Combining shallow and deep processing for a robust, fast, deep-linguistic dependency parser

Schneider, G

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 parsers performance is state-of-the-art.

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

Gerold Schneider

Institute of Computational Linguistics, University of Zurich
Department of Linguistics, University of Geneva *
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 novel, context-free way. It is hybrid in many ways, combining statistical and rule-based approaches, different linguistic grammar theories and different linguistic resources. The performance of the parser is shown to be 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. These are, to name only a few, the Alvey tools [4] for GPSG, Lingo [13] for HPSG, FIPS [39] or PAPPI [18] for GB, and MINIPAR [27] or FDG [37] for DG.

Formal Grammar Parsers generally have good coverage of most syntactic phenomena, since they 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. An example of a non-local relation is the *subject control* relation in the sentence *John wants to leave*, where *John* is not only the explicit subject of *want*, but equally the implicit subject of *leave*. A simple CFG annotation with a coreference expressing the identity of the explicit and implicit subject is shown in figure 1. A parser that fails to recognize control subjects misses very important information, quantitatively about 3 % of all subjects.

But Formal Grammars have the disadvantage that their scoring systems for disambiguation are heuristic, which entails that they are complex to conceptualize and maintain, and without an empirical base. They are hand-written instead of learnt from real-world data and potentially suffer from a number of serious problems.

1. Fully comprehensive grammars are difficult to maintain and considerably increase parsing complexity.

2. Typical formal grammar parser complexity is much higher than the O(n^3) for CFG [15]. The complexity of some formal grammars is still unknown. For Tree-Adjoining Grammars (TAG) it is O(n^7) or O(n^8) depending on the

*This research was partly made possible by the Swiss National Science Foundation, grant 21-59416.99

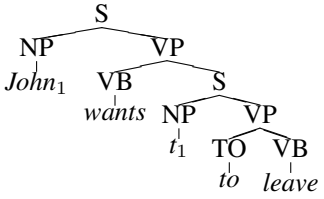


Figure 1: CFG tree structure for *John wants to leave*

implementation [16]. [34] state that the theoretical bound of worst time complexity for Head-Driven Phrase Structure Grammar (HPSG) parsing is exponential. Parsing algorithms able to treat completely unrestricted long-distance dependencies are NP-complete [31]. It is still common practice to exclude sentences longer than 40 words or so from parser evaluations as many of the systems slow down too much or fail or get timed out when applied to long real-world sentences.

3. Returning all syntactically possible analyses for a sentence is only a part of the task that is expected of a syntactic analyzer if it should be of practical use, since for a human there is usually only one, rarely two or three "correct" interpretations. A clear indication of preference, by means of discarding unlikely analyses or ranking the analyses in a preference order is as important.

4. In order to keep search spaces manageable it is in fact necessary to discard unconvincing alternatives already during the parsing process. In a statistical parser, the ranking of intermediate structures occurs naturally, while a rule-based system has to rely on ad hoc heuristics. 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 [32] it is linear). In practical terms, this is perhaps the single most important reason why statistical parsers can be much faster than some rule-based systems. If reasonable pruning is used, parser performance decreases only marginally while time behaviour improves by one or several orders of magnitude.

Broad-coverage statistical syntactic parsers that offer solutions to these problems have now become available [7], [11], [22], but they typically produce CFG constituency data as output, trees that do not express long-distance dependencies.

Although grammatical function and empty nodes annotation expressing long-distance dependencies are provided in Treebanks such as the Penn Treebank [28], most statistical Treebank trained parsers fully or largely ([11] Models 2 and 3) ignore them, which entails two problems: first, the training cannot profit from valuable annotation data. Second, the extraction of long-distance dependencies (LDD) and the mapping to shallow semantic representations is not always possible from the output of these parsers, because first co-indexation information is not available, second a single parsing error across a tree fragment containing an LDD makes its extraction impossible, third some syntactic relations cannot be recovered on configurational grounds only. For example, an S node governing a NP and a VP can express a subject relation, but also a reduced relative clause, e.g. *[the report issued] has shown ....*

[24] presents a pattern-matching algorithm for post-processing the output of such parsers to add empty nodes to their parse trees. While encouraging results are reported for perfect parses, performance drops considerably when using trees produced by the parser. "If the parser makes a single parsing error anywhere in the tree fragment matched by the pattern, the pattern will no longer match. This is not unlikely since the statistical model used by the parser does not model these larger tree fragments. It suggests that one might improve performance by integrating parsing, empty node recovery and antecedent finding in a single system ... " [24]. The parser presented here offers a response to this suggestion by combining a statistical approach with a rule-based approach in Dependency Grammar (DG).

The Pro3Gres parser[1], a CYK-based, lexicalized dependency parser, has been described in more detail elsewhere [36], [35]. Generally, it is a hybrid system differing on the one hand from successful DG implementations (e.g. [27], [37]) by using a statistical base, and on the other hand from state-of-the-art statistical approaches (e.g. [11]) by carefully following an established formal grammar theory. It employs both a hand-written linguistic grammar based on Penn tags, and an attachment probability model based on Maximum Likelihood Estimation (MLE) to rank parses and prune unlikely readings during the parsing process using a beam search. The lexical probabilities of relations between heads of phrases are calculated, similar to [12], but for a large subset of dependency relations instead of for PP-attachment only, including the majority of long-distance dependencies, thus offering on the one hand a parsing complexity as low as for a probabilistic parser, but on the other hand a deep-lingusitic analysis as with a type of formal grammars. Pro3Gres parses about 300,000 words per hour.

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 novel, context-free way. Second, the probability model is illustrated. Then, the many ways in which the parser is hybrid are elaborated. Finally, the performance of the parser and some of its elements are evaluated.

[1] Pro3Gres stands for Parser that is RObust, PRObability-based, PROlog-implemented, Grammatical Relation Extraction System

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

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 techniques are handed over to them. These low-level tasks are 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], [29]. 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 [30]. Parsing takes place only between the heads of chunks, and only using the best tag suggested by the tagger. The chunk to word relation is 1.52 for Treebank section 0. In a test with a toy NP and verb-group grammar parsing was about 4 times slower when using the unchunked section 0 as input. Due to the insufficiency of the toy grammar the lingusitic quality and the number of complete parses decreased. The average number of tags per token is 2.11 for the entire Treebank. With untagged input, every possible tag would have to be taken into consideration. Although untested, at least a similar slowdown as for unchunked input can be expected.

2.2 The Hand-Written Grammar

Writing grammar rules is an easy task for a linguist, particularly when using a framework that is close to traditional school grammar assumptions, such as DG. Acknowledged facts such as the one that a verb has typically one but never two subjects, adjuncts only 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 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 [31], as they are context-sensitive. Most statistical Treebank trained parsers thus fully or largely ([11] Models 2 and 3) ignore them.

As mentioned in the introduction, [24] 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

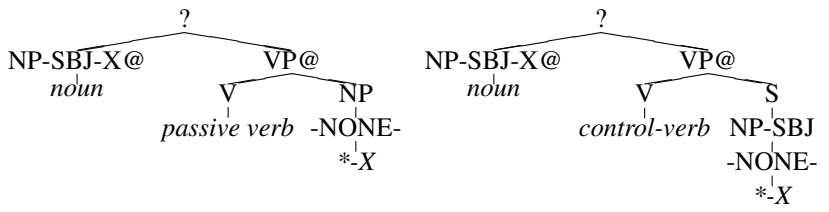


Figure 2: The extaction patterns for passive subjects (left) and subject control (right)

	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 [24])

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 partial mapping of the Penn Treebank to Functional DG [21], [37]. 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 2, 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 fixed. If we neglect the identity between the X-gap and the X-filler, more than 99 % of the passive subject pattern matches happen to have identical X. As this type of movement seems to be hard-coded, it can as well be replaced by a single, local dependency. Since the verb form allows a clear identification of passive structures, we have decided to keep the relation label $subj$, but to use 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 2. 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 [24], (3) the costly overhead for dealing with unbounded dependencies can be largely avoided.

In addition to taking less decisions due to the gained high-level shallowness, 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 dependency relation is less complex but has the same effect as carefully selecting what goes in to the parse history in history-based approaches. "Much of the interesting work is determining what goes into [the history] H(c)". [7]

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 [24] [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

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]

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.

NP Traces A closer look at NP traces ([line 1] of table 2) 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.[2] A further 3,095 are recognized as non-indexed gerunds. Infinitives and gerunds may act as subjects, which are covered by [38] 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 will explain Pro3Gres' main probability model by way of comparing it to [11]. We will first consider the non-generative model [9] and then show how Pro3Gres' rules implement the extensions of [10] Model 2.

3.1 Relation of Pro3Gres to Collins 1996

Both [9] and Pro3Gres are mainly dependency-based statistical parsers parsing over heads of chunks, a close relation can therefore be expected. The [9] MLE and the main Pro3Gres MLE can be juxtaposed as follows:

[9] MLE estimation: $P(R|\langle a, atag\rangle, \langle b, btag\rangle, dist) \cong$

$$\frac{\#(R, \langle a, atag\rangle, \langle b, btag\rangle, dist)}{\#(\langle a, atag\rangle, \langle b, btag\rangle, dist)} \quad (1)$$

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

Main Pro3Gres MLE estimation [35]: $P(R, dist|a, b) \cong p(R|a, b) \cdot p(dist|R) \cong$

$$\frac{\#(R, a, b)}{\#(a, b)} \cdot \frac{\#(R, dist)}{\#R} \tag{2}$$

The following differences are observed:

- Pro3Gres does not use tag information. The first reason for this is because the licensing, hand-written grammar is based on Penn tags.
- The second reason for not using tag information is because Pro3Gres backs off to semantic WordNet classes [17] for nouns and to Levin classes [25] for verbs instead of to tags, which has the advantage that it is more fine-grained.
- 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. [8] have made similar observations for Korean.
- 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 decision probabilities, which is in accordance with the view that parsing is a decision process.
- Relations (R) have a Functional Dependency Grammar definition, including long-distance dependencies.

3.2 Relation of Pro3Gres to Collins 1997 Model 2

[10] 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 revise [10] Model 2 with the example of the rewrite rule *S(bought) → 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.

$$P_{head}(\text{VP}|\text{S,bought}) \cdot P_{lsubcat}(\{\text{NP-C}\}|\text{S,VP,bought}) \cdot P_{rsubcat}(\{\}|\text{S,VP,bought}) \cdot P_l(\text{NP-C(IBM)}|\text{S,VP,bought,}\{\text{NP-C}\}) \tag{3}$$

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.

$$P_l(\text{NP(week)}|\text{S,VP,bought,}\{\}) \tag{4}$$

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.

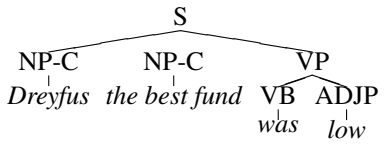
$$P_l(\text{STOP}|\text{S,VP,bought,}\{\}) \cdot P_r(\text{STOP}|\text{S,VP,bought,}\{\}) \tag{5}$$

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

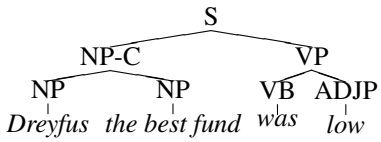
$$\begin{aligned} &P_{head}(\text{VP}|\text{S,bought}) \cdot P_{lsubcat}(\{\text{NP-C}\}|\text{S,VP,bought}) \cdot P_{rsubcat}(\{\}|\text{S,VP,bought}) \\ &\cdot P_l(\text{NP-C(IBM)}|\text{S,VP,bought,}\{\text{NP-C}\}) \cdot P_l(\text{NP(week)}|\text{S,VP,bought,}\{\}) \\ &\cdot P_l(\text{STOP}|\text{S,VP,bought,}\{\})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

Pro3Gres includes a complement/adjunct distinction for NPs. All the examples given in support of the subcategorisation frame model in [10] are dealt with by the hand-written grammar.

(a) incorrect

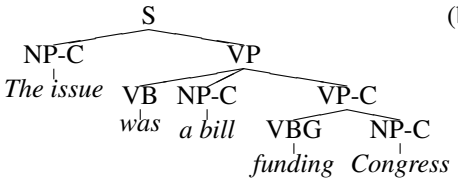


(b) correct

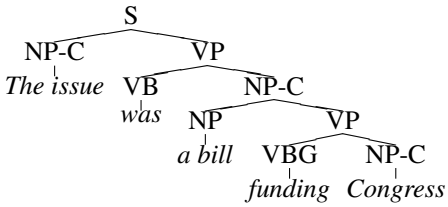


The grammar constraint that a verb is not allowed to have more than one subject forbids the incorrect analysis in Pro3Gres.

(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 A Hybrid Approach On Several Levels

Pro3Gres is hybrid on many levels. A growing number of hybrid approaches form part of the system, or of versions of the system.

1. statistical vs. rule-based: the most obvious way in which Pro3Gres is a hybrid, as described throughout this paper. Unlike formal grammars to which post-hoc statistical disambiguators can be added, Pro3Gres has been designed to be hybrid, carefully distinguishing between tasks that can best be solved by finite-state methods, rule-based methods and statistical methods. While e.g. grammar writing is easy for a linguist, and a naive Treebank grammar suffers from similar complexity problems as a comprehensive formal grammar, the scope of application and the amount of ambiguity a rule creates is often beyond our imagination and best handled by a statistical system.
2. shallow vs. deep: the designing philosophy for Pro3Gres has been to stay as shallow as possible to obtain reliable results, as explained in chapter 2.
3. Treebank constituency vs. DG: the observation that a DG that expresses grammatical relations is more informative, but also more intuitive to interpret for a non-expert, and that Functional DG can avoid a number of LDD types (see section 2.3) has made DG the formalism of our choice. For lexicalizing the grammar, a partial mapping from the largest manually annotated corpus available, the Penn Treebank, was necessary, exhibiting a number of mapping challenges.
4. history-based vs. mapping-based: Pro3Gres is not a parse-history-based approach. Instead of manually selecting what goes into the history, as is usually done (see [22] for an exception), we manually select how to linguistically meaningfully map Treebank structures onto dependency relations by the use of mapping patterns adapted from [24].

	Percentage Values for			
	Subject	Object	noun-PP	verb-PP
Precision	91	89	73	74
Recall	81	83	67	83
	Comparison to Lin (on the whole Susanne corpus)			
	Subject	Object	PP-attachment	
Precision	89	88	78	
Recall	78	72	72	
	Comparison to Buchholz [5]; and to Charniak [7], according to Preiss			
	Subject	Object		
Precision	86; 82	88; 84		
Recall	73; 70	77; 76		

Table 4: Results of evaluating the parser output on Carroll's test suite on subject, object and PP-attachment relations and a partial comparison

5. probabilistic vs. statistical: Pro3Gres is not a probabilistic system in the sense of a PCFG. From a practical viewpoint, knowing the probability of a certain rule expansion per se is of little interest. Pro3Gres models decision probabilities, the probability of a parse is understood to be the product of all the decision probabilities taken during the derivation.
6. local subtress vs. DOP: psycholinguistic experiments and Data-Oriented Parsing (DOP) [3] suggest that people store subtrees of various sizes, from two-word fragments to entire sentences. But [20] suggests that the large number of subtrees can be reduced to a compact grammar that makes DOP parsing computationally tractable. In Pro3Gres, a subset of non-local fragments which, based on linguistic intuition, are especially important, are used.
7. generative vs. structure-generating: DG generally, although generative in the sense that connected complete structures are generated, is not generative in the sense that it is always guaranteed to terminate if used for random generation of language. Since language generation is rarely random, but e.g. derived from a logical form, this is more a theoretical than a practical problem. Whenever full parsing occurs, a complete or partial hierarchical structure that follows CFG assumptions due to the employed grammar is built up for each sentence. Pro3Gres' constraint to allow each complement dependency type only once per verb can be seen as a way of rendering it generative in practice. Based on general DG assumptions, it can also be shown that Pro3Gres is a consistently DG-oriented version of the generative [14], who state that their approach, "using sister-head relationships is a way of counteracting the flatness of the grammar productions; it implicitly adds binary branching to the grammar". Adding binary branching implicitly converts the CFG rules into an ad-hoc DG.
8. Collins and Brooks vs. Hindle and Rooth: Pro3Gres can be seen as a generalisation of [12] to a sufficiently large subset of dependency relations to do full parsing, including the majority of long-distance dependencies, instead of for PP-attachment only as in [12]. Except for the distance measure and an extended backoff chain, Pro3Gres PP-attachment is thus almost identical to [12]. In order to alleviate the sparse-data problem and the dependence on a genre-specific corpus, the Penn Treebank, [23] has thus been implemented into the system. While using [23] alone considerably improves over the unlexicalized baseline, combining [23] and [12] leads to results almost identical to [12] alone on the [6] test-corpus, which admittedly comes from a domain similar to the Penn Treebank.
9. supervised vs. unsupervised: using [23] is the first unsupervised extension to Pro3Gres, but more are under way.
10. syntax vs. semantics: instead of using a back-off to tags [11], semantic classes, Wordnet for nouns and Levin classes for verbs, are used, in the hope that they better manage better to express selectional restrictions than tags. Practical experiments have shown, however, that, in accordance to [19] on head-lexicalisation, there is almost no increase in performance.

5 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. [26] 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 [6]. 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 [35]. Comparing these results to [27] and [33] as far as is possible shows that the performance of the parser is state-of-the-art (see table 4).

	LDD relations results for	
WH-Subject Precision	57/62	92 %
WH-Subject Recall	45/50	90 %
WH-Object Precision	6/10	60 %
WH-Object Recall	6/7	86 %
Anaphora of the rel. clause subject Precision	41/46	89 %
Anaphora of the rel. clause subject Recall	40/63	63 %
Passive subject Recall	132/160	83%
Precision for subject-control subjects	40/50	80%
Precision for object-control subjects	5/5	100%
Precision of $modpart$ relation	34/46	74%
Precision for topicalized verb-attached PPs	25/35	71%

Table 5: Available results for relations traditionally considered to involve LDDs

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

Table 6: Analysis of PP-Attachment Errors

The new local relations corresponding to LDDs in the Penn Treebank have been selectively evaluated as far as the annotations permit, shown in table 5. For NP traces and NP PRO, the annotation does not directly provide all the necessary data. Passivity is not currently expressed in the predicate-argument parser output, only recall values can thus be delivered. Since Carroll's annotation does not directly express control, reduced relative clauses nor the dependency direction, only reliable precision values are available in those cases. As for gerunds, neither Carroll nor the parser output retains tagging information, which makes a selective evaluation of them impossible. The fact that performance for the new local relations corresponding to LDDs is not generally lower than in the dependencies corresponding to local constituency, although they correspond to a sequence of decisions in a traditional statistical parser, indicates that our LDD approach improves parsing performance. Absolute values are given due to the low counts of these relatively rare relations.

Table 6 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.

6 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. An evaluation at the grammatical relation level shows that its performance is state-of-the-art.

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 novel, context-free way, thus offering on the one hand a parsing complexity as low as for a probabilistic parser, but on the other hand a deep-lingusitic analysis as with a type of formal grammars.

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] Rens Bod, Remko Scha, and Khalil Sima'an, editors. *Data-Oriented Parsing*. (CSLI-SCL) Center for the Study of Language and Information, Studies in Computational Linguistics. Chicago University Press, 2003.

[4] Ted Briscoe, Claire Grover, Bran Boguraev, and John Carroll. A formalism and environment for the development of a large grammar of English. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, Milan,Italy, 1987.

[5] Sabine Buchholz. *Memory-Based Grammatical Relation Finding*. Ph.D. thesis, University of Tilburg, Tilburg, Netherlands, 2002.

[6] 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.

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

[8] Hoojung Chung and Hae-Chang Rim. A new probabilistic dependency parsing model for head-final, free word order languages. *IEICE Transaction on Information & System*, E86-D, No. 11:2490–2493, 2003.

[9] 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.

[10] 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.

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

[12] 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.

[13] Ann Copestake and Dan Fickinger. An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second Conference on Language Resources and Evaluation (LREC-2000)*, Athens, Greece, 2000.

[14] Amit Dubey and Frank Keller. Probabilistic parsing for german using sister-head dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo, 2003.

[15] 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.

[16] Jason Eisner. Bilexical grammars and their cubic-time parsing algorithms. In Harry Bunt and Anton Nijholt, editors, *Advances in Probabilistic and Other Parsing Technologies*. Kluwer Academic Publishers, 2000.

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

[18] Sandiway Fong. *Computational Properties of Principle-Based Grammar Theories*. Ph.D. thesis, MIT Artificial Intelligence Lab, Cambridge, MA, 1991.

[19] Daniel Gildea. Corpus variation and parser performance. In *2001 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 167–202, Pittsburgh, PA, 2001.

[20] Joshua Goodman. Efficient parsing of DOP with PCFG-reductions. In Bod et al. [3].

[21] 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.

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

[23] Donald Hindle and Mats Rooth. Structural ambiguity and lexical relations. In *Meeting of the Association for Computational Linguistics*, pages 229–236, 1991.

[24] 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.

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

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

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

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

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

[30] 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.

[31] 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.

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

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

[34] Anoop Sarkar, Fei Xia, and Aravind Joshi. Some experiments on indicators of parsing complexity for lexicalized grammars. In *Proc. of COLING*, 2000.

[35] Gerold Schneider. Learning to disambiguate syntactic relations. *Learning and teaching (in) Computational Linguistics, Linguistik Online 17 (5/03)*, pages 117–136, 2003.

[36] Gerold Schneider. A low-complexity, broad-coverage probabilistic dependency parser for English. In *Proceedings of HLT-NAACL 2003 Student session*, Edmonton, Canada, 2003.

[37] 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.

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

[39] Eric Wehrli. *L'analyse syntaxique des langues naturelles*. Masson, Paris, 1997.