# Combining Shallow and Deep Processing for a Robust, Fast, Deep-Linguistic Dependency Parser

Gerold Schneider

Department of Linguistics, University of Geneva *
Institute of Computational Linguistics, University of Zurich
gerold.schneider@lettres.unige.ch

**Abstract**

This paper describes Pro3Gres, a fast, robust, broad-coverage parser that delivers deep-linguistic grammatical relation structures as output, which are closer to predicate-argument structures and more informative than pure constituency structures. The parser stays as shallow as is possible for each task, combining shallow and deep-linguistic methods by integrating chunking and by expressing the majority of long-distance dependencies in a context-free way. It combines statistical and rule-based approaches, different linguistic grammar theories and different linguistic resources. Preliminary evaluations indicate that the parser's performance is state-of-the-art.

## 1 Introduction

A variety of rule-based deep-linguistic parsers, usually following a formal linguistic theory, have existed for a number of years. Formal Grammar parsers have carefully crafted grammars written by professional linguists. In addition to expressing local relations, i.e. relations between a mother and a direct daughter node, a number of non-local relations, i.e. relations involving more than two generations, are also modeled. Unrestricted real-world texts pose a problem to many NLP systems that are based on Formal Grammars. Robust statistical parsers offer solutions to this problem [5], [8], [14], but they typically produce CFG constituency data as output, trees that do not express long-distance dependencies, grammatical functions or empty nodes, although Treebanks such as the Penn Treebank [21], on which they are typically trained, provide them[1]. This entails that the training cannot profit from valuable annotation data, and that the extraction of long-distance dependencies (LDD) and the mapping to shallow semantic representations is not always possible from the output of these parsers. The problem is even aggravated by parsing errors across an LDD [17].

Thanks to integrating statistics, some deep-linguistic Formal Grammar systems have now also achieved the coverage and robustness needed to parse large corpora: [27] show how a hand-crafted LFG grammar scales to the Penn Treebank with Maximum Entropy probability models. [3] acquire a wide-coverage LFG grammar from the Penn Treebank automatically, [15] a CCG grammar. We use a tag-sequence based, hand-crafted functional dependency grammar (FDG, [13], [29]), combined with

*This research was partly made possible by the Swiss National Science Foundation, grant 21-59416.99. Also many thanks to the anonymous reviewers for their valuable comments

[1] [8] Model 2 uses some of the functional labels, and Model 3 some long-distance dependencies

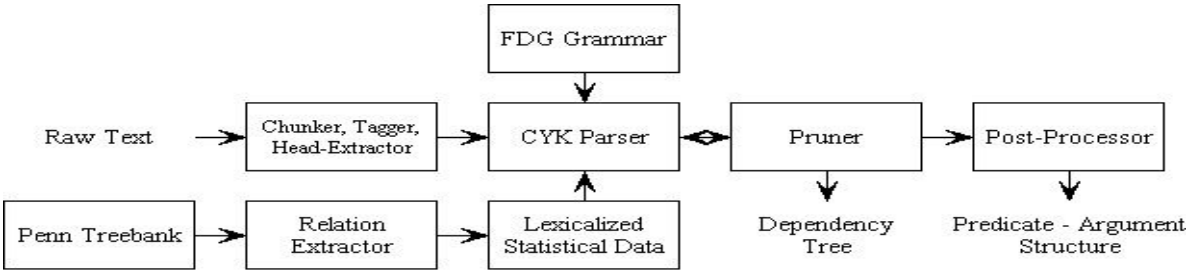



Figure 1: Pro3Gres flowchart

Maximum Likelihood Estimation (MLE) lexicalized probabilities extracted from the Penn Treebank, similar to [9], but for a large subset of dependency relations instead of for PP-attachment only, including the majority of LDDs.

Speed and complexity remain a serious challenge for Formal Grammar based parsers. Typical Formal Grammar parser complexity is much higher than the O($n^3$) for CFG [11]. Parsing algorithms able to treat completely unrestricted long-distance dependencies are NP-complete [24]. Here, we suggest a particularly low-complexity, robust solution, offering on the one hand context-free parsing complexity, but on the other hand a deep-lingusitic analysis as with a type of Formal Grammar. Pro3Gres parses about 300,000 words per hour. Its flow chart is in fig. 1, a sample output in fig. 2.

The low complexity is due to the following reasons:

- Exploiting Functional Dependency Grammar, a Formal Grammar theory that directly delivers simple predicate-argument structures, that is in Chomsky Normal Form and does not know empty nodes, which allows us to employ a low-complexity parsing algorithm (CYK). In terms of [17], we apply the subtree-patterns not after, but before parsing, on the gold-standard and associate groups of patterns to dedicated dependency labels. Like [16] we include empty-node and functional label information to get near-full precision.[2]
- Tagging and chunking: As suggested by [2], we rely on shallow finite-state based approaches at the base phrase level and only parse between heads of chunks.
- Hand-crafted near-full coverage grammar: fully comprehensive grammars are difficult to maintain and considerably increase parsing complexity. We intentionally employ a near-full coverage grammar that does not cover very rare and marked constructions, and that leaves distinctions that are less relevant for the predicate-argument recognition task underspecified.
- Aggressive pruning: in order to keep search spaces small we discard improbable alternatives during the parsing process. Assuming a fixed beam size in a parse-time pruning system, which means that the total number of alternatives kept is constant from a certain search complexity onwards, real-world parsing time can be reduced to near-linear. If one were to assume a constantly full beam, or uses an oracle [25] it is linear in practice [3].

The rest of the paper is structured as follows. First we discuss that the parser stays as shallow as is possible for each task, combining shallow and deep-linguistic methods by integrating chunking and by expressing long-distance dependencies in a context-free way. Second, the probability model is illustrated. Finally, the performance of the parser and some of its elements are evaluated.

[2]Unlike Jijkoun, we apply the patterns directly to the Treebank, not a dependency-mapped version. Our patterns constitute a selective mapping from the Penn Treebank to Functional Dependency

[3]In practical terms, beam or oracle approach have very similar effects. If reasonable parameters are used, parser performance decreases only marginally while time behaviour improves by one to several orders of magnitude

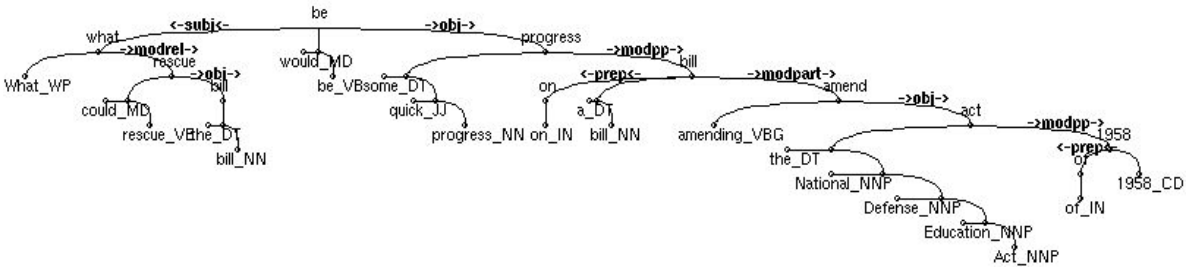



Figure 2: SWI Prolog Implementation Dependency Tree sample output

## 2 Stay as Shallow as You Can

In order to keep parsing complexity as low as possible, aggressive use of shallow techniques is made. For low-level syntactic tasks, tagging and chunking is used, and context-sensitive tasks are reduced to context-free tasks as far as is possible by the use of patterns that are deep-linguistic as they are non-local, but shallow as they are fixed. But it also entails that a hand-written, intentionally only almost fully comprehensive grammar, combined with aggressive pruning and a robust method for collecting partial parses is employed.

### 2.1 Tagging and Chunking

Low-level linguistic tasks that can be reliably solved by finite-state based techniques are handed over to them; namely the recognition of part-of-speech by means of tagging, and the recognition of base NPs and verbal groups and their heads by means of chunking [1], [2], [22]. The chunker and the head extraction method are completely rule-based. A small evaluation shows about 98 % correct head extraction. The extracted heads are lemmatized [23]. Parsing takes place only between the heads of chunks, and only using the best tag suggested by the tagger. The average number of words per chunk is 1.52 for Treebank section 0. In a speed test on section 0 with a toy NP and verb-group grammar parsing was about 4 times slower when using unchunked data as input[4]. The average number of possible tags per token is 2.11 for the entire Treebank. With untagged input, every possible tag would have to be taken into consideration by the parser. Although untested, at least a similar slowdown for untagged input as for unchunked input can be expected.

### 2.2 The Hand-Written Grammar

Acknowledged facts, for example that a verb has typically one but never two subjects, that adjuncts follow after all complements, that verbs can maximally have two noun objects etc. are expressed in hand-written declarative rules. The rules are based on the Treebank tags of heads of chunks. Since the tagset is limited and dependency rules are binary, even a broad-coverage set of rules can be written in relatively little time.

Linguistic constructions that are possible but very marked and rare are ruled out in this practically oriented system. For example, while it is generally possible for nouns to be modified by more than one PP, only nouns seen in the Treebank with several PPs are allowed to have several PPs. Or, while it is generally possible for a subject to occur to the immediate right of a verb (*said she*), this is only

[4]Due to the insufficiency of the toy grammar the lingusitic quality and the number of complete parses decreased, the same quality of analysis would even have been slower

| Relation | Label | Example | Relation | Label | Example |
|---|---|---|---|---|---|
| verb–subject | subj | *he sleeps* | verb–prep. phrase | pobj | *slept in bed* |
| verb–first object | obj | *sees it* | noun–prep. phrase | modpp | *draft of paper* |
| verb–second object | obj2 | *gave (her) kisses* | noun–participle | modpart | *report written* |
| verb–adjunct | adj | *ate yesterday* | verb–complementizer | compl | *to eat apples* |
| verb–subord. clause | sentobj | *saw (they) came* | noun–preposition | prep | *to the house* |

Table 1: The most important dependency types used by the parser

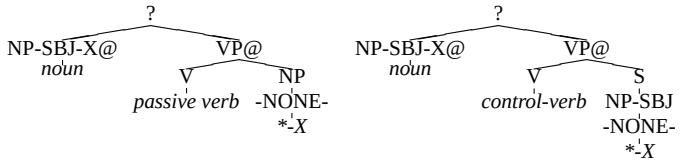



Figure 3: The extaction patterns for passive subjects (left) and subject control (right)

allowed for verbs seen with a subject to the right in the training corpus, typically verbs of utterance, and only in a comma-delimited or sentence-final context.

In a hand-written grammar, some typical parsing errors can be corrected by the grammar engineer, or rules can explicitly ignore particularly error-prone distinctions. Examples of rules that can correct tagging errors without introducing many new errors are allowing *VBD* to act as a participle (*VBN*) or the possible translation of *VBG* to an adjective. As an example of ignoring error-prone distinctions, the tagger's disambiguation between prepositions and verbal particles is unreliable. The grammar therefore makes no distinction between verbal particles and prepositions.

## 2.3 Long-Distance Dependencies

Treating long-distance dependencies is very costly [24], as they are context-sensitive. Most statistical Treebank trained parsers thus fully or largely ignore them.

[17] presents a pattern-matching algorithm for post-processing the Treebank output of such parsers to add empty nodes expressing long-distance dependencies to their parse trees. Encouraging results are reported for perfect parses, but performance drops considerably when using parser output trees. We have applied structural patterns to the Treebank, where like in perfect parses precision and recall are high, and where in addition functional labels and empty nodes are available, so that patterns similar to Johnson's but relying on functional labels and empty nodes reach precision close to 100%. Unlike in Johnson, also patterns for local dependencies are used; non-local patterns simply stretch across more subtree-levels. We use the extracted lexical counts as lexical frequency training material. Every dependency relation has a group of structural extraction patterns associated with it. This amounts to a selective mapping of the Penn Treebank to Functional DG. Table 1 gives an overview of the most important dependencies.

The $subj$ relation, for example, has the head of an arbitrarily nested NP with the functional tag *SBJ* as dependent, and the head of an arbitrarily nested VP as head for all active verbs. In passive verbs, however, a movement involving an empty constituent is assumed, which corresponds to the extraction pattern in figure 3, where VP@ is an arbitrarily nested VP, and NP-SBJ-X@ the arbitrarily nested surface subject and X the co-indexed, moved element. Movements are generally supposed to be of arbitrary length, but a closer investigation reveals that this type of movement is mostly fixed and can as well be replaced by a single, local and thus context-free dependency. Since the verb form allows a clear identification of passive structures, we have decided to keep the relation label $subj$, but to use

| | Antecedent | POS | Label | Count | Description | Example |
|---|---|---|---|---|---|---|
| 1 | NP | NP | * | 22,734 | NP trace | *Sam* was seen * |
| 2 | | NP | * | 12,172 | NP PRO | * to sleep is nice |
| 3 | WHNP | NP | *T* | 10,659 | WH trace | the woman *who* you saw *T* |
| (4) | | | *U* | 9,202 | Empty units | $ 25 *U* |
| (5) | | | 0 | 7,057 | Empty complementizers | Sam said 0 Sasha snores |
| (6) | S | S | *T* | 5,035 | Moved clauses | *Sam had to go*, Sasha said *T* |
| 7 | WHADVP | ADVP | *T* | 3,181 | WH-trace | Sam explained *how* to leave *T* |
| (8) | | SBAR | | 2,513 | Empty clauses | *Sam had to go*, said Sasha (SBAR) |
| (9) | | WHNP | 0 | 2,139 | Empty relative pronouns | the woman 0 we saw |
| (10) | | WHADVP | 0 | 726 | Empty relative pronouns | the reason 0 to leave |

Table 2: The distribution of the 10 most frequent types of empty nodes and their antecedents in the Penn Treebank (adapted from [17])

| Type | Count | prob-modeled | Treatment |
|---|---|---|---|
| passive subject | 6,803 | YES | local relation |
| indexed gerund | 4,430 | NO | Tesnière translation |
| control, raise, semi-aux | 6,020 | YES | post-parsing processing |
| others / not covered | 5,481 | | |
| TOTAL | 22,734 | | |

Table 3: Coverage of the patterns for the most frequent NP traces [row 1]

separate probability estimations for the active and the passive case.

The same argument can be made for other relations, for example control structures, which have the extraction pattern shown in figure 3. Some relations include local alongside non-local dependencies. For example, the $obj$ relation includes copular verb complements and small clause complements.

Grammatical role labels, empty node labels and tree configurations spanning several local subtrees are used as integral part of some of the patterns. This leads to much flatter trees, as typical for DG, which has the advantages that (1) it helps to alleviate sparse data by mapping nested structures that express the same dependency relation, (2) less decisions are needed at parse-time, which greatly reduces complexity and the risk of errors [17], (3) the costly overhead for dealing with unbounded dependencies can be largely avoided. [5]

Let us consider the quantitative coverage of these patterns in detail. The ten most frequent types of empty nodes cover more than 60,000 of the approximately 64,000 empty nodes of sections 2-21 of the Penn Treebank. Table 2, reproduced from [17] [line numbers and counts from the whole Treebank added], gives an overview.

Empty units, empty complementizers and empty relative pronouns [lines 4,5,9,10] pose no problem for DG as they are optional, non-head material. For example, a complementizer is an optional dependent of the subordinated verb. Moved clauses [line 6] are mostly PPs or clausal complements of verbs of utterance. Only verbs of utterance allow subject-verb inversion in affirmative clauses [line 8]. The linguistic grammar provides rules with appropriate restrictions for all of these. In a dependency framework, none of them involve non-local dependencies or empty nodes, [line 6] and [line 8] need rules that allow an inversion of the dependency direction under well-defined conditions.

[5] In addition to taking less decisions, it is ensured that the lexical information that matters is available in one central place, allowing the parser to take one well-informed decision instead of several brittle decisions plagued by sparseness. Collapsing deeply nested structures into a single labeled dependency relation is less complex but has the same effect as carefully selecting what goes in to the parse history in history-based approaches [5]

**NP Traces** The analysis of NP traces ([line 1] of table 2) is shown in table 3. It reveals that the majority of them are recognized by the grammar, and except for the indexed gerunds, they participate in the probability model. In control, raising and semi-auxiliary constructions, the non-surface semantic arguments, i.e. the subject-verb relation in the subordinate clause, are created based on lexical probabilities at the post-parsing stage, where minimal predicate-argument structures are output.

Unlike in control, raising and semi-auxiliary constructions, the antecedent of an indexed gerund cannot be established easily. The parser does not try to decide whether the target gerund is an indexed or non-indexed gerund nor does it try to find the identity of the lacking participant in the latter case. This is an important reason why recall values for the subject and object relations are lower than the precision values.

**NP PRO** As for the 12,172 NP PRO [line 2] in the Treebank, 5,656 are recognized by the $modpart$ pattern (which covers reduced relative clauses), which means they are covered in the probability model. The dedicated $modpart$ relation typically expresses object function for past participles and subject function for present participles.[6] A further 3,095 are recognized as non-indexed gerunds. Infinitives and gerunds may act as subjects, which are covered by [30] translations, although these rules do not participate in the probability model. Many of the structures that are not covered by the extraction patterns and the probability model are still parsed correctly, for example adverbial clauses as unspecified subordinate clauses. Non-indexed adverbial phrases of the verb account for 1,598 NP PRO, non-indexed adverbial phrases of the noun for 268. As the NP is non-indexed, the identity of the lacking argument in the adverbial is unknown anyway, thus no semantic information is lost.

**WH Traces** Only 113 of the 10,659 WHNP antecedents in the Penn Treebank [line 3] are actually question pronouns. The vast majority, over 9,000, are relative pronouns. For them, an inversion of the direction of the relation they have to the verb is allowed if the relative pronoun precedes the subject. This method succeeds in most cases, but linguistic non-standard assumptions need to be made for stranded prepositions.

Only non-subject WH-question pronouns and support verbs need to be treated as "real" non-local dependencies. In question sentences, before the main parsing is started, the support verb is attached to any lonely participle chunk in the sentence, and the WH-pronoun pre-parses with any verb.

# 3 Probability Model

We now explain Pro3Gres' main probability model by way of comparing it to [8]. We first consider the non-generative model [6] and then show how Pro3Gres' rules implement the extensions of [7] Model 2.

## 3.1 Relation of Pro3Gres to Collins 1996

Both [6] and Pro3Gres are mainly dependency-based statistical parsers parsing over heads of chunks, a close relation can therefore be expected. The [6] MLE and the main Pro3Gres MLE can be juxtaposed as follows ($a$ and $b$ are the lexical heads, $R$ the relation, $dist$ the distance):

[6] The possible functional ambiguity is not annotated in the Treebank, hence the reduced relative clause is an unindexed empty NP

[6] MLE estimation: $P(R|\langle a, atag\rangle, \langle b, btag\rangle, dist) \cong \frac{\#(R,\langle a,atag\rangle,\langle b,btag\rangle,dist)}{\#(\langle a,atag\rangle,\langle b,btag\rangle,dist)}$

Main Pro3Gres MLE estimation: $P(R, dist|a, b) \cong p(R|a, b) \cdot p(dist|R) = \frac{\#(R,a,b)}{\#(\sum_{i=1}^{n} R_i a,b)} \cdot \frac{\#(R,dist)}{\#R}$

Four differences are observed: First, Pro3Gres does not use tag information. Reasons for this are because the licensing, hand-written grammar is based on Penn tags, and because Pro3Gres backs off to semantic WordNet classes [12] for nouns and to Levin classes [18] for verbs instead of to tags, which has the advantage that it is more fine-grained. Second, Pro3Gres uses real distances, measured in chunks, instead of a vector of features. While the type of relation $R$ is lexicalized, i.e. conditioned on the lexical items, the distance is assumed to be dependent only on $R$. This is based on the observation that some relations typically have very short distances (e.g. verb-object), others can be quite long (e.g. Verb-PP attachment). This observation greatly reduces the sparse data problem. Third, the co-occurrence count in the MLE denominator is not the sentence-context, but the sum of competing relations. For example, the $object$ and the $adjunct$ relation are in competition, as they are licensed by the same tag sequence ($VB* \; NN*$). Pro3Gres models attachment probabilities, decision probabilities, which is in accordance with the view that parsing is a decision process. Fourth, relations ($R$) have a Functional Dependency Grammar definition, including long-distance dependencies.

### 3.2 Relation of Pro3Gres to Collins 1997 Model 2

[7] Model 2 extends the parser to include a complement/adjunct distinction for NPs and subordinated clauses, and it includes a subcategorisation frame model.

Let us first review [7] Model 2 with the example of the rewrite rule ***S(bought)*** $\rightarrow$ ***NP(week), NP-C(IBM), VP(bought)***. For the subcategorisation-dependent generation of dependencies in Model 2, first the probabilities of the possible subcat frames to the right $p_{rc}$ and to the left $p_{lc}$ are calculated, conditioned on the RHS mother category $P$, the LHS head category $H$ and the lexical head $h$. The selected subcat frame is added as a condition to the left context $l$, respectively the right context $r$.

$$P_{head}(\text{VP}|\text{S,bought}) \cdot P_{lsubcat}(\{\text{NP-C}\}|\text{S,VP,bought}) \cdot P_{rsubcat}(\{\}|\text{S,VP,bought}) \cdot P_l(\text{NP-C(IBM)}|\text{S,VP,bought,}\{\text{NP-C}\}) \quad (1)$$

Once a subcategorized constituent has been found, it is removed from the subcat frame, so that if *IBM* is NP-C, h=*week* has an empty subcat frame.

$$P_l(\text{NP(week)}|\text{S,VP,bought,}\{\}) \quad (2)$$

This ensures that non-subcategorized constituents cannot be attached as complements, which is one of the two major function of a subcat frame. The other major function of a subcat frame is to ensure that, if possible, all the subcategorized constituents are found. In order to ensure this, the probability when a rewrite rule can stop expanding is calculated. Importantly, the probability of a rewrite rule with a non-empty subcat frame to stop expanding is very low, the probability of a rewrite rule with a non-empty subcat frame to stop expanding is higher.

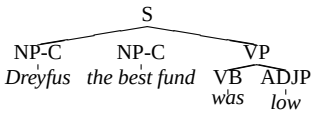
$$P_l(\text{STOP}|\text{S,VP,bought,}\{\}) \cdot P_r(\text{STOP}|\text{S,VP,bought,}\{\}) \quad (3)$$

The entire probability of the phrase S(bought) $\rightarrow$ NP(week), NP-C(IBM), VP(bought) is therefore

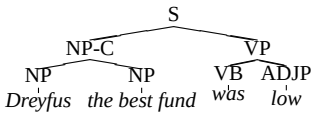
$$\begin{aligned} &P_{head}(\text{VP}|\text{S,bought}) \cdot P_{lsubcat}(\{\text{NP-C}\}|\text{S,VP,bought}) \cdot P_{rsubcat}(\{\}|\text{S,VP,bought}) \\ &\cdot P_l(\text{NP-C(IBM)}|\text{S,VP,bought,}\{\text{NP-C}\}) \cdot P_l(\text{NP(week)}|\text{S,VP,bought,}\{\}) \\ &\cdot P_l(\text{STOP}|\text{S,VP,bought,}\{\}) \cdot P_r(\text{STOP}|\text{S,VP,bought,}\{\}) \end{aligned}$$

Pro3Gres includes a complement/adjunct distinction for NPs. All the examples given in support of the subcategorisation frame model in [7] are dealt with by the hand-written grammar.

(a) incorrect

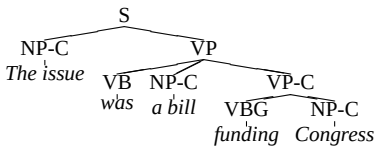



(b) correct

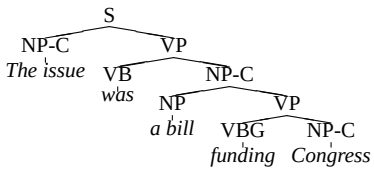



The grammar constraint that a verb is not allowed to have more than one subject forbids the incorrect analysis in Pro3Gres.

(a) incorrect




(b) correct




The $obj$ and $obj2$ relations, the latter exclusively for ditransitive verbs, are separate relations in Pro3Gres. The $obj2$ relation probability is largely a verb ditransitivity probability.

In other words, every complement relation type, namely $subj$, $obj$, $obj2$, $sentobj$, can only occur once per verb, which ensures one of the two major functions of a subcat frame, that non-subcategorized constituents cannot be attached as complements. This amounts to keeping separate subcat frames for each relation type, where the selection of the appropriate frame and removing the found constituent coincide, which has the advantage of a reduced search space: no hypothesized, but unfound subcat frame elements need to be managed. As for the second major function of subcat frames – to ensure that if possible all subcategorized constituents are found – the same principle applies: selection of subcat frame and removing of found constituents coincide; lexical information on the verb argument candidate is available at frame selection time already. This implies that Collins' Model 2 takes an unnecessary detour.

As for the probability of stopping the expansion of a rule – since DG rules are always binary – it is always 0 before and 1 after the attachment. But what is needed in place of interrelations of constituents of the same rewrite rule is proper cooperation of the different subcat types. For example, the grammar rules only allow a noun to be $obj2$ once $obj$ has been found, or a verb is required to have a subject unless it is non-finite or a participle, or all objects need to be closer to the verb than a subordinate clause.

# 4 Evaluation

In traditional constituency approaches, parser evaluation is done in terms of the correspondence of the bracketting between the gold standard and the parser output. [19] suggested evaluating on the linguistically more meaningful level of syntactic relations. For the current evaluation, a hand-compiled gold standard following this suggestion is used [4]. It contains the grammatical relation data of 500 random sentences from the Susanne corpus. The mapping between Carroll's grammatical relation format and our dependency output is explained in [28]. Mapping one annotation scheme to another is non-trivial and can only lead to indicative results [10]. The results in table 4 are similar to those reported in [20] and [26]. They suggest that the performance of the parser is roughly state-of-the-art[7], but more research will be needed. The new local relations corresponding to LDDs in the Penn Treebank have been selectively evaluated as far as the annotations permit, also shown in table 4:

[7] [20] gives results on the whole Susane corpus, [26] compares a number of successful probabilistic parsers using the same gold standard [4] as we do

| | Percentages for some relations, on Carroll testset | | | | | only LDD-involving | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Subject | Object | noun-PP | verb-PP | subord. clause | WHS | WHO | PSubj | CSubj | modpart | RclSubjA |
| Precision | 91 | 89 | 73 | 74 | 68 | 92 | 60 | n/a | 80 | 74 | 89 |
| Recall | 81 | 83 | 67 | 83 | n/a | 90 | 86 | 83 | n/a | n/a | 63 |

Table 4: Evaluation on Carroll's test suite on subject, object, PP-attachment and clause subordination relations and a selection of 6 LDD relations

| Error Classification of PP-Attachment Errors of the first 100 evaluation corpus sentences | | | | | | |
|---|---|---|---|---|---|---|
| Description | Attachment Error | Head Extraction Error | Chunking or Tagging Error | compl/prep Error | Grammar Mistake or incomplete Parse | Grammar Assumption |
| Noun-PP Attach Prec. | 22 | 1 | 8 | 0 | 3 | 3 |
| Verb-PP Attach Prec. | 12 | 1 | 5 | 1 | 1 | 2 |
| Noun-PP Attach Recall | 25 | 1 | 14 | 0 | 12 | 5 |
| Verb-PP Attach Recall | 2 | 0 | 1 | 0 | 0 | 0 |
| Percentages | 51 % | 3 % | 24 % | 1 % | 13 % | 12 % |

Table 5: Analysis of PP-Attachment Errors

WH-Subject (WHS), WH-Object (WHO), passive Subject (PSubj), control Subject (CSubj), reduced relative clause (modpart), and the anaphor of the relative clause pronoun (RclSubjA). Table 5 shows that about half of the PP-attachment errors are real attachment errors. The second most frequent error is deficient tagging or chunking – the price to pay for shallowness.

# 5 Conclusions

We have presented a fast, lexicalized broad-coverage parser delivering grammatical relation structures as output, which are closer to predicate-argument structures than pure constituency structures, and more informative if non-local dependencies are involved. A selective evaluation at the grammatical relation level indicates that its performance is state-of-the-art, but a full evaluation will be needed as future research.

We have shown that the parser stays as shallow as is possible for each task, combining shallow and deep-linguistic methods by integrating chunking and by expressing long-distance dependencies in a context-free way, thus offering on the one hand a parsing complexity as low as for a typical probabilistic parser, but on the other hand a deep-lingusitic analysis as with a type of Formal Grammar. In LFG terms, we parse directly for an f-structure without a c-structure detour. We model the majority of empty nodes and co-references by using an ID/LP grammar, dedicated functional labels, conservative DG assumptions [30] and statistical post-processing.

# References


[1] Steven Abney. Parsing by chunks. In *Principle-Based Parsing*. Kluwer Academic Publishers, Dordrecht, 1991.

[2] Steven Abney. Chunks and dependencies: Bringing processing evidence to bear on syntax. In Jennifer Cole, Georgia Green, and Jerry Morgan, editors, *Computational Linguistics and the Foundations of Linguistic Theory*, pages 145–164. CSLI, 1995.

[3] M. Burke, A. Cahill, R. O'Donovan, J. van Genabith, and A. Way. Treebank-based acquisistion of wide-coverage, probabilistic LFG resources: Project overview, results and evaluation. In *The First International Joint Conference on Natural Language Processing (IJCNLP-04), Workshop "Beyond shallow analyses - Formalisms and statistical modeling for deep analyses"*, Sanya City, Hainan Island, China, 2004.

[4] John Carroll, Guido Minnen, and Ted Briscoe. Corpus annotation for parser evaluation. In *Proceedings of the EACL-99 Post-Conference Workshop on Linguistically Interpreted Corpora*, Bergen, Norway, 1999.

[5] Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the North American Chapter of the ACL*, pages 132–139, 2000.

[6] Michael Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 184–191, Philadelphia, 1996.

[7] Michael Collins. Three generative, lexicalised models for statistical parsing. In *Proc. of the 35th Annual Meeting of the ACL*, pages 16–23, Madrid, Spain, 1997.

[8] Michael Collins. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, 1999.

[9] Michael Collins and James Brooks. Prepositional attachment through a backed-off model. In *Proceedings of the Third Workshop on Very Large Corpora*, Cambridge, MA, 1995.

[10] Richard Crouch, Ronald M. Kaplan, Tracy H. King, and Stefan Riezler. A comparison of evaluation metrics for broad-coverage stochastic parsers. In *Beyond PARSEVAL workshop at 3rd Int. Conference on Language Resources an Evaluation (LREC'02)*, Las Palmas, 2002.

[11] Jason Eisner. Bilexical grammars and a cubic-time probabilistic parser. In *Proceedings of the 5th International Workshop on Parsing Technologies*, pages 54–65, MIT, Cambridge, MA, September 1997.

[12] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.

[13] Jan Hajič. Building a syntactically annotated corpus: The Prague Dependency Treebank. In Eva Hajičová, editor, *Issues of Valency and Meaning. Studies in Honor of Jarmila Panevová*, pages 106–132. Karolinum, Charles University Press, Prague, 1998.

[14] James Henderson. Inducing history representations for broad coverage statistical parsing. In *Proceedings of HLT-NAACL 2003*, Edmonton, Canada, 2003.

[15] Julia Hockenmaier and Mark Steedman. Generative models for statistical parsing with combinatory categorial grammar. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, 2002.

[16] Valentin Jijkoun. Finding non-local dependencies: beyond pattern matching. In *Proceedings of the ACL 03 Student Workshop*, Budapest, 2003.

[17] Mark Johnson. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Meeting of the ACL*, University of Pennsylvania, Philadelphia, 2002.

[18] Beth C. Levin. *English Verb Classes and Alternations: a Preliminary Investigation*. University of Chicago Press, Chicago, IL, 1993.

[19] Dekang Lin. A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of IJCAI-95*, Montreal, 1995.

[20] Dekang Lin. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain, 1998.

[21] Mitch Marcus, Beatrice Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330, 1993.

[22] Andrei Mikheev. Automatic rule induction for unknown word guessing. *Computational Linguistics*, 23(3):405–423, 1997.

[23] Guido Minnen, John Carroll, and Darren Pearce. Applied morphological generation. In *Proceedings of the 1st International Natural Language Generation Conference (INLG)*, Mitzpe Ramon, Israel, 2000.

[24] Peter Neuhaus and Norbert Bröker. The complexity of recognition of linguistically adequate dependency grammars. In *Proceedings of the 35th ACL and 8th EACL*, pages 337–343, Madrid, Spain, 1997.

[25] Joakim Nivre. Inductive dependency parsing. In *Proceedings of Promote IT*, Karlstad University, 2004.

[26] Judita Preiss. Using grammatical relations to compare parsers. In *Proceedings of EACL 03*, Budapest, Hungary, 2003.

[27] Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, Philadephia, PA, 2002.

[28] Gerold Schneider. Extracting and using trace-free Functional Dependencies from the Penn Treebank to reduce parsing complexity. In *Proceedings of Treebanks and Linguistic Theories (TLT) 2003*, Växjö, Sweden, 2003.

[29] Pasi Tapanainen and Timo Järvinen. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71. Association for Computational Linguistics, 1997.

[30] Lucien Tesnière. *Eléments de Syntaxe Structurale*. Librairie Klincksieck, Paris, 1959.