Extracting and Using Trace-Free Functional Dependencies from the Penn Treebank to Reduce Parsing Complexity

Gerold Schneider

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

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. Its output is a hierarchical structure of syntactic relations, functional dependency structures, [6], [17].

Broad-coverage syntactic parsers with good performance have now become available [3], [4], [7], but they typically produce pure constituency data as output, trees that do not include the grammatical function annotation nor the empty nodes annotation provided in Treebanks such as the Penn Treebank [11]. 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, 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 ....*

[8] presents a pattern-matching algorithm for post-processing the output of such parsers to add empty nodes to their parse trees. While encouraging results

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

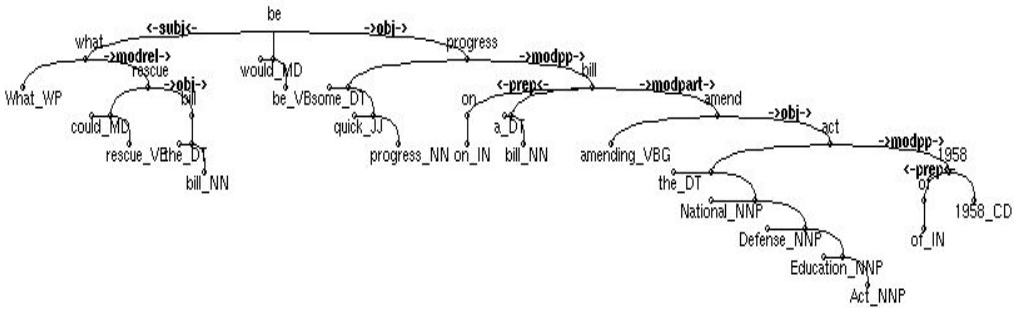

Figure 1: Dependendency Tree output of of the SWI Prolog graphical implementation of the parser

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 ..." [8]. The parser introduced here offers a response to this suggestion by combining a statistical approach with a rule-based approach in Dependency Grammar (DG).

2 Short Description of the Parser

The parser differs on the one hand from successful DG implementations (e.g. [10], [17]) by using a statistical base, and on the other hand from state-of-the-art statistical approaches (e.g. [4]) 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 [5], but for a large subset of dependency relations instead of for PP-attachment only.

2.1 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 and their heads by means of chunking [12]. 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 [13]. Parsing takes place only between the heads of chunks, and only using the best tag suggested by the tagger.

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 are expressed in hand-written declarative rules. The rules of this parser 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.

The dependency rules consist of the Penn tag of the head and the dependent, the possible dependency direction, the tag of the projection, and a list of additional constraints. In DG, the tag of the projection is usually the same as the head tag. Exceptions to the isomorphism between the projection and the head are PPs, which differ from NPs as they cannot be subjects, and the possible functional changes of a word described by [18] as translations. For example, participles may function as adjectives upwards in the tree (*Western industrialized/VBN countries*), or present participle gerunds may function as nouns (*after winning/VBG the race*). As typical in Functional DG, only content words are allowed to be heads. For example, a complementizer is an optional dependent of the subordinated verb. This has the advantage that there are no empty heads. In a nounless noun chunk such as *the poor*, the adjective acts as a noun, another example of a Tesnière translation.

Additional well-known linguistic constraints, such that adjuncts only follow after all complements, that verbs can maximally have two noun objects, or that usually only relational nouns can be modified by several PPs are encoded.

2.3 Attachment Probability Model

It is very difficult 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 vital for any parser, for two reasons: first, the syntactically possible analyses can be ranked according to their probabilities. Second, in the course of the parsing pro-

cess, very improbable analyses can be abandoned, which greatly improves parsing efficiency. Parsing decisions are assigned MLE probabilities of their distance and their lexical participants.

2.3.1 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.

$$p(Distance|REL) = \frac{\#(REL \bigwedge Distance)}{\#REL} \tag{1}$$

2.3.2 Head Lexicalisation

Syntactic preferences depend on the lexical items involved. For a dependeny relation R going to the right involving head word a and dependent word b, the following MLE estimation is used as a base:

$$p(R|right, a, b) = \frac{\#(R, right, a, b)}{\#(right, a, b)} \tag{2}$$

A number of extensions and back-offs are used in the system [16].

2.3.3 Robustness and Speed

Beside being broad-coverage and robust, the parser has explicitly been designed to keep complexity as low as possible during the parsing process in order to be fast enough to be useful for parsing large amounts of unrestricted text [16]. It uses the probabilities for a beam search and for collecting a promising path through partial structures if no complete parse for a sentence has been found. One of the design decisions to keep complexity low is its use of a version of Functional DG that remains completely context-free yet expresses non-local dependencies. A parsing algorithm able to treat completely unrestricted LDDs is NP-complete [14]. Parsing algorithms currently used for formal grammars such as LFG or HPSG often have complexity $O(n^5)$, while the CYK-based CFG used here has $O(n^3)$. In practice complexity is even lower because of the beam. Parsing the held-off 46,527 words of section 0 of the Penn Treebank takes about 10 minutes on a 2.5 GHz Pentium 4 PC, which amounts to 100 words per second or 300'000 words per hour.

Relation	Label	Example
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 1: The most important dependency types used by the parser

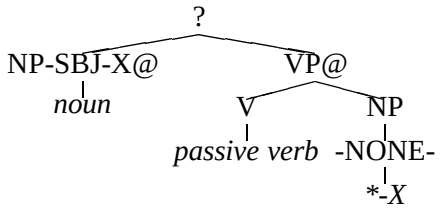

Figure 2: Extraction pattern for passive subjects

3 Extracting Dependency Relations from the Penn Treebank

The lexicalised frequency counts of the dependency relations, such as subject, object, etc., (see table 1 for a list of the main types), are extracted from sections 2-24 of the Penn Treebank by means of *tgrep*, a popular query tool for the extraction of tree structures from Treebanks.

The $subject$ 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

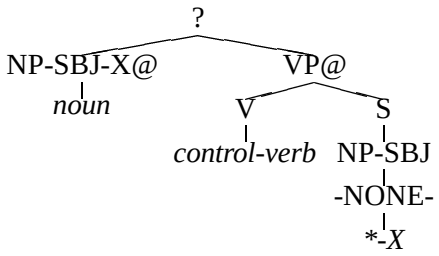

Figure 3: Extraction pattern for subject control

passive structures, we have decided to keep the relation label $subject$, 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 3.

Grammatical role labels, empty node labels and tree configurations spanning several local subtrees are thus 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 [8], (3) the costly overhead for dealing with unbounded dependencies can be largely avoided.

Some relations include local alongside non-local dependencies. For example, the $object$ relation includes copular verb complements and small clause complements.

4 A Frequency Analysis of Types of Empty Nodes

While these dependency relations covering empty nodes and several levels of subtrees obviously express some non-local dependencies and reduce parse tree depth and complexity, the question about their quantitative coverage must arise.

4.1 Overview

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 [8] [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

	Antecedent	POS	Label	Count	Description
1	NP	NP	*	22,734	NP trace
2		NP	*	12,172	NP PRO
3	WHNP	NP	*T*	10,659	WH trace
4			*U*	9,202	Empty units
5			0	7,057	Empty complementizers
6	S	S	*T*	5,035	Moved clauses
7	WHADVP	ADVP	*T*	3,181	WH-trace
8		SBAR		2,513	Empty clauses
9		WHNP	0	2,139	Empty relative pronouns
10		WHADVP	0	726	Empty relative pronouns

Table 2: The distribution of the 10 most frequent types of empty nodes and their antecedents in the Penn Treebank (adapted from [8])

Type	Count	prob-modeled
passive subject	6,803	YES
indexed gerund	4,430	NO
inverted NP-V	2,427	YES
control, raise, semi-aux	6,020	YES
others / not covered	3,054	
TOTAL	22,734	

Table 3: Coverage of the patterns for the most frequent NP traces

complementizer is an optional dependent (or in HPSG a marker) of the subordinated verb (see section 2.2).

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.

4.2 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, as shown in figure 4.

Unlike in control, raising and semi-auxiliary constructions, the antecedent of an indexed gerund cannot be established easily. The fact that almost half of the

```
sentobj(ask, elaborate, _g101293, '->', 36).
modpart(salinger, ask,elaborate, '<-', 36).
appos(salinger, secretary, _g101568, '->', 36).
subj(reply, salinger, ask, '<-', 36).
subj(say, i, _g101843, '<-', 36).
subj(get, it, _g102032, '<-', 36).
subj(go, it, subj_control, '<-' , 36). %  subj-control
prep(draft, thru, _g102286, '<-', 36).
pobj(go, draft, thru, '->', 36).
sentobj(get, go, draft, '->', 36).
sentobj(say, get, it, '->', 36).
sentobj(reply, say, i, '->', 36).
```

Figure 4: Predicate-argument output for sentence 36: "Asked to elaborate, Pierre Salinger, White House press secretary, replied, I would say it's got to go thru several more drafts."

gerunds are non-indexed in the Treebank indicates that information about the unexpressed participant is rather semantic than syntactic in nature, much like in pronoun resolution. Currently, 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.

4.3 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.[1] A further 3,095 are recognized as non-indexed gerunds. Infinitives and gerunds may act as subjects, which are covered by [18] 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.

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

	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 [1]; and to Charniak [3], according to Preiss			
	Subject($ncsubj$)	Object($dobj$)		
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

4.4 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. Search for appropriate verbs starts at the end of the sentence and is not yet enriched with any probability model.

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. [9] suggests evaluating on the linguistically more meaningful level of syntactic relations. For the current evaluation, a hand-compiled gold standard following this suggestion is used [2]. It contains the grammatical relations of 500 random sentences from the Susanne corpus. The mapping between Carroll's grammatical relations and our dependency output is according to figure 5. R_C is a syntactic relation defined by Carroll, non-subscript relations are functional relations of the parser. Comparing these results to [10] and [15] as far as is possible shows that the performance of the parser is state-of-the-art (see table 4)[2].

[2]One reason why recall is low for Buchholz and for Charniak is precisely because relevant LDDs, such as subject-control, are not expressed by these parsers but annotated in Carroll

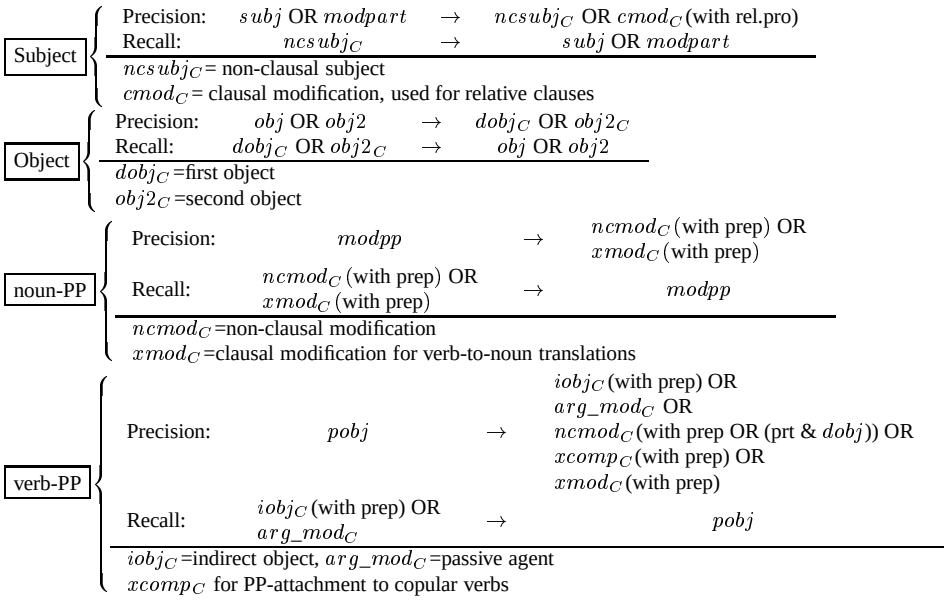

Figure 5: Mapping of dependency relations for evaluation to Carroll's annotation

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. Absolute values are given due to the low counts of these often rare relations, which demands larger dependency-annotated corpora. Since the patterns that are used for the extraction of dependencies tend to have recalls slightly below 100 %, their usability for a reliable evaluation is problematic.

6 Conclusions

We have presented a fast, lexicalized broad-coverage parser that delivers 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.

	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

It has been argued that, for English, the majority of non-local dependencies except for few WH-traces can be treated as local dependencies by (1) using and modeling dedicated patterns across several levels of constituency subtrees partly leading to dedicated but fully local dependency syntactic relations, by (2) lexicalized post-processing rules, and that (3) some non-local dependencies are simply artifacts of the grammatical representation. Due to the low counts of the rarer LDD syntactic relations, larger dependency-annotated corpora are needed.

References

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

[2] 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.

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

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

[5] 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.

[6] 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.

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

[8] 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.

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

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

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

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

[13] 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.

[14] 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.

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

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

[17] 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.

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