntactic and semantic accuracy, we contrast our PCFG parsers with the full, state-of-the-art HPSG system; we further seek to quantitatively gauge the utility of coverage gains in the robust system, even though there are no gold-standard HPSG target representations available. Al-

[1] Somewhat generally speaking, there is a parallel between our work and the study of Fowler and Penn (2010), who demonstrate that competitive parsing accuracy can be obtained by applying a state-of-the-art PCFG parser to CCG. However, they start from a CCG treebank (Hockenmaier & Steedman, 2007) that was mostly automatically derived from the Penn Treebank (PTB), rather than from a hand-crafted computational grammar. Furthermore, in this specific version of CCG, feature structures play a very limited role, and Fowler and Penn (2010) thus actually succeed in giving a proof of strong equivalence to CFG.

though an important part of our practical motivation, we exclude an in-depth investigation of observed parser efficiency from the present study, seeing as there are a great number of engineering trade-offs that need to be addressed in efficient parsing with either very large PCFGs or HPSGs.

The remainder of this article is organized as follows: Section 2 provides a brief introduction to HPSG as a linguistic theory and to its utilization in parsing; Section 3 reviews previous related work on context-free approximations of HPSG and clarifies what kind of approximations we are looking at; Section 4 develops our corpus-driven PCFG approximation approach, using both *internal* and *external* annotations; in Section 6, we turn to aspects of semantic parser evaluation and introduce our own (moderately novel) metric; Section 5.2 then discusses our approach to robust meaning recovery from PCFG derivations; details of data sets and parsers used and contrastive experimental results are presented and discussed in Section 7; Section 8 seeks to put our results into perspective in a larger context, including some recent work on parser self-training; finally, Section 9 concludes by speculating about various forms of (default) unification and other ways to recover from unification failures that we consider for exploration in future work.

2 Parsing with Head-Driven Phrase Structure Grammars

Head-Driven Phrase Structure Grammar (Pollard & Sag, 1994) is a framework for constraint-based, lexicalized, non-transformational grammatical description. Expressed in the formalism of typed feature structures, the HPSG theory assumes a relatively small inventory of general linguistic principles and syntactic construction, which through the interaction with very detailed lexical types (i.e. generalizations over lexeme classes) both delimit the space of grammatically wellformed structures and fully characterize each available analysis.

Several large-scale HPSG-based parsing systems have been built over the past decade. Among them are Enju for English & Chinese (Miyao, Ninomiya, & Tsujii, 2004; Yu, Yusuke, Wang, Matsuzaki, & Tsujii, 2010), Alpino for Dutch (van Noord, 2006), and the LKB & PET (Copestake, 2002; Callmeier, 2000) for English, German, Japanese, and close to a dozen other languages represented in the DELPH-IN network.[2] These systems are showcases of computational grammar engineering contributing to state-of-the-art parsing technologies and some of them have been shown to compete favorably in parsing unrestricted texts. Specialized techniques for feature structure manipulation, unification-based parsing, and statistical disambiguation (see for example Kiefer, Krieger, Carroll, & Malouf, 1999, Oepen, Flickinger, Tsujii, & Uszkoreit, 2002, and Miyao & Tsujii, 2008, inter alios) have enabled practical parsing times averaging at a few seconds or below, but due to the non-restrictive nature of the general formalism, the worst-case time complexity for these parsers nevertheless remains exponential.

Another challenge in parsing with 'deep' grammars lies in the difficulties of statistical modeling and disambiguation of large, internally finely structured representations. For example, Abney (1997) shows that naïve Maximum Likelihood Estimation (MLE) is not

[2] For background, please see http://www.delph-in.net/.

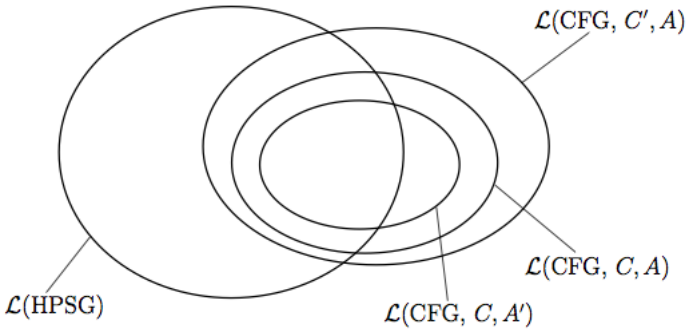

Fig. 1. The correlation between the language accepted by the HPSG $\mathcal{L}$(HPSG) and the approximated context-free language $\mathcal{L}$(CFG, C, A), given a corpus C and a set of (internal and external) annotations A. The approximated language $\mathcal{L}$(CFG, C', A) monotonically grows when given a larger corpus $C' \supseteq C$. The language $\mathcal{L}$(CFG, C, A') usually shrinks when given a larger set of annotations $A' \supseteq A$, but the *number* of rules grows.

consistent for unification-based grammars. In practice, we see most HPSG parsing systems opt for the use of a discriminatively trained *Maximum Entropy Model* (MEM) to probabilistically rank the hypothesis space licensed by the grammar (Miyao & Tsujii, 2002; Malouf & van Noord, 2004; Toutanova, Manning, Flickinger, & Oepen, 2005). For further efficiency, the hypothesis space of HPSG parses can be pruned by supertagging or CFG filtering rules (Matsuzaki, Miyao, & Tsujii, 2007) as preprocessing steps. It remains however unclear how such separate models can be combined to guide the best-first enumeration of HPSG parses without an exhaustive creation of the (packed) parse forest or inexact, ad hoc pruning.

3 A Note on Approximation and Survey of Related Work

In this article, we use the term *approximation* for methods that convert a given source grammar $\mathcal{G}$ in some formalism X into a new grammar $\mathcal{G}'$ in another formalism Y, so that $\mathcal{G}'$ is in some sense 'simpler' than $\mathcal{G}$. Simpler here means that a sentence that lies in $\mathcal{L}(\mathcal{G}) \cap \mathcal{L}(\mathcal{G}')$ can be recognized 'faster' by $\mathcal{G}'$ than by $\mathcal{G}$, due to the fact that the parsing complexity of Y is lower than that of X (in our case here, X is HPSG and Y is CFG). X might even be equal to Y (e.g. $X, Y =$ HPSG) in case rules and/or lexicon entries from G' are more general than those in G (Kasper & Krieger, 1996).

To be compatible with Pereira and Wright (1991), we use *sound* approximation when the following condition also holds for the generated languages: $\mathcal{L}(\mathcal{G}) \subseteq \mathcal{L}(\mathcal{G}')$.[3] For instance, a sound approximation of the context-free language $\{a^n b^n \mid n \geq 0\}$ is the regular language $(a+b)^\star$. Thus an *unsound* approximation can be characterized by $\mathcal{L}(\mathcal{G}) \cap \mathcal{L}(\mathcal{G}') \neq \emptyset$. The corpus-driven method used in this article is by its nature unsound, as depicted schematically in Figure 1.

[3] This notion of soundness should not be confused with soundness as used in logic.

Unsound approximation methods are not 'buggy' when used to produce domain-specific subgrammars, as they do neither produce a true superset nor a subset of the original language. This is a desired effect, since unsound grammars should cover the relevant constructions of a domain (and only these) and they should be robust against ungrammatical input w.r.t. the domain we are interested in. Finally, more training material and more annotations even allow us to monotonically come close to the original HPSG parse trees.

Previous work on approximating HPSG has seen two major approaches: (i) sound grammar-based and (ii) unsound corpus-driven approximations.

The *grammar-based* approach (Kiefer & Krieger, 2000, 2004) seeks to compile out a huge set of categories by flattening typed feature structures (TFSs) into atomic symbols. This approach can in principle derive a CFG equivalent to the original HPSG (at least where the underlying grammar actually circumscribes a finite set of categories). However, in practice it might generate billions of CFG productions, when applied to a genuinely broad-coverage HPSG. Even when carefully addressing only a subset of the information in the HPSG universe, the resulting grammars will often be too large to be useful for parsing or generation. Nevertheless, exhaustive parsing has benefitted by a speedup factor of 2–3 for the *Verbmobil* domain; see Kiefer and Krieger (2004).

The *corpus-based* approach (Krieger, 2004, 2006), on the other hand, builds the approximating CFG by observing the growth of the parse chart when analyzing text with the HPSG. Passive edges on the chart represent the successful application of HPSG rules, hence are modeled by an approximating CFG production. Also, the resulting CFG symbols typically preserve only partial information from the HPSG TFS, for example taking advantage of a small set of feature paths used in the so-called unification *quick check* (Kiefer et al., 1999), i.e. frequently failing and thus most discriminating feature paths in the HPSG universe.[4] As reported in (Krieger, 2006), exhaustive parsing with corpus-based HPSG approximations has led to a speedup of two orders of magnitude (again for the *Verbmobil* domain).

Both approaches are originally purely symbolic in the sense that there is no probabilistic model produced to disambiguate CFG parses. In the context of the corpus-based approach, one could of course also acquire frequency counts for each CFG rule. But since not all passive edges occur in a full parse tree, and not all parses are correct, the statistics available in the method of Krieger (2006) are not very suitable for the parsing task.

Kiefer, Krieger, and Prescher (2002) propose to use a PCFG in a two-stage parsing setup, where the PCFG predicts derivations in the first step, followed by the replay of unification. The experiment was carried out on a relatively small grammar, and, due to the lack of large-scale treebanks at the time, the unsupervised Inside – Outside algorithm was used for probability estimation.

Cahill, Burke, O'Donovan, Van Genabith, and Way (2004) report on an application of PCFG approximation to LFG parsing and the recovery of long-distance dependencies in LFG f-structures. Two main differences between their work and the one presented in this article should be noted. First, the approximation target for Cahill et al. (2004) is a treebank-induced grammar, while this article targets a large hand-crafted grammar. Sec-

[4] The *quick check* paths are practically determined based on the the statistics accumulated by carrying out the complete unifications in parsing a set of representative sentences with the HPSG.

ond, the monostratal representation in HPSG entails that a correct derivation tree will also guarantee the correct recovery of unbounded dependencies represented by the underlying feature structures, while in the LFG universe these have to be resolved separately on the f-structure projection level, instead of directly on LFG c-structures (i.e. the context-free backbone).

Further approaches from other frameworks are presented in depth in (Kiefer & Krieger, 2004) and (Krieger, 2006).

4 Probabilistic Context-Free Approximation of HPSG

Unlike the approaches in previous work which approximate the symbolic HPSG alone, we propose a PCFG approach which approximates the combination of the grammar and its disambiguation model. This allows us to closely model the deep parser behavior with a single approximating PCFG.

For the experiments in this article, we use the English Resource Grammar (ERG; Flickinger, 2002) and the accompanying treebanks (see Section 7.1 below for full details). But the technique presented in this section can be easily applied to other languages and HPSG implementations (and in principle to other unification-based frameworks as well).

4.1 Derivation Normalization

A complete HPSG analysis is recorded in a derivation tree. The terminals of the tree are surface tokens in the sentence. The preterminals name lexical entries used in the analysis. Internal tree nodes (excluding preterminals and the root), finally, correspond to grammatical constructions (rules) applied to derive the full HPSG sign. An extra root node denotes the HPSG 'start symbol' that licenses a complete parse. An example derivation from the ERG is given in Figure 2 together with a simplified feature structure of one constituent. In the derivation, the three binary HPSG schema used, for example, correspond to the subject–head, nominal compound, and head–complement constructions; conversely, unary rules like N_PL_OLR or HDN_BNP_C correspond to plural nominal inflection and the formation of a determinerless, 'bare' noun phrase, respectively. In total, the ERG distinguishes between some 270 construction types and close to 1,000 lexical types of preterminals.

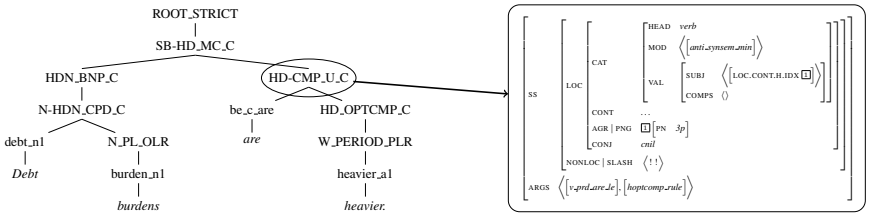

Fig. 2. Example of an original ERG derivation tree and the simplified feature structure of the constituent *"are heavier."*.

Before extracting the approximating grammar, we perform several normalizing transformations on the original derivations. Firstly, in order to acquire an unlexicalized grammar,

we generalize the lexical entry names of the preterminal with their corresponding lexical types, as defined in the ERG lexicon. Secondly, we collapse unary chains of lexical (i.e. derivational and inflectional) rules together with the preterminal lexical types to form a kind of *supertag*. As shown in Figure 2, these unary rules always occur above the preterminals and below any syntactic constructions. Experiments show that this helps improve the parsing accuracy of the PCFG. The last normalization is concerned with the treatment of punctuation. In the ERG, punctuation marks are analyzed as (pseudo-)affixes instead of independent tokens by themselves. For better compatibility with other annotations, we convert the original punctuation-attaching unary lexical rules (which tend to apply after other lexical rules and before any syntactic rules) into a binary branch. The normalized derivation tree of the previous example is shown in Figure 3. It is worth noting that all normalization steps can be reversed without introducing ambiguity. The approximating PCFGs will be extracted from the normalized derivations, while the evaluation will be reported on the original derivations (though the tagging accuracy will be calculated on the lexical types).

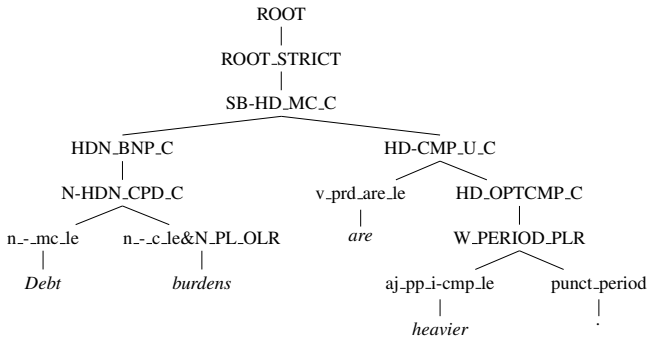

Fig. 3. Example of a normalized derivation tree.

4.2 PCFG with External Annotation

Although the derivation tree records all necessary information to recover the complete HPSG analysis (i.e. carrying out unification on each node of the tree with corresponding grammar rules and lexical entries), it does not always encode the necessary information in an explicit way, due to the fact that rules in HPSG are highly generalized (see Section 2). For example, the rule *hd-cmp_u_c* in ERG can be used to derive a head – complement construction independent of the specific syntactic category of the head. Thus a node marked only with *hd-cmp_u_c* could be a VP, PP, or $\overline{\text{N}}$. Therefore it will be difficult to accurately predict the derivation without such information. To compensate for the lack of local category information in the derivation tree, we add additional *annotations* to the non-terminals.

We further differentiate external annotation, i.e. additional information from the context of the tree node, and internal annotation, i.e. information coming from the HPSG sign.[5]

For the external annotation, we mark each non-terminal node with up to n grandparents. This is an effective technique used in both PCFG parsing (Johnson, 1998; Klein & Manning, 2003)[6] and HPSG parse disambiguation (Toutanova et al., 2005). As ERG rules are either unary or binary, we do not annotate nodes with sibling information, though for a grammar with flat rules this could potentially help, as shown by Klein and Manning (2003) (so-called *horizontal Markovization*). We choose not to annotate the preterminal supertags with grandparents, since the overly fine-grained tagset hurts the parsing coverage.

4.3 PCFG with Internal Annotation

While the external annotations enrich the derivation tree by gathering context information, internal annotations can explore the detailed HPSG sign associated with each local tree node. Note that an average ERG sign contains hundreds of feature paths and their corresponding values, hence it is important to pick a suitably small, yet effective subset of these as annotation. Following the practice of Krieger (2006), we choose to use up to six top-ranked *quick-check* paths (see Table 1), which are the most frequently *failing* feature paths in unification.

To access the HPSG sign, feature structures are *reconstructed* by unifying the corresponding TFS of the HPSG rule with the instantiated TFSs of its daughters. This can be done efficiently even with naïve unification algorithms, since there is no search involved and the unification never fails when the original derivation tree is produced by the ERG. Next, the value of the annotation is determined by the *type* of the TFS at the end of each given feature path, or **undef** in case the path was not defined in the TFS. For example, for the feature path SYNSEM.LOCAL.CAT.COMPS (the remaining list of complements needed to derive a saturated sign), the value **null** denotes an empty list, while **olist** denotes a list of only optional complements. Figure 4 shows an example of an annotated tree with one-level grandparenting (depicted by a preceding ^) and internal annotation using the top-ranked feature path (depicted by surrounding brackets [ ]).

4.4 Grammar Extraction & Probability Estimation

To extract the approximating PCFG, we need a disambiguated treebank annotated with HPSG derivations. The treebank is constructed by first parsing the input sentences with the HPSG parser, then disambiguating either manually or automatically using a MEM-based parse selection model trained on the annotated trees (Toutanova et al., 2005). The CFG symbols and production rules are extracted directly from the annotation enriched derivation trees from the treebank. Each tree node contributes to one frequency count of the corresponding CFG rule with the parent's symbol as the *left-hand side*, and the symbols

[5] Our notion of *internal* and *external* annotation is slightly different from that of Klein and Manning (2003). In our work, *internal* annotation refers to the information from the local HPSG sign, i.e. the feature structure corresponding to a specific node in the derivation tree.

[6] This technique is called *vertical Markovization* by Klein and Manning (2003).

Table 1. *Most frequently failing feature paths, used for internal annotation.*

	Feature Path
1	SYNSEM.LOCAL.CAT.HEAD
2	SYNSEM.LOCAL.CONJ
3	SYNSEM.LOCAL.AGR.PNG.PN
4	SYNSEM.LOCAL.CAT.VAL.COMPS
5	SYNSEM.LOCAL.CAT.HEAD.MOD
6	SYNSEM.LOCAL.CAT.VAL.COMPS.FIRST.OPT

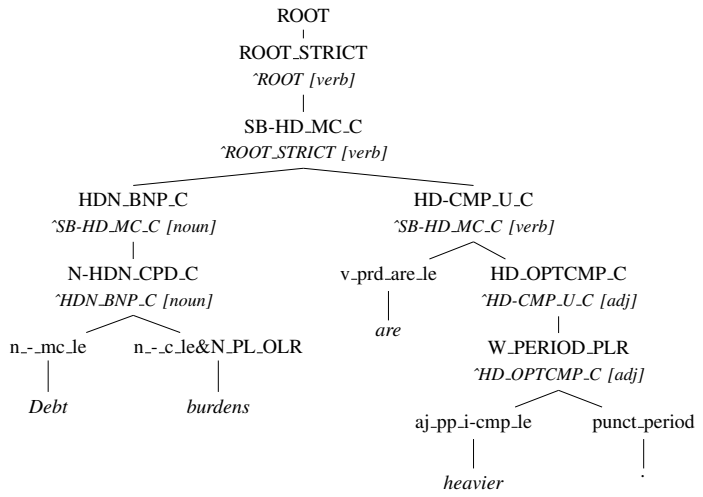

Fig. 4. Example tree with 1-level grandparent and HEAD feature path annotation.

of its daughters as the *right-hand side*. For the experiments in this article, we are solely investigating the accuracy of various approximating PCFGs, and the practical parsing efficiency is not part of the evaluation (though it is an interesting aspect which we would like to investigate in our future research; Section 3 reports speedup numbers for the LinGO forerunner of the English Resource Grammar). Hence we do not prune the CFG symbols or rules, following the practice of (Klein & Manning, 2003). The rule probability P_r is calculated with *Maximum Likelihood Estimate* (MLE) *without* smoothing.

$$P_r(A \to \beta) = P(A \to \beta|A) = \frac{\#(A \to \beta)}{\#A} \tag{1}$$

The lexical model, however, does receive smoothing for unknown word handling. More specifically, words are assigned a signature $sig(w)$ based on their capitalization, suffix, digit and other character features. We then use the MLE estimate of $P(T|sig(w))$ as a prior against which observed taggings T were taken:

$$P(T|w) = \frac{\#(T,w) + \alpha \cdot P(T|sig(w))}{\#T + \alpha} \tag{2}$$

$P(T|w)$ is then inverted to give $P(w|T)$.

The grammar extraction procedure is very efficient. The time required is linear in the size of the treebank. Even with the richest annotations (with unification operations involved), the procedure marches through thousands of trees per minute. This allows us to scale up the extraction with millions of trees.

4.5 Hierarchically Split-Merge PCFG

For comparison we also trained a hierarchically split-merge latent-variable PCFG with the Berkeley parser. The latent-variable approach (Matsuzaki, Miyao, & Tsujii, 2005; Petrov, Barrett, Thibaux, & Klein, 2006) has proven to deliver state-of-the-art parsing performance for multiple languages. The key advantage is that it automatically induces subcategories from the treebank and produces a finer grained grammar without manual intervention. In comparison to our annotation approach (which is similar to Klein and Manning (2003), see Section 4.6), this automatic process requires fewer linguistic decisions and can be easily transferred to a different language or treebank. On treebanks with coarse-grained categories (which is typical for manually annotated treebanks), this is particularly effective.

On a rich annotated treebank, however, the approach is less effective due to the fact that the subtle differences between sub-categories are explicitly annotated. In the case of PCFG approximation of an HPSG, such details in the non-terminal categories are necessary for the disambiguation of related but distinct phenomena and important for the construction of the semantics.

In our experiment, we train the split-merge latent-variable PCFG on the normalized derivation trees. The merging percentage is set to be 50% for each iteration, but calculated separately for the tags and the grammar categories due to their significant difference in size. The number of EM iterations after splitting is 50, whereas the number of EM iterations after merging is 20. The *Expectation-Maximization* training process is much more expensive than our MLE-based PCFG extraction. Also, given that the categories in the normalized derivations are already quite fine-grained (hundreds of non-terminal symbols and thousands of tags), the grammar stopped improving after only three rounds of split-merge iterations.

4.6 Unlexicalized PCFG Annotated with Linguistic Heuristics

Our approximating PCFG is in a sense related to Klein and Manning (2003), which improved the parsing accuracy of a treebank-induced PCFG by enriching the annotation with linguistically motivated heuristics. Similar to our approach, they also used an unlexicalized PCFG, enriched in part by the context in which a certain tree configuration occurs (external annotation). The difference mainly lies in the way in which further linguistic knowledge is injected. Klein and Manning (2003) relied on a set of linguistically motivated heuristics to enrich the coarse grained categories of the Penn Treebank (PTB) annotation. In our case, the complete description of a linguistic sign in a feature structure licensed by HPSG is beyond what one can practically accommodate in a PCFG. Therefore, the main effort lies in the careful selection of most relevant and effective feature paths to be added as the internal annotations. Hence, the annotation process is well motivated from the linguistic point of view, and well supported from systematic grammar engineering.

$\langle\, h_1,$
$\quad h_3$:udef_q$\langle 0:12\rangle$(BV $x_5\{$PERS *3*, NUM *pl*$\}$, RSTR h_6, BODY h_4) ,
$\quad h_7$:compound$\langle 0:12\rangle$(ARG0 $x_9\{$SF *prop*, TENSE *untensed*$\}$, ARG1 x_5, ARG2 x_8) ,
$\quad h_{10}$:udef_q$\langle 0:4\rangle$(BV x_8, RSTR h_{12}, BODY h_{11}) ,
$\quad h_{13}$:_debt_n_1$\langle 0:4\rangle$(ARG0 x_8) ,
$\quad h_7$:_burden_n_1$\langle 5:12\rangle$(ARG0 x_5) ,
$\quad h_{14}$:_heavy_a_1$\langle 17:26\rangle$(ARG0 $e_2\{$SF *prop*, TENSE *pres*$\}$, ARG1 x_5) ,
$\quad h_{14}$:comp$\langle 17:26\rangle$(ARG0 $e_{16}\{$SF *prop*$\}$, ARG1 e_2, ARG2 u_{15})
$\{\, h_{12} =_q h_{13},\, h_6 =_q h_7 \,\}\,\rangle$

Fig. 5. MRS representation of *Debt burdens are heavier.*

5 Semantics

As we stated in Section 1, we consider the recovery of structural relations involved in sentence meaning to be a primary goal of parsing. In parsing with our original HPSG, such information is built up gradually as part of the HPSG sign. This semantic information can be extracted from the full sign, in a form suitable for further use in reasoning, applications or evaluation.

5.1 Minimal Recursion Semantics

The semantic information in the original HPSG can be represented in the form of Minimal Recursion Semantics (MRS: Copestake, Flickinger, Sag, & Pollard, 2005). This framework is a flat semantic formalism that represents semantics as a bag of so-called *elementary predications* and a set of underspecified scope constraints. An elementary predication can be directly related to words in the text, or can reflect a grammatical construction, such as compounding. Each elementary predication has a relation name, a scopal label, and a distinguished variable (designated ARG0). Arguments of a predication are identified by 'bleached' ARG*n* roles (which are to be semantically interpreted for broad classes of predicates, e.g. those corresponding to unergative or causative verbs). Figure 5 shows the MRS analysis of *Debt burdens are heavier.* Here we see seven elementary predications (one per line, where the bottom line depicts scope constraints—which we will blissfully ignore for the benefits of exposition): (a) three with overt lexical referents; (b) two grammaticalized relations representing the compound and comparative constructions; and (c) two predications corresponding to construction-specific covert (underspecified, definite) quantifiers. The ARG1 and ARG2 roles of MRS elementary predications describe semantic arguments relative to the specific predicate, where the ARG1 of the adjectival predication, for example, marks the entity it modifies; conversely, BV (bound variable) is specific to predications that represent quantifiers. Entity and event variables carry properties such as number or tense.

5.2 Robust Meaning Composition

As we sketched in Section 1 above, parsing with our approximated PCFGs by itself produce the semantic information we discussed in the preceding section. In the underlying HPSG, semantic composition and construction of MRS meaning representations are accomplished straightforwardly in the parser. Lexical entries and some constructions (like N – N compounding or bare noun phrases, as seen in our running example above) intro-

duce TFS descriptions of MRS fragments, i.e. one or more partially instantiated elementary predications. In the formation of larger constituents then, unification of component parts is deployed for three core aspects of meaning composition: (a) argument linking, i.e. identifying the label or distinguished variable of one predication with an argument position in another one; (b) variable property aggregation, e.g. recording the temporal contribution of auxiliaries in an event variable; and (c) accumulation of component parts, i.e. concatenation of the multi-sets of elementary predications and scope constraints from daughter constituents.

As observed by Maxwell and Kaplan (1993), among others, unification is predominantly used as an information-combining operation here, rather than as a filter on wellformed combinations of constituents. In other words, in regular parsing unification failures are comparatively rare in the semantic sub-structures of the HPSG sign, although they can occur where grammatical constraints are encoded in terms of semantic properties, for example subject – verb agreement as the unification of number and person properties on the ARG0 and ARG1 variables of the nominal and verbal predications, respectively. All PCFG annotation schemes discussed in Section 4 above preserve sufficient information about the HPSG lexical types and constructions involved, to enable a post-parsing step of deterministic 'reconstruction' of the full HPSG sign, in principle. However, when parsing with our robust PCFGs, there is no guarantee that the resulting derivations are consistent with respect to the underlying HPSG. In the following, we sketch our approach to making such reconstruction robust to derivations that can be inconsistent with respect to the underlying HPSG, thus potentially leading to unification failures.

In a perspective on feature structures as information bundles, an inconsistent PCFG derivation contains conflicting information, say incompatible daughter categories in some syntactic construction, or discrepant number values in two semantic variables to be unified. Based on the above reflections on the division of labor between syntactic vs. semantic sub-components in HPSG signs, we conjecture that unification conflicts (when reconstructing our PCFG derivations within the original HPSG) will be far more frequent in morpho-syntactic sub-structures and relatively rare in semantic composition. Thus, we can approach robust meaning composition through a variant of the unification procedure that allows deleting (i.e. ignoring) some conflicting information, where appropriate—so-called *default unification*. In doing so, our abstract goal will be to maximize the quality of semantic information, while keeping to the deterministic nature of the post-parsing reconstruction of PCFG derivations.

A number of different approaches to default unification (of either untyped or typed feature structures, and often with varying time complexity) have been proposed over the past three decades; a relatively recent overview of some of the more influential proposals is provided by Ninomiya, Matsuzaki, Shimizu, and Nakagawa (2011, Section 4). Key dimensions of variation here are (a) which types of information in feature structures can be 'relaxed' to make graph unification succeed (node labels, outgoing arcs, or reentrancies—or combinations of these) and (b) whether or not to preserve commutativity, i.e. order-independence of the result of default unification (there can, of course, occur conflicting information in multiple places while unifying two structures, in which cases the order of relaxation decisions can effect the outcome).

For conceptual and computational simplicity, we adopt a variant of the procedure sug-

gested in Section 4.1 of Ninomiya et al. (2011). As a failure occurs in the unification of two (sub-)graph nodes, the node label (i.e. type of the corresponding feature structure) will be determined solely by the *more specific* of the two sub-graphs; unification then recursively proceeds on the intersection of outgoing arcs that are appropriate for the result type; reentrancies are never relaxed (explicitly). This procedure maintains the almost-linear time complexity of regular unification, though it is not order-independent. To locally determine which of two graph nodes is more specific (i.e. 'richer' in information), we apply two heuristics: (a) comparing the count of (cycle-free) transitive sub-nodes, which we directly interpret as an indicator of specificity; and (b) comparing the count of sub-types of the two conflicting local types (i.e. node labels), where we view a larger number of sub-types as an indicator of greater uncertainty.[7]

As such (and unlike some other strategies), our approach to robust unification will always succeed, although it could in theory do so by discarding the complete information of one of the two input feature structures. Recall from above that (parts of) our HPSG feature structures serve as *descriptions* of logical forms in the MRS framework. MRS extraction from robust unification results, thus, could in principle fail (e.g. owed to a cyclic sub-structure introduced by default unification) or result in partial or highly underspecified logical forms where unification failures occurred in semantic composition, which we internally partitioned into sub-tasks (a) to (c) above. In practice, it appears that sub-task (c)—the monotonic accumulation of bags of elementary predications and scope constraints—is hardly affected by conflicting feature structure information. Thus, partiality of resulting MRSs does not present a practical problem in our work, whereas of course unification failures in sub-tasks (a) and (b) do occur, with adverse effects on the quality of semantic outputs. In Section 7.4 below, we demonstrate how comparatively high-quality MRSs can be obtained from the combination of parsing with the approximating PCFG and robust unification and further relate the frequency of unification failures to observed output quality.

6 Semantic Evaluation

In order to compare our PCFG approximations to analyses derived from the full HPSG system, we use several parsing metrics that are in common use: the ParsEval labeled bracketing precision, recall, and F_1 (Black et al., 1991), exact match against the full syntactic derivation tree (Toutanova et al., 2005), and—since the ParsEval scores ignore the preterminal nodes—also lexical type tagging accuracy. However, one of our intentions in making use of HPSG is to facilitate a more semantic interpretation of the data, and these standard metrics only evaluate the syntactic tree.

As we show in Section 6.3, no currently available parser evaluation metric evaluates the information we are trying to recover, and so we designed our own.

[7] Obviously, a number of different strategies could be applied here, for example giving preference to information contributed by HPSG constructions (rather than daughter constituents), or to head daughters over non-head daughters. A first round of experimentation with other strategies of picking defeasible vs. indefeasible information in resolving unification failures, however, suggest that there is very little variation in the quality of resulting semantics. Further thoughts along these lines are presented in Section 9.

6.1 Design Considerations

Recall that our ultimate goal in parsing is to extract meaning from text, i.e. make available an abstract, logical-form semantics that captures all grammaticalized contributions of the linguistic signal to interpretation, and only those, while abstracting away from irrelevant surface variation. To evaluate progress towards this goal in a granular fashion, however, one needs to be able to break up the semantic information into discrete elements. For this purpose, we find it useful to distinguish three broad classes of information that contribute to meaning:

CLASS 1	core functor – argument structure, whether syntactic or semantic;
CLASS 2	predicate information, such as the lemma, word category, and sense;
CLASS 3	properties of events and entities, such as tense, number, and gender.

The ParsEval metric evaluates phrase structure, which covers none of these classes directly. Dependency-based evaluation schemes, such as Stanford Dependencies (de Marneffe & Manning, 2008) evaluate CLASS 1 surface information. The annotation used in the DepBank version of Briscoe and Carroll (2006) for parser evaluation also describes just CLASS 1 syntactic information, although the relationships are different to those that Stanford Dependencies reflect. The annotation of the original PARC700 DepBank (King, Crouch, Riezler, Dalrymple, & Kaplan, 2003) does describe all three classes of information, but again in terms of syntactic rather than semantic properties.

A common element between all the dependency types above is the use of grammatical relations to describe CLASS 1 information. That is, the dependencies are usually labels like SUBJ, OBJ, MOD, etc. While these grammatical functions allow one to describe the surface (grammatical) structure, they do not make the underlying 'deep' (logical) structure explicit. This deep structure which captures semantic rather than syntactic arguments and can be seen in resources such as the Prague Dependency Treebank (Böhmová, Hajič, Hajičová, & Hladká, 2003) and the Redwoods Treebank (Oepen, Flickinger, Toutanova, & Manning, 2004) is what we are most interested in evaluating. Using this semantic argument structure for parser evaluation not only gets closer to the actual sentence meaning that we are trying to extract, but is potentially more general, as there tends to be wider agreement on semantic arguments than on specific syntactic analyses, as, for example, whether the main verb depends on the auxiliary, or vice versa.

At the same time, we wish to focus *parser* evaluation on information determined solely by *grammatical* analysis, i.e. all contributions to interpretation by morpho-syntax, and only those. For these reasons, we consider the task of semantic role labeling (SRL) against PropBank-style target representations (Kingsbury, Palmer, & Marcus, 2002) too far removed from parser evaluation proper, i.e. a form of 'semi-extrinsic' parser evaluation. Copestake (2009) elaborates this argument, emphasizing the distinction between (semantic) *argument* labeling vs. *role* labeling and demonstrating that a "semantically consistent" labeling (in the PropBank sense) is impossible in the limit, without resorting to a very fine-grained, possibly predicate-specific inventory of role labels (an argument that in part goes back to Dowty, 1991).

Furthermore, gold-standard PropBank-style target representations are available only for a small subset of pertinent semantic predicate–argument relations and limited to a

few relatively coherent domains and genres, distinctly different from our more diverse training and test data (see Section 7 below). With very few exceptions, PropBank has hardly been used for semantic parser evaluation (as contrasted with SRL), and as in any 'framework-independent' parser evaluation—syntactic or semantic—the conversion from parser-internal to external target representations would inevitably introduce a great deal of uncertainty (Crouch, Kaplan, King, & Riezler, 2002; Clark & Curran, 2007). Therefore, the focus of empirical validation of our robust PCFG parser and meaning construction will be by comparison to the type of semantics most commonly used in the HPSG universe, and specifically the ERG as our 'source' grammar.

6.2 EDM: Elementary Dependency Match

In addition to our focus on semantic information, we considered two other requirements for an effective parser evaluation metric. It should be:

1. understandable not just by parser developers, but also non-expert users;
2. configurable to suit the level of detail required for a particular scenario.

The metric we have devised to satisfy these requirements is Elementary Dependency Match (EDM), based on so-called Elementary Dependency Structures (EDS), a variable-free reduction of MRS developed by Oepen and Lønning (2006).[8] In a nutshell, the full MRS in Figure 5 can be reduced to a semantic dependency graph (EDS), where elementary predications induce the nodes (which then are labelled by predicate symbols, e.g. _heavy_a_1⟨17:26⟩), and semantic argumenthood is encoded by arcs labelled with the underlying roles, e.g. BV, ARG1, etc. Here MRS variables are mapped onto nodes of the dependency graph on the basis of the distinguished variable notion (ARG0, see above), and in cases of a one-to-many mapping (which can arise with scopal arguments, but not in our running example) a few simple disambiguating heuristics are applied.

In our work, we use sub-string character spans (e.g. ⟨5:12⟩) to identify nodes in the dependency graph, to facilitate alignment of corresponding elements across distinct analyses, which in large parts parallels common practise in parser evaluation based on word-to-word dependencies. Note, however, that our MRSs can lexically decompose the meaning contribution of individual words (as observed in the two elementary predications in Figure 5 corresponding to the account of the comparative *heavier*) and can further contain predications introduced by grammatical constructions, e.g. the two-place compound relation corresponding to the phrase *debt burdens*. For these reasons, our scheme of identifying dependency nodes merely by sub-string spans does not establish a unique labelling of the graph nodes. Thus, in our running example the span ⟨17:26⟩ corresponds to both predications involved in the comparative, which will become indistinguishable once the EDS dependency graph is further reduced into basic triples for evaluation purposes. These reductions not only make it easier for us to compare across parsers or analyses but also facil-

[8] In more recent work, Copestake (2009) shows how essentially the same reduction can be augmented with information about the underspecified scope hierarchy, so as to yield so-called Dependency MRS (which unlike EDS facilitates bi-directional conversion from and to the original MRS).

itate the separation of CLASS 1 vs. CLASS 2 information. Our EDM metric hence consists of three triple types which align with the three information classes:

NAMEs:	$span_i$	PRED	$relation_i$
ARGs:	$span_i$	$role_j$	$span_k$
PROPs:	$span_i$	$property_j$	$value_j$

In these forms, $relation$ is the predicate name of an elementary predication from the MRS, $role$ is an argument label such as ARG1, $property$ refers to an attribute such as TENSE or NUM, and $value$ is an appropriate instantiation for the respective property. Figure 6 shows the triples produced for the MRS in Figure 5.

NAME			**ARG**			**PROP**		
⟨0:4⟩	PRED	udef_q	⋆	ROOT	⟨17:26⟩	⟨5:12⟩	NUM	*pl*
⟨0:4⟩	PRED	_debt_n_1	⟨0:4⟩	BV	⟨0:4⟩	⟨0:12⟩	SF	*prop*
⟨5:12⟩	PRED	_burden_n_1	⟨0:12⟩	BV	⟨5:12⟩	⟨0:12⟩	TENSE	*untensed*
⟨0:12⟩	PRED	udef_q	⟨0:12⟩	ARG1	⟨5:12⟩	⟨17:26⟩	SF	*prop*
⟨0:12⟩	PRED	compound	⟨0:12⟩	ARG2	⟨0:4⟩	⟨17:26⟩	SF	*prop*
⟨17:26⟩	PRED	_heavy_a_1	⟨17:26⟩	ARG1	⟨5:12⟩	⟨17:26⟩	TENSE	*pres*
⟨17:26⟩	PRED	comp	⟨17:26⟩	ARG1	⟨17:26⟩			

Fig. 6. EDM triples for *Debt burdens are heavier.*

During evaluation, we can compare the triples from the gold standard analysis with those ranked top by the parser, and calculate precision, recall, and F_1 scores across all triples, as well as across the three separate triple types (NAME, ARG, and PROP).

6.3 Comparison to Other Frameworks and Related Metrics

To see how the EDM metric relates to other dependency-based parser evaluation metrics (and to give at least a bit of an indication of how our parser outputs compare to other analysis frameworks), we show how the same example sentence is represented in four commonly used metrics, in the context of the information classes outlined above. Figure 7 shows the analysis as represented by Stanford Dependencies and Briscoe and Carroll (2006) Grammatical Relations. Both representations present the basic CLASS 1 syntactic information, although they use different analyses of the predication copula construction.

root(ROOT-0, heavier-4)	
nsubj(heavier-4, burdens-2)	(ncsubj are burdens _)
nn(burdens-2, Debt-1)	(ncmod _ burdens Debt)
cop(heavier-4, are-3)	(xcomp _ are heavier)

Fig. 7. Stanford Dependencies (left) and Grammatical Relations (right) for *Debt burdens are heavier.*

CCG dependencies (see Figure 8) combine syntactic CLASS 1 information with CLASS 2 information, since each dependency includes the CCG category (or supertag) of the head

Debt-0	N/**N**	burdens-1
are-2	(S[dcl]**NP**)/(S[adj]\NP)	burdens-1
are-2	(S[dcl]\NP)/(**S[adj]****NP**)	heavier-3
heavier-3	S[adj]**NP**	burdens-1

Fig. 8. CCG dependencies for *Debt burdens are heavier.*

word. Each dependant is considered a separate evaluation element, with the part of the category it fills indicated here by bold face.

Figure 9 shows the PARC700 DepBank annotation for our example. As with the EDM triples, information pertaining to properties of events and entities is represented separately. Some information about the predicates is captured in the lemmatization (i.e. *are* as 'be') or *heavier* as 'heavy'), but, as with the CCG dependencies, this level of the analysis is rolled in with the syntactic analysis.

subj(be-0, burden-2)	stmt_type(be-0, declarative)
xcomp(be-0, heavy-1)	tense(be-0, pres)
subj(heavy-1, burden-2)	num(burden-2, pl)
mod(burden-2, debt-6)	num(debt-6, sg)
adegree(heavy-1, comparative)	
adjunct(heavy-1, null-3)	

Fig. 9. PARC 700 triples for *Debt burdens are heavier.*

Of the many parser evaluation metrics are already in use then, none of them are adequate for our purposes in facilitating semantic interpretation, hence the need to devise our own. Rimell and Clark (2008) discuss some of the issues in designing an evaluation metric, and while not all of their points are relevant—since we are not trying to evaluate across frameworks—their discussion of informativeness and compactness is pertinent. We have made the decision here to be as informative as possible, and, rather than aiming for compactness, are deliberately separating different elements of the analysis into individual evaluation elements. This allows us to be comprehensive in what we present, but also configurable to the evaluation scenario, since information can be included for evaluation when appropriate, and ignored otherwise.

In our current experiments, we are interested in both CLASS 1 and CLASS 3 information but, with unlexicalized PCFGs, we do not attempt to recover CLASS 2 information, and hence do not evaluate it.

7 End-to-End Evaluation & Results

7.1 Grammar & Data

We use the 1010 release of the *English Resource Grammar* (ERG) for the approximation experiments. This version of the ERG contains a total of 200 syntactic constructional rules, and 50 lexical rules (that is derivational, inflectional, or punctuation ones). 145 of the 200 syntactic rules are binary, while the remaining 55 are unary. All lexical rules are unary.

In addition, the grammar contains a hand-compiled lexicon instantiating around 1000 leaf lexical types with over 35K base-form entries.

Several large treebanks have been developed with the ERG. Unlike traditional, manually annotated treebanks, these were developed as Redwoods-style 'dynamic' treebanks (Oepen, Toutanova, et al., 2002). Sentences from the corpus are first parsed by the ERG, and then manually disambiguated (mostly by the grammarian himself). For the experiments in this article, we use the manually disambiguated WeScience Treebank (Ytrestøl, Flickinger, & Oepen, 2009), which currently contains a total of over 11K sentences from a selection of Wikipedia articles in the domain of Natural Language Processing, with an average length of 18 tokens per sentence, pre-processed to strip irrelevant markup, and divided into 13 roughly equal-sized sections. Of all sentences in WeScience, about 78% received exactly one 'gold' analysis during treebanking. The rest either failed to be parsed by the ERG, or there was no single acceptable reading to the annotator. We will only use the subset of sentences with a *gold* parse for the experiment. More specifically we reserve section *WS12* for development and section *WS13* for final testing. Sections *WS01* to *WS11* contain a total of 7,636 'gold' trees, which are used for training.

Apart from WeScience, we also use the large-scale automatically parsed WikiWoods Treecache (Flickinger, Oepen, & Ytrestøl, 2010). WikiWoods contains about 55M English sentences extracted from the full English Wikipedia, organized in some 13,000 segments. The corpus is parsed with the 1010 version of the ERG, and automatically disambiguated using a Maximum Entropy model trained on the manually disambiguated trees in WeScience. Only the one top-ranked reading is preserved in the construction of WikiWoods. Since the correctness of ERG analysis is unchecked in the automatically parsed data, this dataset must be considered potentially noisy. The total number of ERG derivations available for training is about 48M.

7.2 The Parser

The approximating PCFG tends to grow huge when rich annotations and large corpora are used. For efficient application of the resulting grammar, we implemented a CKY-style parser with bit-vector-based algorithm like the one proposed by Schmid (2004). The algorithm shows its strength in extensibility for grammars with millions of rules and hundreds of thousands of non-terminal symbols.

A slight deviation of our implementation from the BitPar algorithm is that, after constructing the bit-vector-based recognition chart, we do not apply the top-down filtering routine before calculating the Viterbi probabilities. Practice shows that in our case the recognition chart is normally sparse, and the filtering routine itself costs more time than what it saves from the additional calculations in the Viterbi step.

For correctness checking, we reproduced the unlexicalized PCFG parsing accuracy reported by Klein and Manning (2003) on PTB with our bit-vector parser while achieving better efficiency (in both training and testing) than the Stanford Parser. Even though our parser is implemented in Java (for better cross-platform compatibility), the low-level bit-vector-based operations make our system competitive even in comparison to the BitPar implemented in C++.

As mentioned earlier, we do not (yet) prune PCFG rules during grammar acquisition or

parsing. For lexical look-up, we allow the lexical model to propose multiple tags (cut by a certain probability threshold). In case a full parse is not found, a partial parsing model tries to recover fragmented analysis according to the Viterbi probabilities of the constituents. With careful design of the PCFG and sufficient training data, the parser normally delivers close to full parsing coverage even without the fragmented partial parsing mode.

7.3 Experimental Setup

For the evaluation of our approximating PCFGs, we compare the top-ranked parses produced by the PCFG with the manually disambiguated gold trees in *WS13*. We assume that parser inputs have been pre-tokenized (according to ERG conventions) but not tagged. All comparisons are against the original derivations.[9] Several different training sets are used.

WS contains 7636 *gold* trees from sections *WS01 – WS11* of the WeScience Treebank, which also served as the training data for the ERG-native MEM parse selection model. This dataset is too small to acquire high coverage PCFGs with heavy annotations. Therefore, only PCFG(0) and PCFG(FP1) results are reported here.

WW000 contains 85,553 automatically parsed and disambiguated trees from WikiWoods (all segments with 000 as suffix). This is less than 0.2% of the entire collection, but close to the limit for the latent-variable PCFG training with the Berkeley Parser.

WW00 contains about 482K sentences (all segments with 00 as suffix), roughly 1% of the entire WikiWoods. We were able to successfully train PCFGs with relatively rich annotations.

WW contains the complete WikiWoods with ∼48M parsed trees. With feature path annotations, the training of the models takes too long. Also, excessive annotation makes it difficult to parse with the resulting huge grammar. We stop at two levels of grandparent annotation, obtaining a PCFG with almost 4M rules and over 128K non-terminal symbols.

7.4 Empirical Results

Tables 2 and 3 summarize the results of the accuracy evaluation. All results are reported on the 785 'gold' trees from section *WS13* of WeScience. PET is the accuracy of the HPSG parser disambiguation model given the candidate parse forest and the gold tokenization. PCFG(0) is the unannotated PCFG read off the bare (normalized) derivation trees. PCFG(GP_m,FP_n) is the annotated PCFG model with m-levels of grandparents and n feature paths. And PCFG-LA(SM3) is the latent-variable PCFG after three split-merge training iterations with the Berkeley Parser.

The discriminative parse disambiguation model of PET serves as the upper bound for

[9] Full details of our evaluation setup—gold standard data, parser outputs and scores for all configurations, pointers to the open-source resources involved, and all software—are available for reproducibility and follow-up experimentation at http://svn.delph-in.net/snug/edm/.

Table 2. *Syntactic evaluation of parsing accuracy on WS13 with various models and training sets. Reported are grammar size (#Rule, #NT, #T); ParsEval precision (P), recall (R), and F_1; as well as exact match ratio (EX) on the original derivation tree and tagging accuracy (TA) on its lexical types.*

		#Rule	#NT	#T	P	R	F_1	EX	TA
ws	PET	-	-	-	87.1	87.1	**87.1**	48.79	96.5
	PCFG(0)	10,251	208	1,152	64.8	59.1	61.8	18.09	83.3
	PCFG(FP1)	12,178	669	1,152	71.7	63.3	67.2	18.73	82.4
ww000	PCFG(0)	25,859	236	1,799	61.3	58.2	59.7	13.76	83.9
	PCFG(GP1)	64,043	3,983	1,799	73.5	70.7	**72.1**	20.25	85.9
	PCFG-LA(SM3)	*	*	*	74.4	69.4	71.8	1.91	88.0
ww00	PCFG(0)	61,426	247	2,546	64.5	62.1	63.3	16.56	87.8
	PCFG(GP1)	187,852	5,828	2,546	78.5	77.9	78.2	25.35	91.5
	PCFG(GP1,FP4)	271,956	16,731	2,546	81.6	80.7	81.2	29.04	92.2
	PCFG(GP1,FP5)	319,511	21,414	2,546	82.0	81.2	**81.6**	28.54	92.4
	PCFG(GP1,FP6)	320,630	21,694	2,546	81.9	81.1	81.5	28.41	92.4
	PCFG(GP2)	489,890	45,658	2,546	80.2	79.8	80.0	28.92	91.6
	PCFG(GP2,FP2)	559,006	66,218	2,546	81.1	80.3	80.7	32.10	91.8
ww	PCFG(GP1)	1,007,563	8,852	4,472	81.3	80.6	80.9	29.43	92.5
	PCFG(GP2)	3,952,821	128,822	4,472	85.0	84.8	**84.9**	37.45	93.6

our grammar approximation task. Despite our non-terminal label set being an order of magnitude above that of the PTB, the ParsEval F_1 is comparable to that from state-of-the-art PCFG parsers trained on twice as much data (McClosky, Charniak, & Johnson, 2006b).[10] With the same small amount of training data, the baseline PCFG without annotation achieved ParsEval F_1 of merely 61.8. With one feature path annotation (SYNSEM.LOCAL.CAT.HEAD value), this improved significantly to 67.2, but the size of the ws data set was insufficient to successfully add grandparent annotation.

Using the larger WW000, the performance of the baseline PCFG decreases by ∼2% ParsEval F_1, most likely due to the noise introduced by the automatic disambiguation in WikiWoods. However, the larger training set enables 1-level grandparent annotation, which brings ParsEval F_1 up to 72.1. The latent-variable PCFG achieved similar syntactic accuracy, delivering the best F_1 at 71.8 after three split-merge iterations. Error analysis shows that its low exact match ratio is mostly due to the inaccurate prediction of the preterminal chain of lexical/morphological rules. With the 85K training sentences, the learning curve of the Berkeley parser has already flatten out at this point, and we were unsuccessful in further scaling up the training set.

The WW00 training set provides sufficient noisy data for the PCFG(0) to outperform the one trained with the 7636 gold trees, but the real bonus comes from having enough data

[10] With less than 20% of the training data, we are still not quite at the top levels seen on §23 of the PTB when trained on §2–21.

Table 3. *EDM-based semantic evaluation of parsing accuracy on WS13 with various models and training sets. FS consistency is the proportion of fully wellformed PCFG derivations (when reconstructed in the HPSG universe), i.e. ones that require no recourse to robust feature structure unification.*

		FS Consistency	EDM_A			EDM_P		
			P	R	F_1	P	R	F_1
WS	PET	**100**%	86.6	86.4	**86.5**	94.1	93.9	**94.0**
	PCFG(0)	22.9%	67.5	54.5	60.3	84.1	76.6	80.2
	PCFG(FP1)	25.9%	73.2	61.5	66.8	86.8	75.8	80.9
WW000	PCFG(0)	17.0%	63.9	51.1	56.8	81.6	75.4	78.4
	PCFG(GP1)	**31.5**%	73.9	65.6	69.5	87.1	79.0	82.8
	PCFG-LA(SM3)	19.6%	76.0	70.6	**73.2**	87.7	82.9	**85.2**
WW00	PCFG(0)	19.1%	67.8	55.7	61.2	84.1	79.0	81.5
	PCFG(GP1)	37.8%	79.5	74.2	76.8	90.3	85.8	88.0
	PCFG(GP1,FP4)	44.5%	81.5	79.9	80.7	91.1	89.8	90.4
	PCFG(GP1,FP5)	45.4%	81.6	80.0	**80.8**	91.2	90.0	**90.6**
	PCFG(GP1,FP6)	45.4%	81.6	80.0	80.8	91.1	89.9	90.5
	PCFG(GP2)	46.0%	81.2	75.9	78.5	90.9	86.4	88.6
	PCFG(GP2,FP2)	**51.2**%	81.5	79.1	80.3	91.3	89.3	90.3
WW	PCFG(GP1)	41.2%	80.7	79.2	79.9	91.0	90.5	90.8
	PCFG(GP2)	**55.4**%	84.6	83.8	**84.2**	92.9	92.6	**92.8**

to learn reliable statistics over annotated trees. With the mixture of 1-level of grandparent and some feature path annotations, ParsEval F_1 reached over 80. The best performance on WW00 is achieved with PCFG(GP1,FP5). 2-levels of grandparents alone outperforms 1-level of grandparent, but the grammar quickly reaches its size limit on this training set and starts to lose coverage when more feature path annotations are added.

Finally, with the complete WikiWoods, both PCFG(GP1) and PCFG(GP2) improved further, with PCFG(GP2) reaching our highest ParsEval F_1 at 84.9, only 2.2 points below the upper bound with the discriminative disambiguation model and working on the full HPSG parse forest produced by the grammar. We attribute such competitive results partially to self-training effects on the huge corpus (see below for further discussion). The gap on the exact match ratio is larger, though, possibly due to the fact that the objective function of the discriminative disambiguation model was, unlike the PCFGs, optimized on complete parse match instead of individual constituents.

Focussing now on the semantic evaluation, the EDM scores for the most part reflect the trends in ParsEval. The EDM_P scores are relatively high, owing to the strong interdependence of the properties, while the EDM_A, as mentioned above, could be considered similar to labelled accuracy scores (LAS) for dependency parsers, except that the dependencies are semantic rather than syntactic, and are comparable to state-of-the-art dependency scores. Looking more closely, we see a few extra trends when examining the EDM results. Based just on the scores, it appears that the feature path annotations have a

greater positive effect on the semantic analysis than the syntactic (comparing, for instance, PCFG(GP1) and PCFG(GP1.FP4) or PCFG(GP2) and PCFG(GP2.FP2)). We also see that mistakes made by the Berkeley parser have less effect on the semantic analysis, meaning it produces the best EDM results of parsers trained on the WW000 data set.

The EDM triples also allow us to examine what type of information the parsers are getting wrong. We looked at which frequent (seen greater than 100 times) argument relations each of the parsers found the most difficult. For the original HPSG parser, we see things well known to cause problems for parsers, prepositional phrase attachment and conjunctions, as well compounds, particularly in compound names. The best performing PCFG approximation, PCFG(GP2) trained on the full WikiWoods, also had problems with PP attachment and conjunctions, however there were two classes of relations with very different performance. In common with all the PCFG approximations, the top performer was very poor at detecting ARG3 relations for verbs. This is a place where the explicit subcategorisation in the HPSG lexicon provides a strong benefit to our original parser. On the other hand, the approximating PCFG was much better at determining the structure of compound names than the original parser. This probably points to the fact that these relationships are largely driven by collocation, rather than syntax, and suggests that a focussed statistical approach would probably improve this problematic area in the PET parser. The other notable trend in the approximating PCFGs trained on less than the full WikiWoods is low accuracy in distinguishing between prepositions with a meaningful semantics, and empty particles. Again, the explicit subcategorisations, and also the constraints defined in the HPSG make these much easier for our original parser to detect, although with the full WikiWoods as training data, the approximating PCFG can also learn to correctly disambiguate these.

The EDM $_P$ triples were less informative in pinpointing differences between the parsers, although effects of the prepositions also showed up in this analysis. The major pattern showing here was that, unlike all the other parsers, the Berkeley parser was less accurate on nominal properties (person and number) than on verbal properties. This may be due to the above observation that the PCFG-LA parser was somewhat inaccurate over the preterminal chain of lexical/morphological rules, which could affect these properties.

As a reflection on the effects of robust unification, observe how higher feature structure consistency rates do not necessarily align with better EDM scores. We interpret this asymmetry as evidence for our stipulation that minor syntactic errors do not prevent correct meaning composition, which is further supported by observation of our best-performing approximation reaching near-competitive EDM scores despite its comparatively low proportion of fully consistent PCFG derivations—which is just above one in two.

Finally, recall that empirical results so far were against only about 80% of our test corpus, i.e. the 785 utterances in *WS13* that (a) the ERG can parse and where (b) human diambiguation found a correct analysis among the top-500 parses.[11] For the remaining 195 test utterances in *WS13*, our top-performing PCFG parser makes available syntactic analyses for all but three, and via robust unification we can derive an MRS meaning

[11] Arguably, this setup introduces a bias in favor of the HPSG system into our contrastive evaluation setup, seeing as the comparison so far was exclusively focussed on inputs that the ERG can handle well. In this light, reaching near-comparable semantic dependency accuracies, in our view, constitutes an even stronger result than one might conclude from Table 3 in isolation.

representation for all of these. These results confirm the greatly increased robustness (to extragrammatical inputs) of our parser, compared to the original HPSG system.

Lacking gold-standard target representations for the extra parses, in the following we seek to contrast the two sets of parses in terms of structural properties. The most important difference to note between the 785 in-scope (for the ERG) and 192 out-of-scope parses is their average length: with about 28 leaf nodes per derivation, the out-of-scope trees are exactly double the average length of the set of in-scope parses. Its ability to successfully analyze much longer inputs further corroborates the robustness gains in our setup.

For further quantitative comparison of in- vs. out-of-scope analyses, we subdivided the in-scope parses into four random subsets, each comparable in size to the set of out-of-scope parses. In terms of tree shape (and with the caveat of averaging over rather small samples), out-of-scope syntactic analyses exhibit a very slightly higher depth, at 3.07 tree nodes per leaf, compared to 2.89–2.95 for the in-scope contrast sets. In terms of MRS wellformedness, out-of-scope analyses show a moderately higher proportion of 'fragmentation', with 1.04 instances of partial graph connectivity among elementary predications per 100 input words, compared to between 0.74 and 1.01 for the in-scope samples. Comparing EDM_A triples per input word, again we see comparable average statistics: out-of-scope meaning representations score at 1.61, contrasting with 1.48–1.59 for in-scope MRSs. While they cannot directly inform us about the downstream utility of the robust, out-of-scope analyses gained in the PCFG universe, these crude structural comparisons at least suggest a comforting degree of similarity to our 785 validated, in-scope results.

8 Discussion

It is important to note that the grammar approximation task we take on in this article is different from traditional treebank-based parsing. Although the accuracy evaluations for both tasks are performed on a fixed set of *gold* trees (out of practical necessity in assessing relative parse quality), in the grammar approximation task we have access to the theoretically infinite pool of training data automatically generated by the original grammar. Some complex grammar extraction algorithms which work fine on a small training corpus fail to scale up to handle millions of trees. On the other hand, our MLE-based PCFG extraction shows its advantage in extensibility.

Our approach of training a PCFG with automatically parsed corpora is in some sense similar to recent self-training work in semi-supervised parsing (McClosky, Charniak, & Johnson, 2006a). However, instead of parsing the unlabeled data with the PCFG directly, we rely on the HPSG and its disambiguation model in constructing PCFG training material. The comparatively highly constrained ERG analyses on unseen data allow us to obtain (seemingly) good-quality trees. And the potential penalty of introducing noisy data is quickly compensated for by the virtually unlimited volume of available training data.

Our approximating PCFGs are much less constrained than the ERG. From a linguistic point of view, it would be difficult (to say the least) to analytically interpret the huge set of PCFG rules. And unlike the ERG, the PCFG is unsuited for making sharp grammaticality judgments, say in a computer-assisted language learning environment. However, for the task of parsing unrestricted running text, our approximating PCFGs have the key advantage of being robust (in the sense of making available analyses for *all* inputs), flexible, and

maintaining a cubic worst-case bound on parsing time complexity. Our results demonstrate that one can choose various combinations of annotations to adequately balance efficiency, accuracy, and coverage—thus facilitating adaptation to specific requirements in individual applications. Although the experiments reported in Section 7 above are only testing on sentences which the ERG actually can parse, we have also applied our PCFGs on the remaining ∼20% text that are out-of-scope for the ERG, and obtained close to full parsing coverage (with less than 1% remaining failures). Although derivation trees constructed by the PCFG do not guarantee a consistent HPSG analysis, they provide an approximate prediction of what the syntacto-semantic structure should look like. Using robust unification, we can thus obtain a largely wellformed semantic structure with the guidance of the approximating PCFG for unrestricted running text.

Looking at the specific annotation strategies, we compared the internal annotations with the external ones. Experimental results show that when grandparent annotation is added, the grammar size grows quickly. On a huge training set, this is rewarded with significant accuracy gains. On our smaller training sets though, 'over-annotating' with grandparents results in a decrease in accuracy due to data sparseness. Instead, annotating with feature path information increases the grammar size more moderately, allowing one to flexibly seek to approach the optimal granularity of the PCFG.[12]

In comparison to the linguistic annotations used by Klein and Manning (2003) for PTB parsing, our annotations are less language- or treebank-specific (with the exception of certain normalizations). This is due to the fact that ERG analyses are relatively fine-grained in treating various linguistic constructions: the most relevant annotations can be gathered from either the grandparents or the internal feature structure. Such a general design allows us to experiment with other 'deep' HPSG parsers (and other grammatical frameworks) in the future.

For the clarity of the experiment, we have chosen not to do constructional pruning or smoothing. We thus focused our evaluation mostly on parsing accuracy. This leaves much room for future investigation. For instance, we observe that a large portion of the grammar rules have very low frequency counts and almost no impact on the parsing accuracy. On the other hand, even with the simple PCFG(GP1), after training with 45 million sentences, the grammar continues to pick up new rules at a rate of one rule per 160 sentences. Most of these new rules are the combination of low frequency supertags.

Regarding our proposal for granular semantic evaluation of MRS meaning representations via EDM, we find the results with the semantic evaluation measures to be largely consistent with our syntactic derivation tree-based evaluation. In cases where differences were observed, e.g. outputs from parsers adopting different approaches, we find the EDM

[12] In a related and contemporaneous study, Evensberget (2011) observe that internal annotation with much larger sets of feature paths can be beneficial too, under certain circumstances. Using up to 33 paths, hand-picked from the standard ERG quick-check vector, and a subset of WikiWoods comparable in size to our WW00, he reports a ParsEval F_1 score suggesting at least competitive accuracy to our results when training on WW00 (though not our best-performing WW parser). These experiments are not fully comparable, however, due to subtle difference in input preprocessing. Furthermore, owed to the greater cost of internal annotation, Evensberget (2011) does not provide empirical results using internal annotation on the full WikiWoods.

scores to be more informative regarding the performance on the consequent semantic compositions.

Last but not least, given the promising parsing accuracy of the approximating PCFG and robustness of meaning composition by default unification, we believe it is worth reconsidering the role of the hand-written grammars in deep linguistic processing. In the past, manual grammar engineering has been taking on the dual role of offering concise and accurate linguistic description on the one hand, while attending to the efficiency and robustness in parsing, on the other hand. These conflicting goals have hindered the development of practical large-scale linguistic grammars. Techniques as the ones presented in this article suggest a way of potentially liberating grammarians from concerns for robustness to out-of-scope inputs and computational processing cost.

9 Conclusion and Outlook

In conclusion, we presented a corpus-driven approach to approximate a large-scale hand-written HPSG with a PCFG for robust and accurate parsing. Different annotations are used to enrich the derivation trees. And with the 48M sentence from the English Wikipedia automatically parsed and disambiguated by the ERG, a MLE-based PCFG achieved ParsEval F_1 of 84.9, close to the performance of the discriminative parse selection model of the original grammar (which has access to the candidate HPSG parse forest). The obvious robustness and potential efficiency advantages of the approximating PCFG suggest a broad range of promising applications in semantics-oriented parsing.

While in this article we just touched on the topic of robust semantic composition via default unification, we see a range of alternative solutions to the interesting task of reconciling conflicting information. Apart from the *default* strategy we have taken in this article, one could also keep the inconsistency either by *upward* or *downward* inference (and potential expansion) of the type inheritance hierarchy, or by producing *non-deterministic* unification results that explicitly encode the uncertainty. Hence, inconsistencies between the fragments of linguistic analysis combined during parsing need not necessarily be resolved on the spot (as we do in the present study). Instead, disambiguation in the space of relaxations addressing conflicting information could be delayed until sufficient (and potentially extra-linguistic) evidence becomes available. The actual implementation and evaluation of various strategies remains to be investigated in the future.

References

Abney, S. (1997). Stochastic attribute-value grammars. *Computational Linguistics*, *23*, 597–618.

Black, E., Abney, S., Flickinger, D., Gdaniec, C., Grishman, R., Harrison, P., ... Strzalkowski, T. (1991). A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the Workshop on Speech and Natural Language* (pp. 306–311). Pacific Grove, USA.

Böhmová, A., Hajič, J., Hajičová, E., & Hladká, B. (2003). The Prague Dependency Treebank: A three level annotation scenario. In A. Abeill (Ed.), *Treebanks: building and using parsed corpora.* Springer.

Briscoe, T., & Carroll, J. (2006). Evaluating the accuracy of an unlexicalised statistical parser on the PARC DepBank. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics and the 21st International Conference on Computational Linguistics (COLING)* (pp. 41–48). Sydney, Australia.

Cahill, A., Burke, M., O'Donovan, R., Van Genabith, J., & Way, A. (2004). Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based LFG approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics* (pp. 319–326). Barcelona, Spain.

Callmeier, U. (2000). PET – a platform for experimentation with efficient HPSG processing techniques. *Journal of Natural Language Engineering*, *6*(1), 99–108.

Carpenter, B. (1992). *The logic of typed feature structures*. Cambridge, UK: Cambridge University Press.

Clark, S., & Curran, J. (2007). Formalism-independent parser evaluation with CCG and DepBank. In *Proceedings of the 45th annual meeting of the association of computational linguistics* (pp. 248–255). Prague, Czech Republic. Retrieved from `http://www.aclweb.org/anthology/P07-1032`

Copestake, A. (2002). *Implementing typed feature structure grammars*. Stanford, USA: CSLI.

Copestake, A. (2009). Invited talk: Slacker semantics: Why superficiality, dependency and avoidance of commitment can be the right way to go. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 1–9). Athens, Greece. Retrieved from `http://www.aclweb.org/anthology/E09-1001`

Copestake, A., Flickinger, D., Sag, I. A., & Pollard, C. (2005). Minimal Recursion Semantics: an introduction. *Research on Language and Computation*, *3*(4), 281–332.

Crouch, R., Kaplan, R., King, T., & Riezler, S. (2002). A comparison of evaluation metrics for a broad coverage parser. In *Proceedings of the Beyond ParsEval workshop at the 3rd International Conference on Language Resources and Evaluation.* Las Palmas, Spain.

de Marneffe, M.-C., & Manning, C. D. (2008). The Stanford typed dependencies representation. In *Proceedings of the COLING Workshop on Cross-framework and Cross-domain Parser Evaluation* (pp. 1–8). Manchester, UK.

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, *67*(3), 547–619.

Evensberget, J. B. (2011). *Context-free approximation of large unification grammars. A walk in the forest*. Msc thesis, Department of Informatics, University of Oslo, Oslo, Norway.

Flickinger, D. (2002). On building a more efficient grammar by exploiting types. In S. Oepen, D. Flickinger, J. Tsujii, & H. Uszkoreit (Eds.), *Collaborative language engineering* (pp. 1–17). CSLI Publications.

Flickinger, D., Oepen, S., & Ytrestøl, G. (2010). WikiWoods: Syntacto-semantic annotation for English Wikipedia. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC).* Valletta, Malta.

Fowler, T. A. D., & Penn, G. (2010). Accurate context-free parsing with Combinatory Categorial Grammar. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 335–344). Uppsala, Sweden.

Hockenmaier, J., & Steedman, M. (2007). CCGbank. A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, *33*(3), 355–396.

Johnson, M. (1998). Pcfg models of linguistic tree representations. *Computational Linguistics*, *24*(4), 613–632.

Kasper, W., & Krieger, H.-U. (1996). Modularizing codescriptive grammars for efficient parsing. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)* (pp. 628–633). Copenhagen, Denmark.

Kiefer, B., & Krieger, H.-U. (2000). A context-free approximation of Head-Driven Phrase Structure Grammar. In *Proceedings of the 6th international workshop on parsing technologies, iwpt 2000* (pp. 135–146). Trento, Italy.

Kiefer, B., & Krieger, H.-U. (2004). A context-free superset approximation of unification-based grammars. In *New developments in parsing technology* (pp. 229–250). Kluwer. (http://www.wkap.nl/prod/b/1-4020-2293-X)

Kiefer, B., Krieger, H.-U., Carroll, J., & Malouf, R. (1999). A bag of useful techniques for efficient and robust parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics* (pp. 473–480). Maryland, USA.

Kiefer, B., Krieger, H.-U., & Prescher, D. (2002). A novel disambiguation method for unification-based grammars using probabilistic context-free approximations. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING).* Taipei, Taiwan.

King, T. H., Crouch, R., Riezler, S., Dalrymple, M., & Kaplan, R. M. (2003). The PARC 700 Dependency Bank. In *Proceedings of the LINC-03 Workshop* (pp. 1–8). Budapest, Hungary.

Kingsbury, P., Palmer, M., & Marcus, M. (2002). Adding semantic annotation to the Penn treebank. In *Proceedings of the Human Language Technology Conference* (pp. 252–256). San Diego, USA.

Klein, D., & Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics* (pp. 423–430). Sapporo, Japan.

Knuth, D. E. (1968). Semantics of context-free languages. *Mathematical Systems Theory*, *2*(2), 127-145.

Krieger, H.-U. (2004). A corpus-driven context-free approximation of head-driven phrase structure grammar. In G. Paliouras & Y. Sakakibara (Eds.), *Grammatical Inference: Algorithms and Applications. 7th International Colloquium, ICGI 2004* (pp. 199–210). Springer.

Krieger, H.-U. (2006). From UBGs to CFGs: A practical corpus-driven approach. *Natural Language Engineering*, *13*(4), 317–351.

Malouf, R., & van Noord, G. (2004). Wide coverage parsing with stochastic attribute value grammars. In *Proceedings of the IJCNLP Workshop: Beyond shallow analyses — formalisms and statistical modeling for deep analyses.* Hainan Island, China.

Matsuzaki, T., Miyao, Y., & Tsujii, J. (2005). Probabilistic CFG with latent annotations. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05)* (pp. 75–82). Ann Arbor, MI, USA.

Matsuzaki, T., Miyao, Y., & Tsujii, J. (2007). Efficient HPSG parsing with supertagging and CFG-filtering. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 1671–1676). Hyderabad, India.

Maxwell III, J. T., & Kaplan, R. M. (1993). The interface between phrasal and functional constraints. *Computational Linguistics*, *19*(4), 571–590.

McClosky, D., Charniak, E., & Johnson, M. (2006a). Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 152–159). New York, USA.

McClosky, D., Charniak, E., & Johnson, M. (2006b). Reranking and self-training for parser adaptation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics and the 21st International Conference on Computational Linguistics (COLING)* (pp. 337–344). Sydney, Australia.

Miyao, Y., Ninomiya, T., & Tsujii, J. (2004). Corpus-oriented grammar development for acquiring a Head-Driven Phrase Structure Grammar from the Penn Treebank. In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP 2004)* (pp. 684–693). Hainan Island, China.

Miyao, Y., & Tsujii, J. (2002). Maximum entropy estimation for feature forests. In *Proceedings of the Human Language Technology Conference.* San Diego, USA.

Miyao, Y., & Tsujii, J. (2008). Feature forest models for probabilistic HPSG parsing. *Computational Linguistics*, *34*(1), 35–80.

Ninomiya, T., Matsuzaki, T., Shimizu, N., & Nakagawa, H. (2011). Deterministic shift-reduce parsing for unification-based grammars. *Natural Language Engineering*(3), 331–365.

Oepen, S., Flickinger, D., Toutanova, K., & Manning, C. D. (2004). LinGO Redwoods: a rich and dynamic treebank for HPSG. *Journal of Research in Language and Computation*, *2*(4), 575–596.

Oepen, S., Flickinger, D., Tsujii, J., & Uszkoreit, H. (Eds.). (2002). *Collaborative language engineering. A case study in efficient grammar-based processing*. Stanford, CA: CSLI Publications.

Oepen, S., & Lønning, J. T. (2006). Discriminant-based MRS banking. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)* (pp. 1250–1255). Genoa, Italy.

Oepen, S., Toutanova, K., Shieber, S., Manning, C., Flickinger, D., & Brants, T. (2002). The LinGO Redwoods treebank: motivation and preliminary applications. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING).* Taipei, Taiwan.

Pereira, F. C., & Wright, R. N. (1991). Finite-state approximation of phrase structure grammars. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics* (pp. 246–255). Berkeley, CA, USA.

Petrov, S., Barrett, L., Thibaux, R., & Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics and the 21st International Conference on Computational Linguistics (COLING)* (pp. 433–440). Sydney, Australia.

Pollard, C. J., & Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. Chicago, USA: University of Chicago Press.

Rimell, L., & Clark, S. (2008). Constructing a parser evaluation scheme. In *Proceedings of the COLING Workshop on Cross-framework and Cross-domain Parser Evaluation* (pp. 44–50). Manchester, UK.

Schmid, H. (2004). Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)* (pp. 162–168). Geneva, Switzerland.

Toutanova, K., Manning, C. D., Flickinger, D., & Oepen, S. (2005). Stochastic HPSG parse selection using the Redwoods corpus. *Journal of Research on Language and Computation*, *3*(1), 83–105.

van Noord, G. (2006). At Last Parsing Is Now Operational. In *Actes de la 13e conference sur le traitement automatique des langues naturelles (TALN 2006)* (pp. 20–42). Leuven, Belgium.

Ytrestøl, G., Flickinger, D., & Oepen, S. (2009). Extracting and annotating Wikipedia subdomains: Towards a new eScience community resource. In *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theory* (pp. 185–197). Groningen, the Netherlands.

Yu, K., Yusuke, M., Wang, X., Matsuzaki, T., & Tsujii, J. (2010). Semi-automatically developing Chinese HPSG grammar from the Penn Chinese Treebank for deep parsing. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)* (pp. 1417–1425). Beijing, China.