

Learning to Disambiguate Syntactic Relations

Gerold Schneider (Zurich/Geneva)

Abstract

Natural Language is highly ambiguous, on every level. This article describes a fast broad-coverage state-of-the-art parser that uses a carefully hand-written grammar and probability-based machine learning approaches on the syntactic level. It is shown in detail which statistical learning models based on Maximum-Likelihood Estimation (MLE) can support a highly developed linguistic grammar in the disambiguation process.

1 Introduction

Many extensions to text-based, data-intensive knowledge management approaches, such as Information Retrieval or Data Mining, focus on integrating the impressive recent advances in language technology. For this, they need fast, robust parsers that deliver linguistic data which is meaningful for the subsequent processing stages. This paper introduces such a parsing system and discusses some of its disambiguation techniques which are based on learning from a large syntactically annotated corpus.

The paper is organized as follows. Section 2 explains the motivations for writing the parser, and why it profits from Dependency grammar assumptions. Section 3 gives a brief introduction to the parsing system and to evaluation questions. Section 4 presents the probabilistic models and the conducted experiments in detail.

2 Current Parsing Approaches

2.1 Formal Grammar Parsers

On the one hand, a variety of parsers that are based on a formal linguistic theory have existed for a number of years. These are, to name only a few, the Alvey tools (Briscoe et al. 1987) for GPSG, Lingo (Copestake/Flickinger 2000) or Babel (Müller 1996) for HPSG, FIPS (Wehrli 1997) or PAPPI (Fong 1991) for GB, and MINIPAR (Lin 1998) or FDG (Tapanainen/Järvinen 1997) for DG. These systems generally have a very good coverage of most syntactic phenomena, but some of them suffer from run-time problems due to the complexity of the grammar or from coverage problems due to the rigidity of the grammar or the incompleteness of the lexicon. Typical formal grammar parser complexity is about $O(n^5)$, which means that parsing time for an exhaustive parse is constant to the fifth power of the number of words in a sentence.

They have in common that they are rule-based, and that scoring systems for disambiguation are heuristic, hand-written instead of learnt from real-world data. For example, in *I saw the man [PP with an umbrella]*, it is syntactically unclear whether the PP should attach to the verb or the noun. A lexicon which is enough specific to include the information that one cannot see by means of umbrellas is too cumbersome to compile, and structural heuristics have to fail since the structurally potentially equivalent *I saw the man [PP with a periscope]* is semantically rather a case for verb-attachment. A corpus-based probabilistic approach, however, based on whether it is more likely to see the participating words in a verb- or a noun-attachment, can prefer and reject readings based on empirical grounds (Collins/Brooks 1995).

2.2 Probabilistic Parsers

On the other hand, broad-coverage syntactic parsers that learn from syntactically annotated corpora have now become available (Charniak 2000, Collins 1999, Henderson 2003). They generally have good performance, but they typically produce pure constituency data as output, trees that do not include the deep-linguistic information, i.e. grammatical function annotation nor the empty nodes annotation provided in Treebanks such as the Penn Treebank (Marcus/Santorini/Marcinkiewicz 1993) on which they are usually trained. This means that the extraction of long-distance dependencies (LDD) and the mapping to shallow semantic representations is not always possible, because first co-indexation information is not available, second a single parsing error across a tree fragment containing an LDD makes its extraction impossible (Johnson 2002), third some syntactic relations cannot be recovered on configurational grounds only.

Implementations of probabilistic parsers are very efficient since no LDDs are expressed. The CYK algorithm, or versions of it such as Nivre (2003), have parsing complexity $O(n^3)$.

The parser described here aims at combining both approaches into a hybrid: using a deep-linguistic formal grammar theory, dependency grammar (DG), with hand-written rules, but assigning probabilistic lexicalized scores for disambiguation and pruning. While it profits from the low parsing complexity of a CYK implementation, it expresses most LDDs like a formal grammar, as briefly explained in the following (see Schneider 2003b for a detailed discussion).

2.3 Functional Dependency Structures

Dependency Grammar (DG) is essentially a valency grammar in which the valency concept is extended from verbs to nouns and adjectives and finally to all word classes.

In its simplest definition, a projective DG is a binary version of a constituent grammar which only knows lexical items, which entails that

- for every mother node, the mother node and exactly one of its daughters, the so-called head, are isomorphic
- projection is deterministic, endocentric and can thus not fail, which gives DG a robustness advantage

- equivalent constituency CFG trees can be derived, as illustrated in figure 1

The last point deserves elaboration. When converting dependency structures to constituency trees, very flat trees are produced. In figure 1, the top V node corresponding to S and its head daughter corresponding to VP are intrinsically identical, only the deliberate choice of an ordering rule that attaches subjects higher than objects leads to a common constituency representation. In this sense, constituency structures are more expressive than dependency structures. On the other hand, the typical dependency syntactic function label is lost in a conversion to common constituency representation.

DG was originally conceived to be a deep-syntactic, proto-semantic theory (Tesnière 1959). The version of DG used here retains syntactic functions as dependency labels, like in LFG, which means that the dependency analyses returned by the parser are also a simple version of LFG f-structures, a hierarchy of syntactic relations between lexical heads which serves as a bridgehead to semantics.

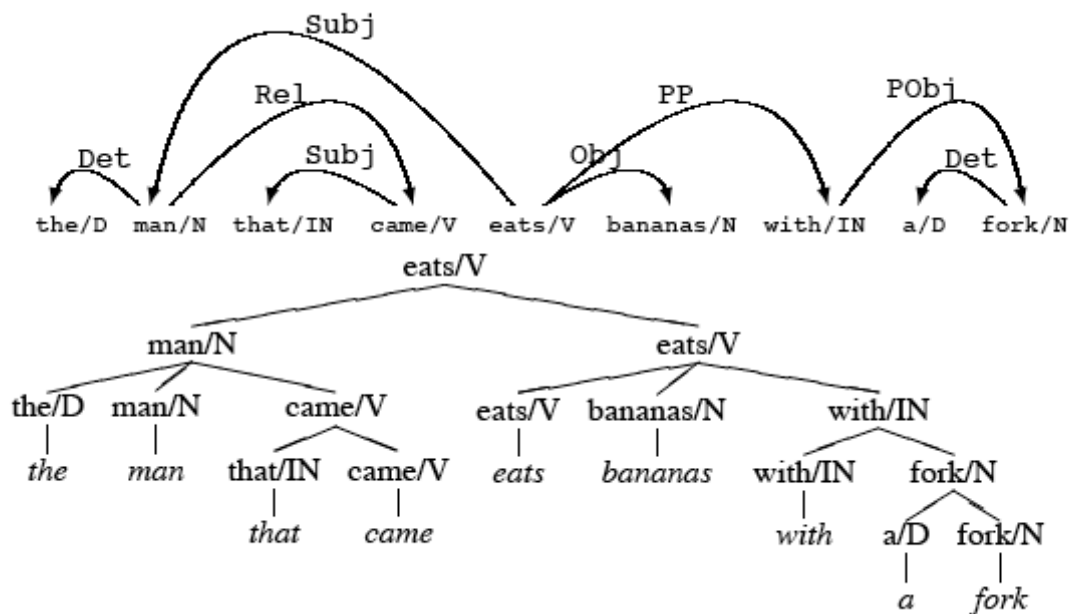


Figure 1: A dependency representation and its typically unlabeled constituency counterpart

They are close to predicate-argument structures or other surface semantic representations such as discourse representation theory (DRT). They are also easy to read and interpret as they are based on traditional school grammar terms and far less nested than constituency-based theories, e.g. GB (Chomsky 1981). Unlike the latter, functional dependency exhibits only a minimal amount of non-projectivity. Non-projectivity, which is typically expressed by transformations, co-indexation or structure-sharing, is absent from functional dependency except for WH-movement, due to the following reasons.

Empty units, empty complementizers and empty relative pronouns pose no problem to DG as they are non-head material. Functional DG only accepts content words as heads. For example, a complementizer is an optional dependent of the subordinated verb. Moved clauses like PPs or clausal complements of verbs of utterance (e.g. *Torrential rainfalls are forecast, said the*

weatherman) do not involve non-local dependencies but are expressed by a non-canonical dependency direction under well-defined conditions. The same applies for subject-verb inversion in affirmative clauses for verbs of utterance (same example). Noun-participle relations (e.g. *the report written by ...* are covered by a dedicated syntactic relation expressing the noun-modification and the object character of the noun at the same time. Infinite verbs and gerunds may act as subjects (e.g. *after winning/VBG the race*), which are covered by grammar rules that Tesnière called translations, although these rules do not participate in the probability model. Finally, non-surface semantic arguments, for example the subject-verb relation in the subordinate clause of a control construction, are created at the post-parsing stage, where minimal predicate-argument structures are output.

Tables 1 and 2, which are adapted from Johnson (2002) show that the vast majority of LDDs except for WH-movement, i.e. dependencies involving several local subtrees and thus several CFG rules are covered by strictly local rules expressing local dependencies by the functional DG grammar. Most of the relations expressed by these rules also participate in the machine learning model described in section 4. The numbers in table 2 show how many occurrences were recognised by the patterns described in section 3.3. Because the patterns are formulated with an emphasis on precision, generally not all occurrences of a type are extracted, recall is not 100 %. Many of the structures that are not covered by the patterns are still parsed correctly, for example adverbial sentences as unspecified subordinate clauses, but information about shared semantic arguments is no longer expressed.

	<i>ANTECEDENT</i>	<i>POS</i>	<i>LABEL</i>	<i>COUNTS SECTIONS 1-21</i>	<i>DESCRIPTION</i>
1	NP	NP	*	18,334	NP trace
2		NP	*	9,812	NP PRO
3	WHNP	NP	*T*	8,620	WH trace
4			*U*	7,478	Empty units
5			0	5,635	Empty complementizers
6	S	S	*T*	4,063	Moved clauses
7	WHADVP	ADVP	*T*	2,492	WH-trace
8		SBAR		2,033	Empty clauses
9		WHNP	0	1,759	Empty relative pronouns
10		WHADVP	0	575	Empty relative pronouns

Table 1: The distribution of the 10 most frequent types of empty nodes and their antecedents in the Penn Treebank

<i>TYPE</i>	<i>COUNT</i>	<i>PROB-MODELED</i>
passive subject	7,126	YES
indexed gerund	4,430	NO
inverted NP-V	2,427	YES
control, raise, semi-aux	2,284	YES
small clause object	452	YES
others / not covered	6,015	
TOTAL in sections 0-24	22,734	

Table 2: Coverage of the patterns for the most frequent NP traces

3 The Parsing System

The parser differs on the one hand from successful deep-linguistic Dependency Grammar implementations (e.g. Lin 1998, Tapanainen/Järvinen 1997) by using a statistical base, and on the other hand from state-of-the-art statistical approaches (e.g. Collins 1999) by carefully following an established formal grammar theory, Dependency Grammar (DG). The parser has been implemented in Prolog. It has been tested in SWI-Prolog and Sicstus Prolog under Solaris, Linux, Windows 2000 and Windows XP. Figure 2 shows a screenshot of the graphical SWI-Prolog version. Its parsing speed, including the tagging and chunking preprocessing, is above 100'000 words per hour on a modern PC.

The parser uses a hand-written linguistic grammar. Except for the pre-processing tagger, this part of the system is rule-based and non-probabilistic. But the parser also integrates a probability-based model of language, similar to Collins (1999). It is supervised and based on Maximum Likelihood Estimation (MLE). All the probability models tested are discussed in section 4.

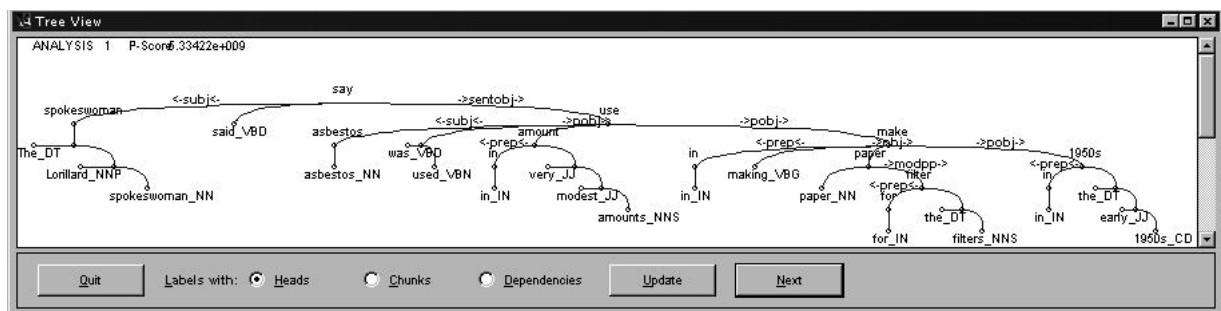


Figure 2: Tree Display window of the SWI Prolog graphical implementation of the parser

3.1 Tagging and Chunking

The parsing system uses a divide-and-conquer approach. 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 by means of chunking. Finite-state technology is fast enough for unlimited amounts of data. Tagging and chunking is done by a standard tagger and chunker, *LTPos* (Mikheev 1997). The parser then relies on the disambiguation decisions of the tagging and chunking stage and can profit from a reduced search space, at the cost of a slightly decreased

performance due to tagging and chunking errors. The grammar rules are based on the tags of the linguistic heads.

Tagging is a classical machine learning application in Computational Linguistics. Every word token in the running text is assigned a part-of-speech tag depending on its context. For example, the word *run* is a verb token in *I run*, as in most cases when it follows a pronoun, but a noun token in *the long run*, as in most cases when it follows an adjective. Taggers achieve error rates of 2-5%. They often fail if the context that is taken into consideration around the target word is too small for a decision to be taken or if structural information would be needed. For example in *the article featured in the newspaper discusses that ...* the verb participle (e.g. *appeared*) is likely to be mistagged as (the generally more frequent) verb past tense, because their immediate left (e.g. a noun) and right (e.g. a preposition) context can be identical. Some frequent tagging errors can be corrected by the parser. For example, the grammar allows verb past forms to act as participles, which leads to local but rarely to global parsing ambiguity.

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 (Minnen/Carroll/Pearce 2000). Parsing takes place only between the heads of phrases, and only using the best tag suggested by the tagger, which leads to a reduction in complexity.

3.2 The Hand-Written Grammar

Because of the DG isomorphism of a head and its projection, there are no phrase levels, and projection can never fail. All rules apply at the word level, and are based on the Penn Treebank tagset tags.

The rules contain a simple, tag-based scoring system which disprefers marginal and infrequent relations. A large number of linguistic constraints, such that a verb can only have one subject, that adjuncts only follow after all complements, that verbs can maximally have two noun objects etc. are encoded. Some linguistically possible but very rare constructions are ruled out. For example, a verb with a subordinate clause is not allowed to attach a PP after the subordinate clause. Linguistic constructions that are considerably rarer than the error rate they introduce are not desirable in a practically oriented system. A closed list of temporal expressions, possible noun adjuncts, is also used. Only verbs of utterance are allowed to undergo subject-verb inversion in affirmative clauses, and topicalized clauses are only allowed sentence-initially.

Writing grammar rules is a feasible task when using a framework that is close to traditional school grammar assumptions, such as DG. Acknowledged facts are expressed in hand-written declarative rules. Since the tagset is limited and dependency rules are binary, even a broad-coverage set of rules can be written in relatively little time.

Much more difficult than grammar writing is to assess the scope of application of a rule and the amount of ambiguity it creates. Long real-world sentences typically have dozens to hundreds of syntactically correct complete analyses and thousands of partial analyses, although most of them are semantically so odd that one would never think of them. Here, machine-learning approaches, such as probabilizing the manually written rules, are very useful

for a successful parser, for two reasons: first, the syntactically possible analyses can be ranked according to their probabilities. For subsequent processing stages like semantic interpretation or document classification it then often suffices to take the first ranked or the n first ranked readings. Second, in the course of the parsing process, very improbable analyses can be abandoned, which greatly improves parsing efficiency.

3.3 Mapping and Evaluation

Since the parser does not use the more common Treebank constituency structures, both mapping Treebank structures to functional dependency for training and evaluating the output of the parser is non-trivial.

As for the mapping from Treebank trees to functional dependencies, configurational patterns corresponding to the individual syntactic relations have been devised. As some of these express LDDs (see section 2.3), grammatical role labels, empty node labels and tree configurations spanning several local subtrees are used as integral part of some of these patterns, for example in the noun-participle relation *modpart*. Some relations include local alongside non-local dependencies. For example, the *object* relation includes copular verb complements and small clause complements. The *subject* relation is different for active and passive verbs. Because the verb form allows a clear identification of passive clauses, a surface analysis is sufficient, since two distinct probability models are used. Table 3 shows the most important relation types.

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

Table 3: The most important dependency types used by the parser

The patterns are applied to sections 2-24 of the Penn Treebank, the lexical heads are extracted and then used for training. These patterns are optimized for precision, they do not reach 100 % recall. A first evaluation used the extracted, held-out heads from section 0 as gold standard. The results reported on parsing section 0 are about 5 % to 10 % too low, however, as the patterns do not have full recall (Schneider 2003).

In traditional constituency approaches, parser evaluation is done in terms of the correspondence of the bracketing between the gold standard and the parser output. Lin (1995) 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 (Carroll/Minnen/Briscoe 1999). 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 according to the following mapping scheme. R_c is a syntactic relation defined by Carroll, non-subscript relations are the functional relations of the parser.

Subject	$\left\{ \begin{array}{l} \text{Precision: } \textit{subj} \text{ OR } \textit{modpart} \quad \rightarrow \quad \textit{ncsubj}_C \text{ OR } \textit{cmod}_C(\text{with rel.pro}) \\ \text{Recall: } \textit{ncsubj}_C \quad \rightarrow \quad \textit{subj} \text{ OR } \textit{modpart} \\ \text{LEGEND: } \textit{ncsubj}_C = \text{non-clausal subject} \\ \textit{cmod}_C = \text{clausal modification, used for relative clauses} \end{array} \right.$
Object	$\left\{ \begin{array}{l} \text{Precision: } \textit{obj} \text{ OR } \textit{obj2} \quad \rightarrow \quad \textit{dobj}_C \text{ OR } \textit{obj2}_C \\ \text{Recall: } \textit{dobj}_C \text{ OR } \textit{obj2}_C \quad \rightarrow \quad \textit{obj} \text{ OR } \textit{obj2} \\ \text{LEGEND: } \textit{dobj}_C = \text{first object} \\ \textit{obj2}_C = \text{second object} \end{array} \right.$
noun-PP	$\left\{ \begin{array}{l} \text{Precision: } \textit{modpp} \quad \rightarrow \quad \textit{ncmod}_C(\text{with prep}) \text{ OR } \\ \textit{xmod}_C(\text{with prep}) \\ \text{Recall: } \textit{ncmod}_C(\text{with prep}) \text{ OR } \quad \rightarrow \quad \textit{modpp} \\ \textit{xmod}_C(\text{with prep}) \\ \text{LEGEND: } \textit{ncmod}_C = \text{non-clausal modification} \\ \textit{xmod}_C = \text{clausal modification for verb-to-noun translations} \end{array} \right.$
verb-PP	$\left\{ \begin{array}{l} \text{Precision: } \textit{pobj} \quad \rightarrow \quad \textit{iobj}_C(\text{with prep}) \text{ OR } \\ \textit{arg_mod}_C \text{ OR } \\ \textit{ncmod}_C(\text{with prep OR (prt \& dobj)}) \text{ OR } \\ \textit{xcomp}_C(\text{with prep}) \text{ OR } \\ \textit{xmod}_C(\text{with prep}) \\ \text{Recall: } \textit{iobj}_C(\text{with prep}) \text{ OR } \quad \rightarrow \quad \textit{pobj} \\ \textit{arg_mod}_C \text{ OR } \\ \text{LEGEND: } \textit{iobj}_C = \text{indirect object, } \textit{arg_mod}_C = \text{passive agent} \\ \textit{xcomp}_C \text{ for PP-attachment to copular verbs} \end{array} \right.$

Let us look at the subject case: a surface subject relation is expressed by the parser as *subj* or as *modpart*. From the precision perspective, they are correct if they map to a relation in Carroll's annotation that expresses subjecthood. In most cases, that will be *ncsubj_c*. But subjects of relative clauses (the relative pronoun) are found in the *cmod_c* relation (clausal modification). Since not all *cmod_c* relations express subjecthood, but include many other cases of clause subordination, not every *cmod_c* can be expected to map to *subj* or to *modpart* from a recall perspective. Therefore, only the "prototypical" nominal subject, *ncmod_c*, is required to map onto a subjecthood relation of the parser in order to show correct recall.

In the object case, no distinction between direct and indirect objects are made. In the noun-PP attachment case, the parser's *modpp* maps to *ncmod_c* if the head of the description noun is a noun, and to *xmod_c* if the head of the description noun is a gerund or an infinitival relative.

In the verb-PP attachment case, *xmod_c* also expresses purpose infinitives, which are included in the parser's *sentobj* relation rather than *pobj*. *ncmod_c* expresses adjuncts both of verbs and nouns, since no tagging information is available on Carroll's annotation, no distinction is

possible. The vast majority of $ncmod_c$ governors are nouns, where they are not the noun-PP recall metric returns a spurious error. But since $ncmod_c$ is by far the major noun-PP attachment relation, it cannot be excluded.

The grammar contains a simple scoring system that disprefers a few rare grammatical relations. There is only one case in which information beyond structural preferences is used by the Baseline System: PPs are attached to the verb or the noun based on the lexical statistical preference of the preposition. Since prepositions are a closed class, this small amount of lexical preference can easily be coded in a hand-written rule-based grammar.

A small number of closed-class words have rules based on the word instead of on the tag. These are complementizers, the English postpositions *ago*, *later* and *after*, and a few gerunds (e.g. *including*) that can act as prepositions.

4 Probability-Based Disambiguation Learning

In the Baseline System, dependencies can span unlimited distances, subject to the fact that dependencies are not allowed to cross each other, and lexical information is not taken into account. Table 4 shows the performance, which is, as expected, rather poor. The values for verb-PP-attachment are especially low because ambiguous attachments get equal weight, which means that whether a verb- or noun-PP-attachment is ranked as the first parse depends on chance factors: The CYK algorithm finds the closer noun attachment first, but the module that finds the best path through a sentence is stack-driven, starting with the last found analyses and preferring them if all analyses have equal weight, which leads to a strong and linguistically incorrect preference for verb-PP-attachment.

<i>PRECISION AND RECALL MEASURES</i>		
subj_prec	801 of 936	85.5 %
subj_recall	752 of 956	78.6 %
obj_prec	333 of 398	83.6 %
obj_recall	257 of 391	65.7 %
nounpp_prec	155 of 227	68.2 %
verbpp_prec	275 of 722	38.0 %
ncmod_recall	610 of 801	76.1 %
iobj_recall	87 of 157	55.4 %
argmod_recall	21 of 41	51.2 %

Table 4: Results of evaluating the Baseline System output on Carroll's test suite on subject, object and PP-attachment relations

Distance information and detailed lexical information are cumbersome to include in a grammar, and if based on hand-written, arm-chair linguistic heuristics are often unreliable. The use of empirical measures is desirable.

4.1 The Distance Measure

The distance between a head and a dependent is a limiting factor for the probability of a dependency between them. Not all relations have the same typical distances, however. While objects are most frequently immediately following the verb, a PP attached to the verb may easily follow only at the second or third position, after the object and other PPs etc. A relation-specific simple MLE estimation is thus employed to prefer typical distances. Distance is measured in chunks.

Formula 1 shows the classical MLE calculation: the probability of a certain *Distance* across which to span, given the relation *REL*, corresponds to the corpus count of the instances of relation *REL* that span this distance divided by the count of all instances of relation *REL*.

$$p(\textit{Distance}|\textit{REL}) = \frac{\#(\textit{REL} \wedge \textit{Distance})}{\#\textit{REL}} \quad (1)$$

The Distance Measure leads to a tremendous increase in performance, as shown in table 5, which confirms our intuitions. To a lesser degree, the algorithm's preference for verb-PP attachment is still apparent.

<i>PRECISION AND RECALL MEASURES</i>		
subj_prec	818 of 942	86.8 %
subj_recall	758 of 956	79.2 %
obj_prec	427 of 489	87.3 %
obj_recall	316 of 391	80.8 %
nounpp_prec	322 of 450	71.5 %
verbpp_prec	360 of 515	69.9 %
ncmod_recall	599 of 801	74.7 %
iobj_recall	131 of 157	83.4 %
argmod_recall	29 of 41	70.7 %

Table 5: Results of evaluating the Distance Measure Only System output on Carroll's test suite

4.2 The fully Lexicalised, Backed-Off Model

Let us come back to the PP-attachment examples discussed in section 2:

- (1) I saw the man with an umbrella
- (2) I saw the man with a periscope

Intuitively, it is not so much the distance, much rather the participating words that seem to influence the attachment preference. Collins/Brooks (1995) introduce such a lexicalized PP-

attachment algorithm, which has been adapted and extended, on the one hand to all important syntactic relations, and on the other hand a more refined back-off procedure is used.

Given two adjacent lexical heads (say a and b), the probabilities of the possible dependency relations between them are calculated as Maximum Likelihood (MLE) estimates. In a binary CFG, constituents which are adjacent at some stage in the parsing process are candidates for the right-hand side (RHS) of a rewrite rule. If a rule exists for these constituents (say A and B), then in DG one of these is isomorphic to the LHS, i.e. the head. DG rules additionally use a syntactic relation label R , for which the probabilities are calculated in this probability model. The dependency rules used are based on Treebank tags, the relation probabilities are conditioned on them and on the lexical heads.

$$p(R|A \rightarrow AB, a, b) = \frac{\#(R, A \rightarrow AB, a, b)}{\#(A \rightarrow AB, a, b)} \quad (2)$$

All that $A \rightarrow AB$ expresses is that in the dependency relation the dependency is towards the right, it is therefore rewritten as *right*.

$$p(R|right, a, b) = \frac{\#(R, right, a, b)}{\#(right, a, b)} \quad (3)$$

The PP-attachment model probabilities are conditioned on three lexical heads - the verb, the preposition and the description noun and backed-off (Collins/Brooks 1995). Backing-off generally means that less specific information is used when no fully specified information is available. In the example of PP-attachment it means that if an MLE estimation for a triple (verb, preposition, description noun) is not possible because it was never seen in the training corpus, the least significant lexical participant, the description noun, is discarded and an MLE estimation for the less specific tuple (verb, preposition) is sought, and if that still yields no estimation just the preposition is used.

The probability model used here is backed off across more levels than in Collins/Brooks (1995). Before discarding lexical participants, semantic classes are also used in all the modeled relations, for verbs the Levin classes (Levin 1993), for nouns the top Wordnet class (Fellbaum 1998) of the most frequent sense.

4.2.1 An Example: Modification by Participle

The noun-participle relation is also known as reduced relative clause. In the Treebank, reduced relative clauses are adjoined to the NP they modify, and under certain conditions also have an explicit *RRC* label. Reduced relative clauses are frequent enough to warrant a probabilistic treatment, but considerably sparser than verb-non-passive-subject or verb-object relations. They are in direct competition with the subject-verb relation, because its candidates are also a NP followed by a VP. We probably have a subject-verb relation in *the report announced the deal* and a noun-participle relation in *the report announced yesterday*. The majority of modification by participle relations, if the participle is a past participle, functionally

correspond to passive constructions (*the report written* \square *the report which has been written*). In order to reduce data sparseness, which could lead to giving preference to a verb-non-passive-subject reading (*asubj*), the verb-passive-subject counts (*psubj*) are added to the noun-participle counts. Some past participles also express adjunct readings (*the week ended Friday*); therefore the converse, i.e. adding noun-participle counts to verb-passive-subject counts, is not recommended. Since the *modpart* relation is always to the right, this parameter is neglected.

The next back-off step maps the noun *a* to its Wordnet-class \hat{a} and the verb *b* to its Levin-class \hat{b} . If the counts are still zero, counts on only the verb and then only the noun are used.

$$p(\text{modpart}|a, b) = \tag{4}$$

$$\frac{\#(\text{modpart, right, } a, b) + \#(\text{psubj, left, } a, b)}{\#(\text{modpart, right, } a, b) + \#(\text{psubj, left, } a, b) + \#(\text{asubj, left, } a, b)}$$

if > 0, *else*

$$\frac{\#(\text{modpart, right, } \hat{a}, \hat{b}) + \#(\text{psubj, left, } \hat{a}, \hat{b})}{\#(\text{modpart, right, } \hat{a}, \hat{b}) + \#(\text{psubj, left, } \hat{a}, \hat{b}) + \#(\text{asubj, left, } \hat{a}, \hat{b})}$$

if > 0, *else*

$$\frac{\#(\text{modpart, right, } b) + \#(\text{psubj, left, } b)}{\#(\text{modpart, right, } b) + \#(\text{psubj, left, } b) + \#(\text{asubj, left, } b)}$$

if > 0, *else*

$$\frac{\#(\text{modpart, right, } a) + \#(\text{psubj, left, } a)}{\#(\text{modpart, right, } a) + \#(\text{psubj, left, } a) + \#(\text{asubj, left, } a)}$$

As the last backoff, a low non-zero probability is assigned. In the verb-adjunct relation, which drastically increases complexity but can only occur with a closed class of nouns (mostly adverbial expressions of time), this last backoff is not used.

The evaluation shows a clear improvement over the Distance Measure Only system, although considering the elaboration of the model, the improvement is modest.

Comparing these results, shown in table 6, to other parsers that have been tested on syntactic relations shows that it is state-of-the-art (Preiss 2003). Although better results are reported for PP-attachment disambiguation in isolation (e.g. Collins/Brooks 1995), the same results cannot be expected to be found in the context of real parsing. On the one hand, some PP-attachments are unambiguous, which should lead to better results. On the other hand, some PP-attachments are multiply ambiguous (they have more than two possible attachment sites) or occur fronted in a sentence-initial position, or a participant in the PP-relation is mistagged or mischunked. Mistagging and mischunking is involved in about 20-25 % of the PP-attachment precision errors.

<i>PRECISION AND RECALL MEASURES</i>		
subj_prec	828 of 946	87.5 %
subj_recall	767 of 956	80.2 %
obj_prec	430 of 490	87.7 %
obj_recall	316 of 391	80.8 %
nounpp_prec	343 of 479	71.6 %
verbpp_prec	350 of 482	72.6 %
ncmod_recall	593 of 801	74.0 %
iobj_recall	132 of 157	84.0 %
argmod_recall	30 of 41	73.1 %

Table 6: Results of evaluating the Fully Lexicalized, Backed-Off System output on Carroll's test suite

4.3 Head-Lexicalisation

In order to take a closer look at the backoff model it has been investigated down to which level the backoff model descends until a decision can be taken. Table 7 reveals that full count decisions, especially in the case of PP-attachment, are relatively rare. This is due to sparse data, most pronouncedly in noun-PP-attachment, where a Zipfian head noun as well as a Zipfian description noun (the noun inside the PP) is involved.

The values at level 6 and 7 of verb-PP-attachment are very high because there are prepositions that are very rarely verb-attaching (e.g. *of*) and because the Penn Treebank preposition tag (*IN*) is also used for complementizers. In these cases, very low MLE counts are only found at the preposition-backoff level (6) or not at all (7), in which case a low non-zero probability is assigned.

Since low full counts can be unreliable, and since the class-based backoff does not include Word Sense Disambiguation, it is questionable how trustworthy the MLE counts at these levels are. A system that only lexicalizes the head of the relation, respectively the head and the preposition in PP-attachment, has been tested. It turns out that its performance is virtually equivalent if not marginally better than that of the full model, as table 8 shows.

BACKOFF DECISION POINTS			
<i>subj</i>	0	full	377
	1	verbclass & nounclass	530
	2	verb	384
	3	noun	41
	4	NONE	15
<i>obj</i>	0	full	437
	1	verb & nounclass	939
	2	verbclass & noun	32
	3	verbclass & nounclass	145
	4	verb	92
	5	noun	40
	6	NONE	12
<i>pobj</i>	0	full	124
	1	verb & prep & nounclass	2624
	2	verb & prep	2631
	3	verbclass & prep & noun	337
	4	verbclass & prep & nounclass	5004
	5	prep & noun	995
	6	prep	4762
	7	NONE	4747
<i>modpp</i>	0	full	30
	1	noun & prep & descnounclass	197
	2	nounclass & prep & descnoun	100
	3	noun & prep	208
	4	nounclass & prep	696
	5	prep & descnoun	73
	6	prep	227
	7	NONE	281
<i>modpart</i>	0	full	0
	1	nounclass & verbclass	144
	2	verb	45
	3	noun	7
	4	NONE	11

Table 7: Backoff decision points for the Fully Lexicalized, Backed-Off System on Carroll's test suite

<i>PRECISION AND RECALL MEASURES</i>		
subj_prec	829 of 948	87.4 %
subj_recall	769 of 956	80.4 %
obj_prec	429 of 488	87.9 %
obj_recall	317 of 391	81.0 %
nounpp_prec	346 of 473	73.1 %
verbpp_prec	356 of 488	72.9 %
ncmod_recall	598 of 801	74.6 %
iobj_recall	133 of 157	84.7 %
argmod_recall	31 of 41	75.6 %

Table 8: Results of evaluating the Head-Lexicalized System output on Carroll's test suite

4.4 Example-Based Extensions against Sparse Data

In order to fight the sparse data problem, a number of extensions have been tested. The first two are example-based.

Since a head places strong selectional restrictions on its dependent(s) of the same head, or heads with the same dependent are often similar. This fact can be exploited for Word Sense Disambiguation (Lin 1997). Including all counts with the same head if head-dependent pair has a zero count would simply amount to using the next backoff level. The fact that a dependent also places restriction on a head is thus taken into consideration. For a target zero-count head-dependent pair, if non-zero counts are found for both

1. a head'-dependent,
2. a head- dependent' and
3. a head'-dependent'

(where head' and dependent' are any word of the same tag), then their MLE counts are used. In a more restrictive version, only dependent' of the same noun class or verb class are allowed. Versions that allow one or two of the above pairs to originate from a chunked, syntactically approximated, much larger, unannotated corpus, the Reuters-21578 newswire corpus, and then only using the MLE counts of the remaining Penn Treebank pairs, have also been tested. All of them shows similar, generally very slightly lower performance than the full or the Head-Lexicalized model. An analysis of the decision points shows that non-zero values at between 2 and 10 times the original full level can be obtained, but the unreliability of the similarity and the increased coverage seem to level each other out.

4.5 The PP-Interaction Model

For the verb-prepositional-phrase relation, two models that take the interaction between the several PPs of the same verb into account have been implemented. They are based on the verbal head and the prepositions.

The first one estimates the probability of attaching a PP introduced by preposition p_2 , given that the verb to which it could be attached already has another PP introduced by the preposition p_1 . Back-offs using the verb-class and then the preposition(s) only are used.

$$\begin{aligned}
 p(p2|v, p1) = & \frac{\#(p2,v,p1)}{\#(v,p1)} \text{ if } > 0, \text{ else} \\
 & \frac{\#(p2,\dot{v},p1)}{\#(\dot{v},p1)} \text{ if } > 0, \text{ else} \\
 & \frac{\#(p2,\cup v,p1)}{\#(\cup v,p1)} \text{ if } > 0, \text{ else} \\
 & \frac{\#(p2,\cup v)}{\#(\cup v)}
 \end{aligned}
 \tag{5}$$

The second model estimates the probability of attaching a PP introduced by preposition $p2$ as a non-first PP. The usual backoffs are not printed here.

$$p(p2|v, \cup p1) = \frac{\#(p2,v,\cup p1)}{\#(v,\cup p1)}
 \tag{6}$$

As prepositions are a closed class, a zero probability is assigned if the last back-offs fail.

The evaluation, not printed for space reasons, shows that using the PP-interaction model leads to results that are equivalent or very slightly worse.

4.6 The VP Expansion Model

Verbs often have several dependents. Ditransive verbs, for example, have up to three NP complements, the subject, the direct object and the indirect object. An indeterminate number of adjuncts can be added. Transitivity, expressed by a verb's subcategorization, is strongly lexicalized. But because the Treebank does not distinguish arguments and complements, and because a standard lexicon does not contain probabilistic subcategorization, a probabilistic model has advantages. Dependency models as discussed hitherto fail to model complex dependencies between the dependents of the same mother, unlike PCFGs. A simple PCFG model for the production of the VP rule which is lexicalized on the VP head and has a non-lexicalized backoff, is therefore used. RHS constituents C , for the time being, are unlexicalized phrasal categories like *NP*, *PP*, *Comma*, etc. At some stage in the parsing process, given an attachment candidate C_n and a verbal head v which already has attached constituents C_1 to C_{n-1} , the probability of attaching C_n is estimated. This probability can also be seen as the probability of continuing versus ending the VP under production.

The evaluation, not printed for space reasons, shows results that are almost identical to the corresponding models without the VP expansion models.

4.7 Linear Interpolation

Observing that the head-lexicalized and the full, but sparse model have nearly equal performance suggests that a combination of them may be useful. A linear interpolation version of the Fully Lexicalized, Backed-Off system has been implemented. Instead of only considering the first non-zero value in the back-off chain, all subsequent back-off level probabilities are also taken into consideration and an unweighted average is used.

The result (see table 9) is between the Head-Lexicalized and the full model, with the provision that differences between all 3 models are marginal.

<i>PRECISION AND RECALL MEASURES</i>		
subj_prec	828 of 946	87.5 %
subj_recall	767 of 956	80.2 %
obj_prec	432 of 492	87.8 %
obj_recall	318 of 391	81.3 %
nounpp_prec	353 of 490	72.0 %
verbpp_prec	345 of 466	74.0 %
ncmod_recall	595 of 801	74.2 %
iobj_recall	131 of 157	83.5 %
argmod_recall	28 of 41	68.3 %

Table 9: Results of evaluating the Interpolated System output on Carroll's test suite on subject, object and PP-attachment relations

5 Conclusions

A state-of-the-art parser has been described and some of its machine learning techniques analyzed in detail. The performance of a disambiguation system based on syntactic structure clues only is quite low. While a probabilistic distance measure and head-lexicalization lead to considerable parsing performance improvements, more elaborate extensions are seen to have extremely little or no effect.

References

- Briscoe, E./Grover, C./Boguraev, B./Carroll, J. (1987): "A formalism and environment for the development of a large grammar of english". In: MacDermott, John (ed.): *Proceedings of the 10th International Joint Conference on Artificial Intelligence, Milan, Italy*. Los Altos, Calif.: 703-708.
- Carroll, John/Minnen, Guido/Briscoe, Ted (1999): "Corpus annotation for parser evaluation". In: Uszkoreit, H./Brants, T./Krenn, B. (eds): *Proceedings of the EACL-99 Post-Conference Workshop on Linguistically Interpreted Corpora*. Bergen, Norway: 35-41.
- Charniak, Eugene (2000): "A maximum-entropy-inspired parser". In: *Proceedings of the North American Chapter of the ACL*. Seattle, WA: 132-139.
- Chomsky, Noam (1981): *Lectures on Government and Binding*. Dordrecht.
- Collins, Michael (1999): *Head-Driven Statistical Models for Natural Language Parsing*. Ph.d. dissertation, University of Pennsylvania, Philadelphia, PA.
- Collins, Michael/Brooks, James (1995): "Prepositional attachment through a backed-off model". In: Yarowsky, David (ed.): *Proceedings of the Third Workshop on Very Large Corpora*. Cambridge, MA: 27-38.
- Copestake, Ann/Flickinger, Dan (2000): "An open-source grammar development environment and broad-coverage english grammar using hpsg". In: Gavrilidou, Maria et al. (eds.): *Proceedings of the Second conference on Language Resources and Evaluation (LREC-2000), Athens, Greece*. Paris: 591-600.

- Fellbaum, Christiane (ed.) (1998): *WordNet: An Electronic Lexical Database*. Cambridge, MA.
- Fong, Sandiway (1991): *Computational Properties of Principle-Based Grammatical Theories*. Ph.d. dissertation, MIT Artificial Intelligence Lab, Cambridge, MA.
- Henderson, James (2003): "Inducing History Representations for Broad Coverage Statistical Parsing". In: *Proceedings of the joint meeting of the North American Chapter of the Association for Computational Linguistics and the Human Language Technology Conference (HLT-NAACL 2003)*. Edmonton, Canada: 103-110.
- Johnson, Mark (2002): "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*: 136-143.
see <http://cog.brown.edu:16080/~mj/Publications.htm>
- Levin, Beth C (1993): *English Verb Classes and Alternations: a Preliminary Investigation*. Chicago, IL.
- Lin, Dekang (1995): "A dependency-based method for evaluating broad-coverage parsers". In: Mellish, Chris S. (ed.): *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, Montréal, Canada (IJCAI-95)*. San Mateo, Calif.: 1420-1425.
- Lin, Dekang (1997): "Using syntactic dependency as local context to resolve word sense ambiguity". In: *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics (ACL/EACL-97)*. Madrid.: 64-71.
see <http://www.cs.ualberta.ca/~lindek/papers.htm>
- Lin, Dekang (1998): "Dependency-based evaluation of MINIPAR". In: Rubio, Antonio et al. (eds.): *Proceedings of the Workshop on the Evaluation of Parsing Systems, First International Conference on Language Resources and Evaluation (LREC '98), Granada, Spain*. Paris.
see <http://www.cs.ualberta.ca/~lindek/papers.htm>
- Marcus, Mitch/Santorini, Beatrice/Marcinkiewicz, M.A. (1993): "Building a large annotated corpus of English: the Penn Treebank". *Computational Linguistics* 19: 313-330.
- Mikheev, Andrei (1997): "Automatic rule induction for unknown word guessing". *Computational Linguistics* 23(3): 405-423.
- Minnen, Guido/Carroll, John/Pearce, Darren (2000): "Applied morphological generation". In: *Proceedings of the 1st International Natural Language Generation Conference (INLG 2000)*. Mitzpe Ramon, Israel: 201-208.
see <http://www.cogs.susx.ac.uk/lab/nlp/carroll/abs/00mcp.html>
- Müller, Stefan (1996): "The Babel-System-an HPSG Prolog implementation". In: Reintjes, Peter (ed.): *Proceedings of the Fourth International Conference on the Practical Application of Prolog, London*. Blackpool, Lancashire: 263-277.
- Nivre, Joakim (2003): "An Efficient Algorithm for Projective Dependency Parsing". In: *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*. Nancy: 149-160.
see http://www.masda.vxu.se/~nivre/research/cv_eng.html#publ

- Preiss, Judita (2003): "Using grammatical relations to compare parsers". In: *Proceedings of EACL 03, Budapest, Hungary*: East Stroudsburg, PA: 291-298.
- Schneider, Gerold (2003a): "A low-complexity, broad-coverage probabilistic dependency parser for english". In: *Proceedings of NAACL/HLT 2003 Student session*. Edmonton, Canada: 31-36.
- Schneider, Gerold (2003b): "Extracting and Using Trace-Free Functional Dependencies from the Penn Treebank to Reduce Parsing Complexity". In: Nivre, Joakim/Hinrichs, Erhard (eds.): *Proceedings of Treebanks and Linguistic Theories (TLT) 2003*. Växjö, Sweden: 153-164.
- Tapanainen, Pasi/Järvinen, Timo (1997): "A non-projective dependency parser". In: *Proceedings of the 5th Conference on Applied Natural Language Processing*. Association for Computational Linguistics: 64-71.
see <http://www.ling.helsinki.fi/~tapanain/tekeleet.html>
- Tesnière, Lucien (1959): *Eléments de Syntaxe Structurale*. Paris.
- Wehrli, Eric (1997): *L'analyse syntaxique des langues naturelles*. Paris.