

Disambiguation for a Linguistically Precise German Parser

Von der Philosophisch-Historischen Fakultät der Universität Stuttgart
zur Erlangung der Würde eines Doktors der
Philosophie (Dr. phil.) genehmigte Abhandlung

Vorgelegt von
Martin Forst
aus Tübingen

Hauptberichter: Prof. Dr. Christian Rohrer
Mitberichter: Prof. Hinrich Schütze, PhD

Tag der mündlichen Prüfung: 5. März 2007

Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart

2007

Contents

Acknowledgements	vii
I Introduction	1
1 Motivation and structure	3
1.1 Motivation for this dissertation	3
1.2 Structure of this dissertation	4
2 The German <i>ParGram</i> LFG	7
2.1 Generalities	7
2.2 Setup of the system	10
2.3 Evaluation of the symbolic LFG	12
2.4 Summary	14
3 Disambiguation for Unification-based Grammars	15
3.1 A hand-crafted disambiguation mechanism	15
3.2 Log-linear models for disambiguation	16
3.2.1 Log-linear models maximizing the joint probability of strings and analyses	16
3.2.2 Log-linear models maximizing the conditional probability of analyses given strings	17
3.2.3 Exploiting auxiliary distributions	18
3.2.4 Regularization and property selection	19
3.3 Summary	20
II Data Acquisition — Treebank Conversion	21
4 Treebank Conversion for the Construction of Training Data	23
4.1 The need for data in grammar development	23
4.2 Similar work in treebank conversion	25
4.3 The TIGER Treebank and the relational TIGER representation	25

Contents

4.4	Lemmatization	28
4.5	Treebank conversion by (MT) transfer rules	30
4.5.1	The transfer system	30
4.5.2	Transfer phenomena	32
4.6	Evaluation of the treebank conversion procedure	36
4.7	Matching grammar output against TIGER-derived f-structure charts	37
4.8	Summary	38
5	The TiGer Dependency Bank — A Dependency-Based Gold Standard for German Parsers	39
5.1	A gold standard for (hand-crafted) German parsers	39
5.2	Constructing the TiGer DB	41
5.2.1	Automatic derivation of f-structure charts from the TIGER Treebank	41
5.2.2	Automatic disambiguation of TIGER-derived f-structure charts	42
5.2.3	Manual disambiguation of TIGER-derived f-structure charts	43
5.2.4	Conversion into dependency triples and validation	43
5.3	Grammatical relations and features encoded in the TiGer DB	44
5.3.1	Indices, Reentrancies and Lemmatization	44
5.3.2	Grammatical relations	46
5.3.3	Atomic features	51
5.4	Uses of the TiGer DB	53
5.5	Summary	54
III	OT-inspired Disambiguation	55
6	Manually Defined OT Constraint Rankings for Disambiguation	57
6.1	Optimality-Theoretically inspired disambiguation in XLE	57
6.2	Optimality marks in the original German grammar	59
6.2.1	NOGOOD OT marks	60
6.2.2	STOPPOINT OT marks	61
6.2.3	General OT marks	66
6.3	Summary	107
7	Corpus-Based Learning of OT Constraint Rankings	109
7.1	Disambiguation: a two-stage approach	109
7.2	Methodology	111
7.2.1	Trimming the linguistic pre-filter	111
7.2.2	Measuring the quality of the pre-filter	111
7.2.3	Corpus-based learning of a ranking	112
7.2.4	Augmenting the set of OT marks	113

7.2.5	Relaxing the filter	114
7.3	Experiments	115
7.3.1	Data	115
7.3.2	Training and first results	116
7.3.3	Relaxing the filter	118
7.4	Summary	120
IV A Log-linear Model for Disambiguation		123
8	A first model	125
8.1	Properties based on XLE property templates	125
8.1.1	C-structure-based properties	126
8.1.2	F-structure-based properties	133
8.2	Estimation of property weights	139
8.3	Evaluation	140
8.4	Summary	142
9	Property design for the disambiguation of German LFG parses	143
9.1	Additional properties based on grammar-internal information	143
9.1.1	Properties for resolving ambiguities concerning the dependency mod	144
9.1.2	Properties for resolving ambiguities due to case-ambiguous DPs on the basis of the nature of these DPs	147
9.1.3	Properties for resolving ambiguities due to case-ambiguous DPs on the basis of their relative linear order	152
9.1.4	Properties for the resolution of PP and ADVP attachment ambiguities	158
9.1.5	Properties for the resolution of GEND and NUM ambiguities in XCOMP-PREDS	162
9.1.6	Properties for the resolution of attachment ambiguities concerning extraposed ADJ-RELS, APP-CLAUSES, COMPS and VCOMPS	164
9.1.7	Properties for the resolution of part-of-speech ambiguities	169
9.1.8	Properties capturing dependencies	171
9.2	Additional properties based on external resources	180
9.2.1	Properties based on <i>GermaNet</i> information on humanness, ‘groupness’ and animacy	182
9.2.2	Auxiliary distributions from <i>Gramotron</i> data	186
9.3	Evaluation	193
9.4	Comparison to similar systems	193
9.5	Summary	196

Contents

10 Property selection and regularization	197
10.1 The problem of overfitting	197
10.2 Property selection	198
10.2.1 Frequency-based cutoffs	198
10.2.2 Incremental property selection	204
10.2.3 Correlational analysis	204
10.3 Regularization	207
10.4 Combined regularization and property selection	209
10.5 Summary	213
V Conclusions	215
11 Summary and Conclusions	217
11.1 Summary	217
11.2 Conclusions	218
11.3 Outlook to future work	220
Summary in German – Zusammenfassung	223
Bibliography	239

Acknowledgements

This dissertation has emerged from my work in two DFG-funded projects at IMS: the *TiGer* project on the creation of a large syntactically annotated corpus of German newspaper text and the *DLFG* project on the development of a disambiguation module for a broad-coverage German LFG. My work in both projects has been embedded in the larger context of the *ParGram* initiative, a collaborative effort for the development of parallel grammars. I would like to thank the German Research Council (DFG) for their financial support and all the members of the *ParGram* community for offering such a pleasant and lively platform for research in LFG grammar development.

I would like to thank all the members of IMS for providing such an enriching working environment. In particular, I would like to thank Christian Rohrer for his excellent support and advice over several years, leading up to his supervision of this dissertation. He also made possible several short-term stays at the Palo Alto Research Center (PARC) which were invaluable for the development of the disambiguation module presented in this dissertation. Furthermore, I would like to thank Jonas Kuhn, who provided me with invaluable input for the Optimality Theoretically-inspired part of this dissertation towards the beginning of my time in Stuttgart, and Aoife Cahill, who, during the last months, has made important contributions by discussing several machine learning topics and by proof-reading this dissertation. Finally, I would like to thank Hinrich Schütze, Sabine Schulte im Walde and Helmut Schmid for discussions on topics related to this dissertation and their feedback.

For inspiring comments and discussions that we had during my stays at PARC and on other occasions, I owe thanks to Ron Kaplan, Tracy H. King, Stefan Riezler, John Maxwell and Richard Crouch as well as to Anette Frank from Saarland University and DFKI. Stefan has always been responsive when I have felt the need for discussing issues of stochastic disambiguation in detail, Anette helped me to familiarize myself with the transfer system of XLE and John has been extremely cooperative every time I have asked for additional functionalities in XLE. Furthermore, I would like to thank the audiences at the LINC Workshop 2004 in Geneva, at the LFG Conference 2005 in Bergen, at several *ParGram* meetings and at the Workshop on Large-Scale Grammar Development and Grammar Engineering organized by Shuly Wintner at the University of Haifa in June 2006 for their important feedback. Finally, this dissertation has greatly benefitted from comments that Tracy and Victoria Rosén from the University of Bergen made on draft versions of it.

I would like to conclude my acknowledgements by thanking my parents for their support and encouragement and my wife Mónica for coming with me to Stuttgart and accompanying me in the ups and downs of this dissertation. I therefore dedicate it to her.

Acknowledgements

The publication of this dissertation is supported by the German Research Council (DFG) in the context of the Collaborative Research Center on *Incremental Specification in Context* at the University of Stuttgart (SFB 732).

Part I
Introduction

Chapter 1

Motivation and structure

1.1 Motivation for this dissertation

In computational linguistics an important point of interest is grammar development, i.e. the implementation of grammars. The goal of grammar development is to produce detailed linguistic analyses for arbitrary sentences or even texts in a given language. Detailed analyses are important for applications like machine translation or question answering, which involve at least some meaning construction, and this level of granularity is generally only achieved by hand-crafted grammars, such as the *Delphin* HPSGs (Oepen et al. 2002), the HPSG-inspired *Alpino* parser for Dutch (van Noord 2006) or the *ParGram* LFGs (Butt et al. 2002). In recent years, there have also been attempts to induce deep grammars from treebank resources, such as the English CCG parser (Hockenmaier 2003, Clark & Curran 2004), the English Enju HPSG (Miyao et al. 2004) or the LFG approximations developed at Dublin City University (O'Donovan et al. 2005a). These obtain very good results for English, and the basic approaches have been successfully applied to other languages, such as German (Cahill et al. 2005), Japanese (Yoshida 2005) and Spanish (O'Donovan et al. 2005b); so far, the parsing accuracy achieved is less good for these less configurational languages, however.

Hand-crafted grammars are symbolic in nature. This means that they do not contain information about the probabilities with which the rules in them are to be applied. Due to the manifold and very frequent syntactic ambiguities that occur in most sentences of real texts, a hand-crafted grammar, however detailed and precise it is, always outputs several analyses for an average sentence. For many sentences, even hundreds, thousands or millions of analyses are constructed.

Most applications for which linguistically precise grammars are developed cannot build on top of such an ambiguous output. In other words, the often massive ambiguity in the output of these grammars is a major obstacle for their

usefulness. The creation of disambiguation modules that perform a systematic selection among the parses produced by the symbolic grammar helps to overcome this obstacle. Furthermore, from a more theoretical perspective, it is interesting and important to investigate ways of integrating so-called ‘soft’ constraints known to be at work in languages into language-processing systems. Purely symbolic approaches can only capture ‘hard’, i.e. categorical, constraints.

This dissertation deals with methods for the disambiguation of linguistically precise analyses of German sentences, namely the parses produced by the German *ParGram* LFG. It presents the data on the basis of which the disambiguation module of the German *ParGram* LFG is developed and evaluated, as well as the architecture of this disambiguation module, which comprises two submodules: an Optimality-Theoretically inspired pre-filter and a log-linear model for the final selection of the most probable parse(s). It thus describes how the originally purely symbolic German *ParGram* LFG was converted into a stochastic unification-based grammar that produces systematically disambiguated precise syntactic analyses for free German text.

Rather than refining the machine learning methods employed for the development of stochastic disambiguation modules, this dissertation takes the perspective of the grammar writer in that it focuses on the linguistic information provided to the different disambiguation components in the form of so-called optimality marks and learning properties. Nevertheless, it also addresses the issue of how to make the stochastic models used less prone to overfitting the training data.

1.2 Structure of this dissertation

Reflecting the different steps involved in the development of the disambiguation module of the German *ParGram* LFG, this dissertation is structured as follows:

- Part I presents the foundations of the work described. In this context, Chapter 2 focusses on the German *ParGram* LFG as it is documented in Dipper (2003), but it also addresses further developments in the grammar. Chapter 3 gives an overview over developments and the state of the art in the domain of disambiguation techniques for unification-based grammars.
- Part II addresses the construction of data for training and evaluation on the basis of an existing treebank resource, namely the TIGER Treebank. Chapter 4 describes how the annotation graphs of the TIGER Treebank were converted into potentially ambiguous packed representations of LFG f-structures and how these were subsequently matched against the representations produced by the German *ParGram* LFG for the construction of

1.2 Structure of this dissertation

the training corpus. Chapter 5 documents how the TIGER-derived representations were disambiguated and hand-checked for the creation of a gold standard for German parsers, the TiGer Dependency Bank, comparable in granularity with the f-structures produced by the German *ParGram* LFG.

- Part III deals with disambiguation on the basis of so-called optimality marks (OT marks), which are employed as a pre-filter in the final system, mainly for the sake of efficiency. Chapter 6 documents the state of the OT-mark-driven disambiguation in the grammar that we started out with. Chapter 7 reports on experiments in which the relative ranking of the OT marks was learned on the basis of corpus data and which also helped us to find out which OT marks are suited for a pre-filter, which is supposed to preserve the intended reading of a sentence very reliably.
- Part IV addresses various aspects that have to be taken into account in the development of a log-linear model for the disambiguation of syntactic analyses. In this context, Chapter 8 documents the development of a first log-linear model that we trained along the lines of Riezler et al. (2002) and Riezler & Vasserman (2004). Chapter 9 is dedicated to the design of new properties for the disambiguation of German LFG parses; it demonstrates that property design is of crucial importance in the development of well-performing log-linear models. Chapter 10, finally, addresses the question of what different methods of property selection and various regularization methods can contribute to the development of models that generalize well to unseen data.
- Part V provides a summary of the dissertation and the principal conclusions from the results of the experiments presented, as well as an outlook to future work on and with the German *ParGram* LFG.

Chapter 2

The German *ParGram* LFG

In this chapter, we introduce the German *ParGram* LFG, which is the linguistically precise German parser the disambiguation module presented in this dissertation has been developed for. We give an overview of the context in which it has evolved and the main development stages that it has gone through as well as the setup of the final system. At the end of the chapter, we also provide an evaluation of the symbolic part of the German *ParGram* LFG.

2.1 Generalities

The German *ParGram* LFG documented in Dipper (2003) was the starting point of the work described in this dissertation. It is a computational grammar formalized in the framework of Lexical Functional Grammar (LFG) (Kaplan & Bresnan 1982) and implemented in the grammar development and processing platform XLE (Crouch et al. 2006). It has been developed as part of the *ParGram* family of LFG grammars. *ParGram* stands for “parallel grammars”. The *ParGram* initiative comprises the Palo Alto Research Center (PARC) in California (English, Chinese, French), the Fuji Xerox Research Center in Japan (Japanese), the University of Stuttgart in Germany (German), the University of Bergen in Norway (Norwegian), the University of Constance in Germany (Urdu), the Universities of Oxford (Malagasy), Essex (Welsh) and Manchester (Arabic) in Britain, Sabancı University in Turkey (Turkish), the University of Debrecen in Hungary (Hungarian) and the Ho Chi Minh City Institute of Information Technology (Vietnamese). It aims at developing grammars for typologically and genetically diverse languages in parallel, “parallel” meaning here that the abstract syntactic analyses produced, the so-called f-structures, are parallel across languages whenever this is not contrary to the linguistic adequacy of the analyses. Most importantly this means that all the *ParGram* grammars employ a common feature space, i.e. parallel representations, but we have also started to share common templates, i.e. parallel means of constructing representations

(King et al. 2005).

LFG is a grammar formalism which makes use of two representation levels to encode syntactic properties of sentences: constituent structure (c-structure) and functional structure (f-structure). C-structures are context-free trees that encode the constituents and the linear order of the corresponding sentences. F-structures are attribute-value matrices that encode grammatical relations and morphosyntactic features. There is thus a clear distinction between the category of a constituent and its function in LFG.¹ In a crosslinguistic perspective, it is assumed that translational equivalents of sentences vary considerably at the level of c-structure, but that at the level of f-structure, languages behave much more alike. Let us consider the English and German sentences in (2.1) and (2.2) and the c- and f-structures that are associated with them by the English and German *ParGram* LFGs.

- (2.1) The letter will have arrived tomorrow.
- (2.2) Der Brief wird morgen angekommen sein.
The letter will tomorrow arrived be.

While the c-structures associated with the translational equivalents in the two languages differ considerably,² the corresponding f-structures are largely parallel. Apart from the PRED values, the only differences found between them are minor differences in the encoding of tense and aspect (TNS-ASP).

In the initial phase of its development, the German *ParGram* LFG was extended and revised in a *phenomena-driven* manner. This means that, first, basic DPs and PPs were implemented; then, the basic clausal syntax was covered; subsequently, subordinate clauses were included, etc. A selection of aspects treated during this phase are documented in Butt et al. (1999). Further constructions and much more detail are provided in Dipper (2003). As a consequence of the linguistic focus in grammar writing, the main criteria were linguistic adequacy and generality and, to a lesser extent, coverage. Efficiency and ambiguity were much less taken into account.

As a result, the grammar had good coverage for most frequent (and many less frequent) constructions of German syntax, but when parsing newspaper corpora, which contain a lot of elliptical sentences and “messy” material, such as abbreviations, currencies, complex named entities etc., and where the average sentence length is around 16 in German, the coverage of the grammar

¹For information on how c-structures and f-structures are related in LFG, the reader is referred to Chapter 2 in Dipper (2003). For thorough introductions to LFG, the reader is referred to Bresnan (2001), Dalrymple (2001), Falk (2001), Dalrymple et al. (1995).

²Many of the differences, in particular differences in the naming of categories, are, of course, due to the fact that no effort has been made to make c-structures parallel. Many others, however, in particular differences resulting from divergences in word order, could not be avoided, even if an effort to keep c-structures parallel were made.

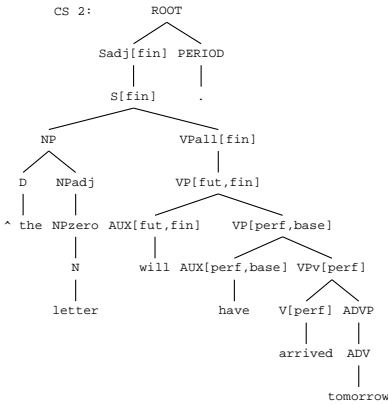


Figure 2.1: C-structure associated with (2.1)

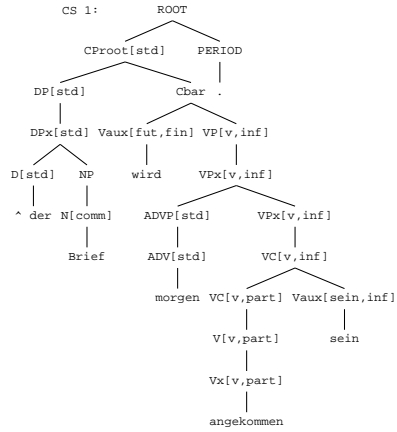


Figure 2.3: C-structure associated with (2.2)

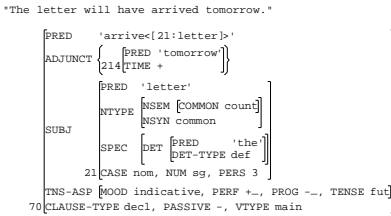


Figure 2.2: F-structure associated with (2.1)

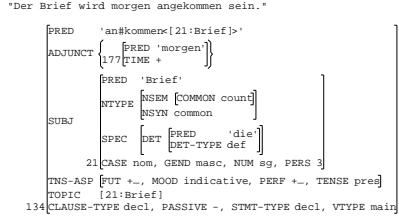


Figure 2.4: F-structure associated with (2.2)

was only between 35% (Dipper 2003) and 55% (Rohrer 2003, p.c.) in terms of analyses spanning the entire input sentences. A grammar with a coverage this low is not useful for most applications, even if it is coupled with partial parsing techniques like the fragment parsing technique presented in Riezler et al. (2002). In order to achieve competitive results on free text, we thus needed to improve coverage considerably.

Scaling the grammar to free text involved mainly two types of modifications. On the one hand, the grammar had to be extended in order to cover a higher proportion of elliptical constructions and “messy” material. On the other hand, rules that could arguably be considered excessively general on the basis of our corpus data or that, while being expensive to process, cover only very rare

phenomena had to be restricted in order to reduce the efficiency problems faced by the initial grammar.³ Rohrer & Forst (2006a,b) document what concrete modifications were made to the grammar in the course of this phase of *corpus-driven* grammar development.

Besides low coverage, the main obstacle to the usefulness of the grammar for applications used to be the fact that, for an average sentence from a newspaper corpus, several analyses were produced. For many sentences, there were even thousands or sometimes millions of analyses. The original German *ParGram* LFG, just like any other hand-crafted deep grammar, thus needed to be complemented with a disambiguation module. The disambiguation techniques used now in the German *ParGram* LFG are the topic of this dissertation. All of these techniques are *corpus-driven* and make use of *statistical* methods. This dissertation thus reports on how a (symbolic) unification-based grammar (UBG) was made a stochastic or probabilistic unification-based grammar (SUBG or PUBG).

2.2 Setup of the system

The final grammar can be seen as consisting of three main parts: (i) a cascade of finite-state transducers used for pre-processing, (ii) the symbolic grammar proper and (iii) the disambiguation module. The main focus of this dissertation lies on the disambiguation module, as illustrated in Figure 2.5.

The cascade of finite-state transducers performs mainly tokenization and morphological analysis. For words unknown to the morphology, a guesser is included as well and there is a (still limited) transducer for the treatment of multiwords. Below are the outputs the tokenizer produces for sentence (2.2) and the outputs the morphology produces for the word *angekommen*. As we can see, all transducers are non-deterministic, so that at each pass through a transducer the number of outputs can increase.

tokenizer input:

Der Brief wird morgen angekommen sein.

tokenizer outputs:⁴

Der TB Brief TB wird TB morgen TB angekommen TB sein TB . TB

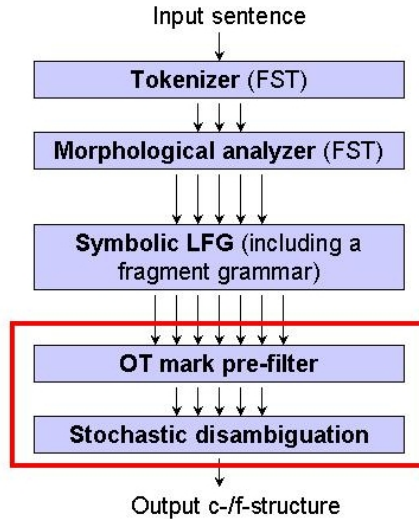
Der TB Brief TB wird TB morgen TB angekommen TB sein. TB . TB

ˆ der TB Brief TB wird TB morgen TB angekommen TB sein TB . TB

ˆ der TB Brief TB wird TB morgen TB angekommen TB sein. TB . TB

³For a considerable proportion of corpus sentences, the initial grammar actually did not fail because it lacked rules that were necessary in order to analyze the sentences, but due to limitations in time and/or computational resources.

⁴TB stands for 'token boundary'. The ˆ indicates that decapitalization has been performed.

Figure 2.5: Basic setup of the German *ParGram* LFG

morphology input:

angekommen

morphology outputs:

an#kommen ^VPAST +ADJ .Pos .PA

an#kommen +V .PPast

All the outputs of the finite-state transducer cascade, represented as a graph, are then analyzed by the LFG proper. This is basically done in two steps, namely the construction of all possible context-free c-structures and the resolution of the constraints in the f-annotations of these c-structures. If it is impossible to construct a valid c-/f-structure pair for an entire sentence, XLE collects c-/f-structure pairs for fragments of this sentence, so that 100% coverage and, hence, perfect robustness is achieved. The output produced by the grammar consists of packed representations of c-/f-structure pairs. It is after this step that the number of analyses attains its maximum and is often in the dozens, thousands or even millions.

Many, if not most, applications cannot use output structures that exhibit such a high degree of ambiguity. Therefore, the symbolic grammar is com-

plemented with a disambiguation module. In our setup, this disambiguation module is organized in two stages: (i) the Optimality-Theory-inspired disambiguation, which acts as a pre-filter in the final system (see Part III), and (ii) the stochastic disambiguation component based on a log-linear model (see Part IV).

2.3 Evaluation of the symbolic LFG

In order to give the reader an idea of what the contribution of the disambiguation module can be in quantitative terms, we provide evaluation figures for the analyses of the symbolic part of the German *ParGram* LFG filtered by the Optimality-Theoretically inspired pre-filter in Table 2.1. These figures are computed on the 1,497 TiGer Dependency Bank (TiGer DB) structures of our test set with the help of the triple evaluation software of Crouch et al. (2002). (What exactly is encoded in the TiGer DB structures and how they were established is documented in Chapter 5.) We provide the labeled precision and recall for all grammatical relations and morphosyntactic features that occur at least 40 times in the gold standard structures of the test set. The upper bound for these metrics is set by the best possible selection (according to the gold standard annotations) among the analyses contained in the packed representations of c- and f-structure pairs; the lower bound is determined by an arbitrary selection among them (lower bound).⁵ In addition, we provide the ‘general’ upper bound and lower bound precision and recall (all grammatical relations *and* morphosyntactic features) as well as the corresponding ‘PREDS only’ (grammatical relations only) figures. F-score as the measure combining precision and recall cannot be given for reasons of space. However, it is used in later tables, and the upper bound and lower bound F-scores achieved by the grammar are given in Table 8.1 (p. 141).

⁵A detailed discussion of this lower bound can be found in Section 8.3.

2.3 Evaluation of the symbolic LFG

relation/feature	upper bound		lower bound	
	precision	recall	precision	recall
all	86.17	84.83	80.91	79.94
PREDS only	80.68	78.08	73.71	71.51
app (close apposition)	67	59	60	50
app_cl (appositive clause)	100	36	100	30
cc (comparative complement)	47	20	33	15
cj (conjunct of coord.)	82	62	76	57
da (dative object)	68	66	50	61
det (determiner)	91	93	89	91
gl (genitive in spec. pos.)	90	89	86	84
gr (genitive attribute)	87	90	79	79
mo (modifier)	70	70	59	59
mod (non-head in compound)	94	94	87	88
name_mod (non-head in compl. name)	79	85	77	82
number (number as determiner)	77	89	76	85
oa (accusative object)	83	73	69	61
obj (arg. of prep. or conj.)	89	90	85	86
oc_fin (finite cl. obj.)	69	65	66	63
oc_inf (infinite cl. obj.)	88	78	87	77
op (prepositional obj.)	65	50	61	45
op_dir (directional argument)	62	19	47	14
op_loc (local argument)	59	59	49	41
pd (predicative argument)	65	59	60	55
pred_restr	90	94	76	83
quant (quantifying determiner)	70	70	68	67
rc (relative clause)	82	67	66	54
sb (subject)	76	75	69	68
sbp (logical subj. in pass. constr.)	74	63	62	48
case	88	86	80	79
comp_form (complementizer form)	89	63	89	61
coord_form (coordinating conj.)	89	84	88	83
degree	87	91	84	89
det_type (determiner type)	93	98	93	97
fut (future)	92	80	92	80
gend (gender)	92	91	87	88
mood	93	88	93	87
num (number)	90	93	76	85
pass_asp (passive aspect)	91	72	90	71
perf (perfect)	96	79	95	78
pers (person)	90	81	83	76
pron_form (pronoun form)	95	59	95	59
pron_type (pronoun type)	67	77	65	75
tense	95	89	94	89

Table 2.1: Upper and lower bound precision and recall (in %) in our test set

We observe that, in general, recall is lower than precision, which can be explained by the fact that partial analyses systematically contain fewer dependencies than the corresponding gold standard structures. We also observe that the gap between the lower bound and the upper bound is bigger for grammatical relations than for morphosyntactic features and that, for some grammatical relations, e.g. *da* (dative objects), *mo* (modifiers), *oa* (accusative objects), *op_dir* (directional arguments), *op_loc* (local arguments), *pred_restr* (predicates in nominalized adjectives), *rc* (relative clauses) and *sbp* (logical subjects in passive constructions), this gap is considerably bigger than for others. This means that the grammatical relations just mentioned are considerably more often affected by ambiguities in the packed *c-/f*-structure representations produced by the grammar and that, hence, high-quality disambiguation is particularly important for achieving good results on these types of dependencies. Finally, we can see that, for some grammatical relations, namely *det* (determiners), *gr* (genitive attributes), *number* (numbers as determiners) and *pred_restr* (predicates in nominalized adjectives), recall is actually higher than precision, which indicates that the rules in the symbolic grammar concerning these dependencies should possibly be made more restrictive; for grammatical relations such as *app_cl* (appositive clauses), *cc* (comparative complements), *cj* (conjuncts of coordinations), *oc_inf* (infinitival clausal objects), *op* (prepositional objects), *op_dir* (directional arguments), *op_loc* (local arguments), *rc* (relative clauses) and *sbp* (logical subjects in passive constructions), for which precision is not only slightly, but far higher than recall, the symbolic grammar apparently lacks coverage. In both cases, the problems can only be addressed by modifying the symbolic part of the grammar, since the disambiguation module can only make a selection within the bounds defined by this symbolic part.

2.4 Summary

This chapter has given an overview of the broader context and the phenomena-based and corpus-based stages in which the German *ParGram* LFG has been developed. It has presented the setup of this now stochastic unification-based grammar, which consists of a preprocessing step based on finite-state transducers, an LFG grammar augmented with robustness techniques and a two-stage disambiguation module. Finally, it has provided evaluation figures for the symbolic part of the grammar: The upper-bound F-score computed on the grammatical relations and morphosyntactic features encoded in the TiGer DB structures of our test set is 85.50%; the lower-bound F-score is 80.42%.

Chapter 3

Disambiguation for Unification-based Grammars

This chapter provides an overview over developments and the state of the art in the domain of disambiguation techniques for unification-based grammars. It presents a hand-crafted disambiguation mechanism which is used in the *Par-Gram* LFGs as well as the more widely used stochastic approach to disambiguation based on log-linear models.

3.1 A hand-crafted disambiguation mechanism

The only example of a hand-crafted disambiguation mechanism that we are aware of is described in Frank et al. (2001). It is a mechanism inspired by the well-known evaluation mechanism of Optimality Theory (henceforth OT), and evaluates competing analyses according to so-called optimality marks that are projected for these analyses. Optimality marks can be both dispreference marks (comparable to the classical constraints in OT) and preference marks and are hierarchically ordered. The evaluation mechanism compares the counts for optimality marks in the competing analyses starting with the marks that are highest in the hierarchy and, at each step, keeps the analysis or analyses with the least number of marks in the case of dispreference marks or the highest number of marks in the case of preference marks.

This disambiguation mechanism has been employed successfully in the *Par-Gram* grammars for cutting down the number of readings that the parsing system outputs, but it has two important shortcomings:

- Optimality marks are introduced in lexical entries or f-annotated c-structure rules. This means that, as to their expressiveness, they are limited to very local phenomena.¹

¹One could introduce very complex f-annotations in order to project optimality marks only

- Optimality marks are strictly ranked with respect to each other (or not ranked at all), which means that an optimality mark M_1 that is lower in the hierarchy than another optimality mark M_2 can never override the effect of the latter, however often it appears in a given analysis.

As a consequence of these shortcomings, the mechanism has never been used for complete disambiguation, i.e. the selection of the one most probable parse. In fact, it is often the case that several competing analyses (and sometimes even tens or hundreds of thousands) are indistinguishable with respect to their optimality mark profile.

In the final system presented at the end of this dissertation, we make use of this OT-inspired disambiguation mechanism. However, instead of being the only disambiguation device, which it used to be in the German *ParGram* LFG and which is documented in Chapter 6, it only serves as a pre-filter whose filter fidelity (or recall) is ensured with the help of corpus-based methods (Chapter 7).

3.2 Log-linear models for disambiguation

Maximum entropy models have become standard in computational linguistics for a wide variety of tasks (parse reranking, tag sequence reranking, anaphora resolution, etc.). One example of a maximum entropy model is a probabilistic context-free grammar (PCFG), for which the empirical relative rule frequencies are actually maximum likelihood estimates.

Attempts to extend stochastic models developed for context-free (and regular) grammars to unification-based grammars date back to Eisele (1994) and Brew (1995). As Abney (1997) points out, however, these proposals do not yield maximum likelihood estimates, since for context-sensitive grammars, empirical relative rule frequencies are no longer maximum likelihood estimates. For this reason, the estimation of the parameters of a maximum entropy model is considerably more complex for unification-based grammars than it is for context-free (or regular) grammars.

3.2.1 Log-linear models maximizing the joint probability of strings and analyses

In order to create probabilistic versions of unification-based grammars, Abney (1997) suggests using maximum entropy or log-linear models based on property functions. Property functions are basically arbitrary functions that can be

for configurations that range over various levels of the c-structure or the f-structure. However, these additional constraints, which are *not* necessary to insure syntactic wellformedness, would seriously affect the efficiency of a grammar modified in this way.

3.2 Log-linear models for disambiguation

calculated on the basis of parses and that return a real-numbered value. Most often, properties correspond to certain tree or attribute-value matrix (AVM) configurations and just return the number of occurrences of the configuration under consideration. Properties of this kind have natural-numbered values. However, properties do not need to correspond to tree or AVM configurations; they can be (comparative) weight measures, distance measures, auxiliary distributions (see Subsection 3.2.3), etc., and the values that they return can thus range over the entire space of real numbers, including negative numbers.

A log-linear model with m properties is one in which the the probability $P(\omega)$ of an analysis-string pair ω is:

$$P_{\lambda}(\omega) = \frac{e^{\sum_{j=1}^m \lambda_j \cdot f_j(\omega)}}{\sum_{\omega' \in \Omega} e^{\sum_{j=1}^m \lambda_j \cdot f_j(\omega')}}$$

$\omega = (x, y)$ and $\omega' = (x', y')$ are analysis-string pairs, $\lambda = (\lambda_1 \dots \lambda_m)$ is a vector of property weights, $f = (f_1 \dots f_m)$ is a vector of property functions and Ω is the set of all analysis-string pairs that are part of the formal language defined by the grammar.

By determining the probability of each analysis, the model allows us to select the best (i.e. most likely) or the n best analyses for further processing.

3.2.2 Log-linear models maximizing the conditional probability of analyses given strings

Johnson et al. (1999) convincingly argue that parameter estimation, i.e. the estimation of the property weights $\lambda_1 \dots \lambda_m$, for a log-linear model maximizing the joint probability of strings and analyses is not feasible for large-scale unification-based grammars. The reason for this is that, for normalization, it is necessary to sum over all analyses that the grammar under consideration can produce, but that there is an infinite number of such analyses. Therefore, Johnson et al. (1999) propose using log-linear models that maximize the conditional probability of analyses given strings instead of the joint probability of analyses and strings. These models have the following form:

$$P_{\lambda}(x|y) = \frac{e^{\sum_{j=1}^m \lambda_j \cdot f_j(x,y)}}{\sum_{x' \in X(y)} e^{\sum_{j=1}^m \lambda_j \cdot f_j(x',y)}}$$

x is a possible analysis of string y , λ is again a vector with m property weights, f is again a vector of m property functions and $X(y)$ is the set of all possible analyses of string y .

Apart from solving (or at least alleviating) the practical problems of the joint exponential models, these conditional probability models are arguably better

suiting for the task of disambiguation, since the strings are actually given when disambiguating. For these two reasons, they are now standard in disambiguation modules for precision grammars, as is illustrated by the variety of systems that make use of them: Johnson et al. (1999), Riezler et al. (2002), Toutanova et al. (2002), Miyao & Tsujii (2002), Malouf & van Noord (2004), van Noord (2006), Clark & Curran (2004).

3.2.3 Exploiting auxiliary distributions

Unsupervised methods for the estimation of the property weights of log-linear models have (at least so far) been considerably less successful than (semi-) supervised methods (Riezler et al. 2000). But (semi-)supervised methods presuppose that (partially) labeled data, i.e. syntactically annotated corpora in the case of log-linear models for parse disambiguation, are available. This, however, is only the case for a relatively small (albeit growing) number of languages, and even for these, the amount of syntactically annotated data is limited to several thousand or several tens of thousands of sentences.² This limited availability of training data is problematic for the estimation of weights of highly specialized and hence rare properties. Many properties that capture bilocal dependencies, for example, do not occur at all in corpora of the size mentioned above, or they do not occur often enough for reliable parameter estimation.

One way to address this problem is to exploit so-called auxiliary distributions. These are typically statistical models of lexical selectional preferences that can be estimated from very large corpora that have been processed with shallow methods (chunkers, context-free grammars, etc.). Although they are called *auxiliary distributions*, they actually do not need to be probability distributions; they can basically be any kind of (more or less) meaningful statistical information on the basis of which a score can be computed for any given parse.

The first publication that presented experiments with auxiliary distributions as properties of a log-linear model used as part of a stochastic unification-based grammar was Johnson & Riezler (2000). Although the results with the particular auxiliary distributions they used were not very conclusive, they showed very nicely how straightforward it is to integrate into a log-linear model as a property basically any numerically expressible piece of information. In spite of this, auxiliary distributions are, to our best knowledge, not used in the log-linear models used for the disambiguation of precision grammars. Where they are used very successfully, however, is in log-linear models for realization ranking, such as the ones described in Velldal & Oepen (2005) and Nakanishi et al.

²For languages like English, Chinese, Czech, Dutch and German, there are treebanks with a substantial amount of syntactic detail that comprise several tens of thousands of sentences. For other languages, such as French and Spanish, this number drops to several thousands.

(2005), where language model scores for the competing strings that are generated are integrated into the log-linear models as properties. This leads us to assume that auxiliary distributions have quite some potential, but more work is needed to find out what kinds of auxiliary distributions are useful for parse disambiguation and how they are optimally formulated. First experiments in this direction are reported in Section 9.2.2.

3.2.4 Regularization and property selection

Another problem that regularly arises with log-linear models that make use of a very large number of potentially sparse properties is that property weights are overly adapted to the training data. (Several tens of thousands are standard; often, several hundreds of thousands are used, and there are systems, e.g. Charniak & Johnson (2005),³ that employ more than a million properties.) Regularization and property selection are both (potentially interwoven) strategies to avoid overfitting. Regularization consists in assuming a certain distribution for property weights and penalizes extreme property weights that are very unlikely according to the assumed distribution. Property selection aims at ignoring part of the set of properties, namely the properties that do not contribute much to disambiguation and whose effect may thus be a pure coincidence in the training data.

The standard regularization in training log-linear models for parse disambiguation involves applying a Gaussian prior to the property weights, as it was already proposed in Johnson et al. (1999). This strategy is applied in the models presented in Johnson et al. (1999), Riezler et al. (2002), Toutanova et al. (2002), Malouf & van Noord (2004), van Noord (2006).

The standard property selection strategy consists in imposing a frequency-based cutoff c on properties, meaning that a property has to be discriminative in at least c sentences in order to be considered for training. This strategy is applied in the model presented in Malouf & van Noord (2004), van Noord (2006) and a variant of it is discussed in Riezler & Vasserman (2004). It is also reported for the log-linear model used as a reranker in the Brown reranking parser (Charniak & Johnson 2005).

A frequency-based cutoff does address the problem of data sparsity that arises for highly specialized properties and it does reduce the number of properties used in the resulting log-linear model. However, it does not address the redundancy among less sparse properties. A property selection method that aims at identifying the relevant properties among all properties is presented in

³Charniak & Johnson (2005) actually do not describe a log-linear model for the disambiguation of a unification-based grammar, but a reranker for the n -best analyses of the Charniak parser. Nevertheless, this reranker is a log-linear model that works in the very same way as the models used for the disambiguation of UBG analyses.

Riezler & Vasserman (2004). It relies on l_1 regularization, i.e. regularization via a double-exponential or Laplacian prior, and iteratively adds properties to the set of properties considered on the basis of their gradient. Among the property selection mechanisms that we are aware of, it allows for the leanest sets of properties that, at the same time, generalize best to unseen data.

Apart from improving the generalizability of a model, property selection is also important for efficiency reasons, as pointed out in Kaplan et al. (2004). The extraction of the (at least in theory) arbitrarily complex properties from all competing parses is time-consuming, even if the extraction can be performed on a packed representation of all parses. It is reasonable to assume that the time spent on property extraction is proportional to the number of properties retained in the model. Hence, if a property selection method allows us to discard, say, 80% of the properties used initially, the time spent on property extraction should diminish by roughly 80% as well. As to the time spent on the computation of the probability of competing parses, we can safely say that it is also affected positively by a reduction of the number of properties (and, hence, property values to be taken into account), but we prefer abstaining from making claims as to the exact importance of this effect.

3.3 Summary

This chapter has given an overview over the disambiguation techniques used for the disambiguation of unification-based grammars. It has briefly presented the manually defined OT-inspired disambiguation proposed by Frank et al. (2001), before turning to the more widely used stochastic approach based on log-linear models. In this latter context, we have mainly presented the type of log-linear model that is now commonly used for disambiguation, but we have also addressed the topics of how to circumvent the problem of data sparseness resulting from the limited availability of annotated training data by using auxiliary distributions and of how to prevent log-linear models from overfitting the training data.

Part II

Data Acquisition — Treebank Conversion

Chapter 4

Treebank Conversion for the Construction of Training Data

This chapter addresses the construction of training data on the basis of an existing treebank resource, namely the TIGER Treebank. It justifies the approach that we chose; then, it describes how the annotation graphs of the TIGER Treebank were converted into potentially ambiguous packed representations of LFG f-structures and how these were subsequently matched against the analyses produced by the German *ParGram* LFG for the construction of the training corpus.

4.1 The need for data in grammar development

In grammar development, the lack of large annotated test suites is a serious obstacle to the further extension and adaptation of the grammars concerned, because it makes it extremely costly to evaluate grammars systematically and to keep track of the consequences of grammar modifications on coverage, efficiency and accuracy. Without large LFG (or LFG-like) test suites for German, a linguist involved in the development of a broad-coverage grammar such as the German *ParGram* LFG can, of course, run the grammar on large corpora and state afterwards how many sentences in a given corpus were parsed, what percentage timed out or failed because of storage overflow, and how many did not get any analysis. It is virtually impossible, however, to determine the accuracy of the analyses obtained, which relativizes the informational value of the figures mentioned considerably.

Moreover, large collections of annotated LFG-parsed sentences are required for the supervised training of probabilistic disambiguation modules, such as the log-linear models mentioned in Chapter 3, which select the most probable parse out of the sometimes extremely numerous analyses typically proposed by hand-crafted grammars. Similarly, the empirically based tuning of the OT-mark-

driven disambiguation module presented in Chapter 7 is only possible with the help of large collections of at least partially disambiguated LFG-parsed data.

Since the purely manual creation of such large data collections would be extremely time-consuming, it seems reasonable to use an existing treebank, the TIGER Treebank (Brants et al. 2003) in our case, and to use it in order to determine which reading among all readings produced by the symbolic grammar used is the intended one. However, the two following circumstances cause the realization of this idea to be much less trivial than it might appear to be at first sight:

- The level of granularity of the TIGER Treebank annotations is inferior to the level of granularity of the information encoded in the representations produced by the German *ParGram* LFG. In other words, the German *ParGram* LFG makes certain morphosyntactic distinctions that are not encoded in the TIGER Treebank graphs.
- The hybrid phrase structure/dependency structure representations of the TIGER Treebank differ in so many ways from the c-structures constructed by the German *ParGram* LFG that it is unlikely that the constituent boundaries encoded in the TIGER Treebank could be used as a bracketing that restricts the possible analyses to the intended one(s).

The first point is a problem for the deterministic conversion of TIGER Treebank graphs into LFG representations, but fortunately, Riezler et al. (2002) have shown that it is possible to train log-linear models for syntactic disambiguation on partially labeled data. This means that sentences for which a proper subset of the LFG analyses can be identified as better than the remaining ones because they are compatible with the corresponding TIGER Treebank annotation can be used as training data. The second point prevents us from using for our purposes the approach used for establishing the training data for the disambiguation module of the English *ParGram* LFG (Riezler et al. 2002), which involves constraining the LFG analyses of Penn Treebank material via the Penn Treebank bracketing.

The most promising approach therefore is the conversion of TIGER Treebank graphs into LFG f-structures or, if unavoidable, into packed representations of several f-structures, so called f-structure charts. These can then be matched against the grammar output, and this way, the LFG analyses compatible with the TIGER Treebank annotations are identified. This approach also has the advantage of producing f-structures (or f-structure charts) for all TIGER Treebank sentences rather than being limited to the sentences that can be analyzed by the German *ParGram* LFG, even if the matching against the grammar output is evidently not possible for sentences that are outside of grammar coverage.

4.2 Similar work in treebank conversion

Efforts in constructing f-structure banks on the basis of treebanks have been reported on in van Genabith et al. (1999), Sadler et al. (2000), Frank (2000), van Genabith et al. (2001), and Cahill et al. (2002). Since in all that work the source format (AP Corpus, Susanne Corpus, Penn Treebank) differs considerably from the TIGER Treebank format in that it encodes mainly phrase-structural information, our approach is quite different from the ones mentioned. Because dependency information is expressed explicitly in the edge labels, we do not need to f-annotate the treebank. Rather we can directly convert the hybrid TIGER representation into f-structures. More importantly even, the automatic f-annotation of treebank trees only allows for one unambiguous f-structure per corresponding tree. Since we know, however, that some TIGER edge labels can correspond to various f-structure features (see Section 4.5), we prefer the approach of a (more or less) direct conversion.

Another related work is Frank (2001a,b), which involves the extraction of a Lexicalized Tree-Adjoining Grammar (LTAG) from the NEGRA corpus. Here, the source format is comparable to ours, the TIGER format being an extension of the NEGRA format, and the main differences with respect to our work are due to the different target format. For the conversion of the corpus to a collection of f-structures, constituency information is almost irrelevant, whereas it is crucial for the extraction of an LTAG.

Finally, our conversion is, of course, in many ways similar to the inverse conversion from LFG analyses to TIGER trees (Zinsmeister et al. 2002). We use for example the same term-rewriting system. However, since the relation between TIGER trees and f-structures is far from being a one-to-one mapping, the new direction raises new questions. Moreover, we aim to convert the entire TIGER Treebank into an ‘f-structure bank’ with hardly any human intervention, an objective that is quite different from grammar-based treebank annotation.

4.3 The TIGER Treebank and the relational TIGER representation

The TIGER Treebank comprises about 50,000 syntactically annotated German newspaper sentences. Release 1 of the TIGER Treebank, which was used for the work described in this chapter, comprises about 40,000 sentences. The annotation consists of generalized graphs, i.e. trees which may contain crossing and secondary edges. Edges are labeled, so that a TIGER tree encodes both phrase-structural information and information on dependency relations. For more details on the annotation scheme, the reader is referred to Brants et al. (1997), Skut et al. (1998), Brants & Hansen (2002) and Brants et al. (2002).

Trebank Conversion for the Construction of Training Data

The TIGER trees are represented in a specific XML format, the so-called TIGER XML (Mengel & Lezius 2000). Figure 4.2 illustrates what the TIGER XML representation of an annotated sentence like the one in Figure 4.1 looks like. The sentence is given (and glossed) in (4.1).

- (4.1) Auf Armeseite seien 35 Soldaten getötet und mindestens 200 weitere verwundet worden.
 On army side were 35 soldiers killed and at least 200 further wounded been.
 ‘On the side of the army, 35 soldiers had been killed and at least 200 others, wounded.’¹

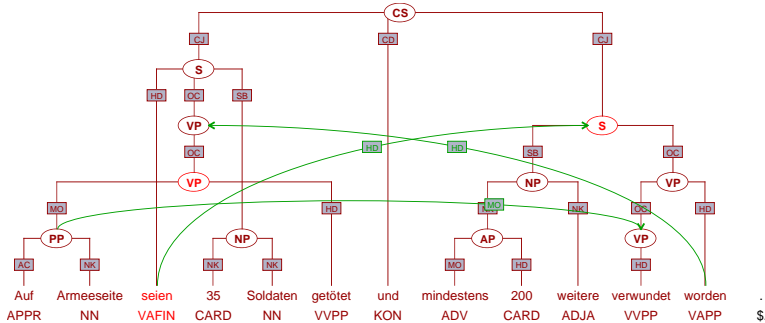


Figure 4.1: TIGER tree representation of (4.1)

In order to be able to use the XLE term-rewriting system for the conversion of the TIGER trees into f-structures, we first need to have the TIGER Corpus available in a relational Prolog-like representation that can be read into XLE. Instead of being a generalized graph, a TIGER tree then has to take the form of a feature structure.

This conversion raises a first problem: In a TIGER tree, there can be several identically labelled edges that go from one single node to several of its daughter nodes. In feature structures, on the contrary, a given attribute can only have one unique value. It is thus not possible to convert a TIGER tree into a feature structure by a one-to-one mapping. Fortunately, there is quite a straightforward solution to this problem: As attributes in a feature structure can be set-valued, all identically labeled daughter nodes of a given node can be put into a set. The

¹s1376

4.3 The TIGER Treebank and the relational TIGER representation

```
...
<t id="s1376_7" word="und" pos="KON" morph="--"/>
<t id="s1376_8" word="mindestens" pos="ADV" morph="--"/>
<t id="s1376_9" word="200" pos="CARD" morph="--"/>
<t id="s1376_10" word="weitere" pos="ADJA" morph="--"/>
<t id="s1376_11" word="verwundet" pos="VVPP" morph="--"/>
<t id="s1376_12" word="worden" pos="VAPP" morph="--">
  <secedge label="HD" idref="s1376_507" />
</t>
<t id="s1376_13" word="." pos="$. " morph="--"/>
</terminals>
<nonterminals>
<nt id="s1376_500" cat="PP">
  <edge label="AC" idref="s1376_1" />
  <edge label="NK" idref="s1376_2" />
  <secedge label="MO" idref="s1376_503" />
</nt>
<nt id="s1376_501" cat="NP">
  <edge label="NK" idref="s1376_4" />
  <edge label="NK" idref="s1376_5" />
</nt>
<nt id="s1376_502" cat="AP">
  <edge label="MO" idref="s1376_8" />
  <edge label="HD" idref="s1376_9" />
</nt>
...
```

Figure 4.2: excerpt of the TIGER XML representation of (4.1)

resulting representation differs somewhat from the initial tree, but it contains basically the same information.

Another problem that we need to deal with when converting TIGER trees into feature structures is the fact that, generally, the latter do not encode any information about precedence relations. This kind of information can be crucial, however, for subsequent steps in the conversion from one format to the other. Genitive attributes, for example, are labeled AG in the TIGER treebank, whether they are on the left or on the right of their head noun. The German *ParGram* LFG, on the contrary, analyzes them in two different ways, either as a SPEC POSS, when they are in prenominal position, or as an ADJ-GEN, when they appear postnominally. This means that a minimum amount of information about precedence needs to be encoded in the relational TIGER representation.

This can be done with the help of a special XLE predicate ‘scopes’ that

allows us to state that a certain node A precedes another node B. By means of ‘scopes’, we express precedence relations between daughters of the same mother node. This kind of information is sufficient to disambiguate all TIGER-LFG mismatches which can be disambiguated on the basis of precedence information.

The first step of the conversion of TIGER trees into f-structures thus consists of transforming the trees into feature structures. As this task does not require any major structural changes, it can be carried out quite comfortably by means of an XSL style sheet.² Figure 4.3 shows an excerpt of the relational Prolog-like representation of the corpus sentence displayed in Figure 4.1 (p. 26) that results from the XSL conversion of the TIGER XML representation illustrated in Figure 4.2 (p. 27). Figure 4.4 displays the corresponding feature structure.

```
...
, cf(1, eq(attr(var(6), 'TI-FORM' ), 'getötet'))
, cf(1, eq(attr(var(6), 'TI-ID' ), 6))
, cf(1, eq(attr(var(6), 'TI-POS' ), 'VVPP'))
, cf(1, eq(attr(var(504), 'MO' ), var(1011504)))
, cf(1, in_set(var(500), var(1011504)))
, cf(1, eq(attr(var(500), 'TI-CAT' ), 'PP'))
, cf(1, eq(attr(var(500), 'TI-ID' ), 500))
, cf(1, scopes(var(1), var(2)))
, cf(1, eq(attr(var(500), 'AC' ), var(1001500)))
, cf(1, in_set(var(1), var(1001500)))
, cf(1, eq(attr(var(1), 'TI-FORM' ), 'Auf'))
, cf(1, eq(attr(var(1), 'TI-ID' ), 1))
, cf(1, eq(attr(var(1), 'TI-POS' ), 'APPR'))
, cf(1, eq(attr(var(500), 'NK' ), var(1012500)))
, cf(1, in_set(var(2), var(1012500)))
, cf(1, eq(attr(var(2), 'TI-FORM' ), 'Armeeseite'))
...
```

Figure 4.3: excerpt of the relational Prolog-like representation of (4.1)

4.4 Lemmatization

LFG f-structures are assumed to be a first level of abstraction towards a semantic representation of sentences. Therefore, they generally do not encode

²Thanks to Hannes Biesinger for a first version of the XSL style sheet and to Stefanie Dipper for her contribution to its final adaptation.

4.4 Lemmatization

"Auf Armeeseite seien 35 Soldaten getötet und mindestens 200 weitere verwundet worden . "

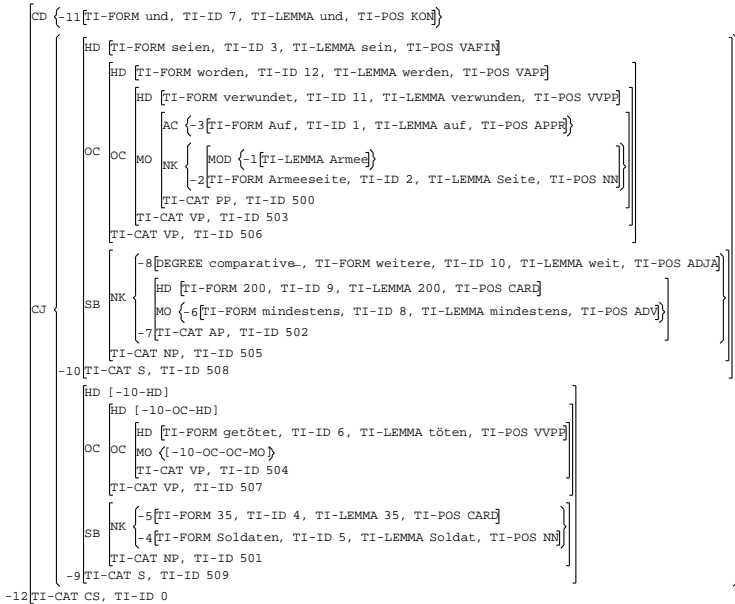


Figure 4.4: representation of (4.1) as a TIGER-annotated feature structure

surface properties such as word forms, but factor the information coming from words into semantic forms or PREDs and morphosyntactic features. Semantic forms are named according to the *lemma* of the corresponding word. However, Release 1 of the TIGER Treebank does not encode the lemmas of words in the corpus sentences at all, and in Release 2, the conventions applied in lemmatization differ somewhat from the conventions used in the German *ParGram* LFG. The most important difference is that compounds are not decomposed in the TIGER Treebank, whereas the finite-state morphology used in the German *ParGram* LFG does decompose them, since compounds are productive formations in German. Other differences are mostly found in the lemmatization of closed class items, such as determiners and pronouns, but also in the lemmatization of participles and adjectives that do not occur in an uninflected form, e.g. *allerbeste(m/n/r/s)*.

In order to derive f-structures from TIGER Treebank graphs, it was therefore necessary to lemmatize the unlemmatized representations or to modify

the lemma information in the lemmatized ones respectively. This is done with the help of a Perl program that we developed and that makes use of the finite-state morphology employed in the German *ParGram* LFG. Each word form of the sentence being lemmatized is analyzed morphologically and the resulting lemma is encoded in the relational TIGER representation of the sentence. If several lemmas are possible for a word or if a compound can be decomposed in several ways, the alternatives are compared to the lemma originally encoded, if available, and all lemmas compatible with the original information are encoded as alternatives in the relational TIGER representation, which becomes ambiguous in this step if there remain several possible morphological analyses. For words that cannot be analyzed by the finite-state morphology, the Perl program assumes that the lemma is the lemma annotated by the TIGER annotators or the surface form of the word, if no lemma is annotated.

4.5 Treebank conversion by (MT) transfer rules

Although the f-structures we obtain from the German *ParGram* LFG and the TIGER treebank representations coincide in core aspects, e.g. the encoding of grammatical functions, there are mismatches in analysis details that are comparable to translation mismatches in natural language translation. One such phenomenon is the flat analysis of auxiliary constructions generally adopted in LFG versus the intricate analysis that has been chosen for the TIGER treebank. This kind of mismatch motivates the use of transfer technology originally developed for machine translation.

4.5.1 The transfer system

The transfer system we use is a term-rewriting system based on Prolog. It was originally developed by Martin Kay and is now part of the XLE grammar development platform. The rules it processes are ordered, which means that the output of a given rule r_i is input to rule r_{i+1} . Each rule replaces a certain set of predicates (those on the left-hand side of the rule) by another set of predicates (those on its right-hand side). Input and output predicates are separated by a rewriting symbol, the operator ‘ \Rightarrow ’. The most basic rules simply rewrite the name of the predicate and pass on the values of the arguments unchanged. For example, the rule given in (4.2) maps the TIGER edge label OA (accusative object) to the LFG function OBJ.

$$(4.2) \quad \begin{array}{l} \text{oa}(X, Y) \\ \Rightarrow \\ \text{obj}(X, Y) . \end{array}$$

4.5 Treebank conversion by (MT) transfer rules

In addition, it is possible to specify predicates on the left-hand side that have to be matched, but are not replaced (marked with a '+'), as well as predicates that must not be matched for the rule to be applied (marked with a '-'). These mechanisms are used in the following rule, which takes a partial feature structure whose attribute TI-POS has the value PIAT (for 'attributive indefinite pronoun') out of the set that is the value of the feature NK (for 'noun kernel') and attributes it to a new feature SPEC QUANT, if that feature does not yet exist.

```
(4.3)  +nk(A,SET), in_set(B,SET), +ti_pos(B,'PIAT'), -spec(A,-)
      ==>
      spec(A,SPEC), quant(SPEC,B).
```

It is also possible to delete features by writing a zero on the right-hand side of a rule, which stands for the empty set. In this case, all predicates on the left-hand side of the rule are deleted from the set of terms without replacement.

```
(4.4)  ti_form(-,-)
      ==>
      0.
```

Finally, the possibility of defining rules as optional needs to be mentioned as well. Optional rules are characterized by the use of the operator '?=>' instead of '=>'. They allow us to transfer a given input feature structure to two alternative output structures — or more, if several optional rules are applied. We can thus handle cases where we cannot clearly decide solely on the basis of the input what the output must look like. The TIGER label MO (for 'modifier'), for example, is such a phenomenon, because the context is not always sufficient to determine whether it is to be transferred to an element of the set-valued feature ADJUNCT, to an OBL-DIR (directional oblique), an OBL-LOC (locative oblique) or still another grammatical function. The following rule optionally transfers a MO-PP with an AC (the edge label used for adpositions in TIGER) that has the form 'nach' into an OBL-DIR.

```
(4.5)  +ti_cat(S,'S'), +mo(S,MO), in_set(PP,MO), +ti_cat(PP,'PP'),
      +ac(PP,APPR), +ti_form(APPR,'nach')
      ?=>
      obl_dir(S,PP).
```

For reasons of userfriendliness and maintainability, the XLE transfer system also allows the use of templates and macros. They are shorthand notations for sets of rules and predicates respectively. As they are not directly relevant for our presentation, however, we do not present them here in more detail. The interested reader is referred to the relevant sections of Crouch et al. (2006).

4.5.2 Transfer phenomena

Unlike transfer in machine translation, the transfer from TIGER trees to LFG f-structures does not aim to change the surface string. The task is rather to map a limited set of grammatical features into another limited set of grammatical features. Nevertheless, the format conversion is far more complex than a simple mapping from one feature set to another, because (i) there is no one-to-one correspondence between features and (ii) the different analyses chosen for certain grammatical phenomena can have relatively heavy repercussions on the structure of the representations involved.

Ambiguous edge labels

In Section 4.3, we mentioned the case of the TIGER edge label AG, which, depending on the position of the AG constituent with respect to its head noun, corresponds to either a SPEC POSS feature or an ADJ-GEN feature in a German LFG analysis. Still, this kind of ambiguity can easily be resolved on the basis of precedence information, so that we simply need two obligatory rules for the transfer of AGs, one for prenominal ones and a ‘default rule’ for postnominal ones. As rules are ordered, the ‘default rule’ is only applied if the more specific rule was not.

- (4.6)(a.) `+ti_cat(NP, 'NP'), +nk(NP, NKSET), +in_set(HEAD, NKSET),`
`+ti_pos(HEAD, 'NN'), +scopes(AG, HEAD), ag(NP, AG)`
`==>`
`spec(NP, SPEC), poss(SPEC, AG).`
- (b.) `ag(NP, AG)`
`==>`
`adjunct(NP, ADJUNCT), in_set(AG, ADJUNCT).`

A somewhat more complex case is the transfer of the predicate MO. It can correspond to the predicates ADJUNCT, OBL-DIR, OBL-LOC and OBL-MANNER. This is due to the fact that PPs such as *auf dem Operationstisch* in (4.7) are analysed as subcategorized arguments in the German LFG (see Figure 4.6 on p. 34), and not as ADJUNCTS (or MOs respectively), as it is the case in the TIGER Treebank (see Figure 4.5).

- (4.7) Weil er bei seiner Morgentoilette Zeit sparen wollte ..., ist ein
 Because he in his morning hygiene time save wanted ..., is a
 17jähriger Zimmerer in Wales auf dem Operationstisch gelandet.
 17-year-old carpenter in Wales on the operation table landed.
 ‘A 17-year-old carpenter ended up on the operating table in Wales because
 he wanted to save time in his morning hygiene ...’³

4.5 Treebank conversion by (MT) transfer rules

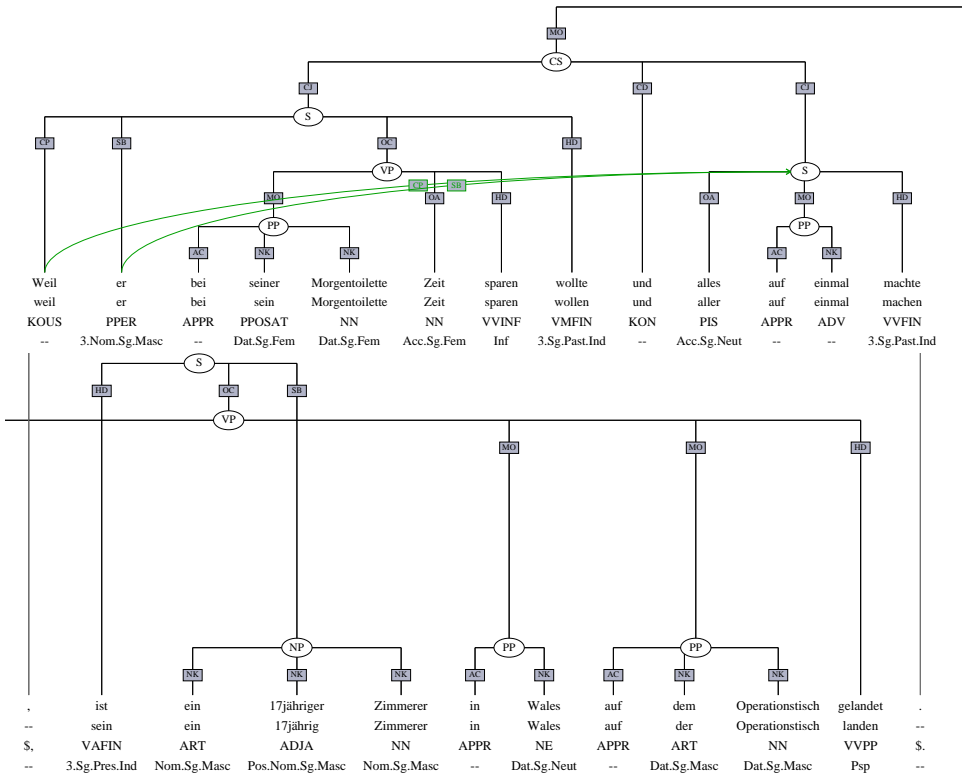


Figure 4.5: TIGER tree representation of (4.7)

We deal with this case by first using the optional rule in (4.8a), which similarly to the one in (4.5) converts a MO into an OBL-LOC, and then applying the default rule given in (4.8b), which transfers all MOs to ADJUNCTS. In order not to obtain too many output f-structures, we try to limit the application of the optional rules to as few contexts as is reasonably possible, while keeping them general enough to cover all cases that we need for a justifiable comparison of the output of the German LFG and the TIGER annotation. Unfortunately though, due to the fact that there are often multiple ambiguous edge labels in a given TIGER graph, the f-structure chart derived from it contains numerous

³s23751

Trebank Conversion for the Construction of Training Data

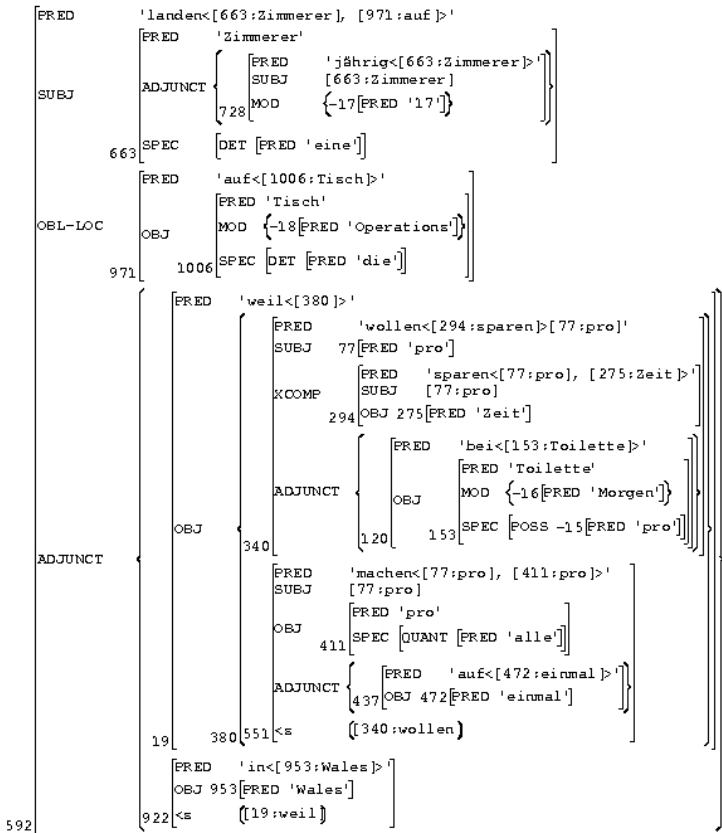


Figure 4.6: f-structure associated with (4.7) by the German *ParGram* LFG

readings.

- (4.8)(a) +ti_cat(S, 'S'), +mo(S,MO), in_set(PP,MO),
 +ti_cat(PP, 'PP'), +ac(PP,APPR), +ti_form(APPR, 'auf')
 ?=>
 obl_loc(S,PP).
- (b.) mo(S,MO)
 ==>
 adjunct(S,MO).

Structural changes

Given that TIGER trees on the one hand encode information about both phrase structure and dependency relations and that f-structures on the other hand only represent the latter type of information, it is not surprising that the analyses of a few grammatical phenomena differ considerably between the TIGER Corpus and the German LFG analyses. This is the case of analytic tenses and passives, for example, which generally get a flat analysis in LFG, the auxiliary and the main verb being treated as f-structure co-heads, whereas in TIGER the VP containing the non-finite main verb form is analysed as a clausal object (OC) of the auxiliary. Figure 4.7 shows a TIGER tree containing an analytic verb form and Figure 4.8 (p. 36), the corresponding f-structure.

- (4.9) Die landwirtschaftliche Nutzung sei dort untersagt worden.
 The agricultural exploitation was there prohibited been.
 'Agricultural exploitation had been prohibited there.'⁴

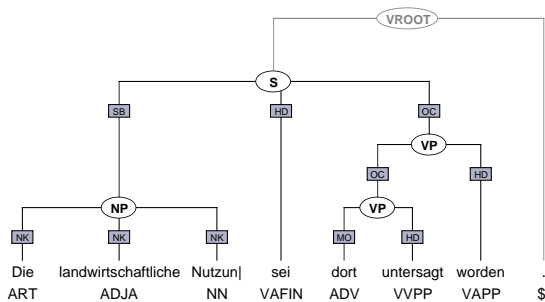


Figure 4.7: TIGER tree representation of (4.9)

This kind of structural change is known as head-switching in the field of machine translation. As studies about the treatment of head-switching phenomena, e.g. Dorna et al. (1998), have shown, they can be dealt with without major difficulty by a term-rewriting system.

Another phenomenon for which structural changes have to be made on the way from a TIGER tree representation to an f-structure is the attachment of ADJUNCTS or MOs that modify a verb which is embedded under a modal verb. For the German LFG it has been decided that this kind of ADJUNCT is a feature of the outer f-structure and not of the XCOMP sub-f-structure within it. This analysis helps to avoid a systematic ambiguity, which would arise if the attachment

⁴s2458

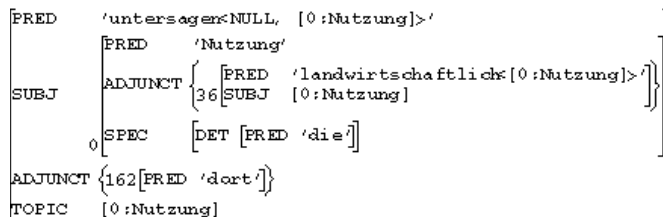


Figure 4.8: f-structure associated with (4.9) by the German *ParGram* LFG

to both the outer and the embedded verb were allowed, but which generally is not of importance from a semantic point of view. According to the TIGER annotation principles, a MO is always attached where it belongs semantically, which means that in most cases it is embedded in the TIGER counterpart of XCOMP, namely OC (for 'clausal object'); in unclear cases, it is attached as low as possible. This difference in adjunct attachment can be observed between Figures 4.5 (p. 33) and 4.6 (p. 34) with respect to the PP *bei seiner Morgentoilette*.

Again, this and all other kinds of structural changes needed for the conversion of TIGER trees into f-structures can be handled quite comfortably with a term-rewriting system.

4.6 Evaluation of the treebank conversion procedure

A rough evaluation of our treebank conversion has been carried out on the basis of sentences 8,001 through 8,200 of the TIGER Corpus.⁵ For this purpose, we established a gold standard for those sentences using roughly the same methodology and format as King et al. (2003) did for the PARC 700 Dependency Bank. Then the result of our treebank conversion was matched against this gold standard, taking into account predicate-argument relations only. (Release 1 of the TIGER Treebank was not yet annotated morphologically.) Of the 200 sentences, six consisted of one single word, which makes it impossible to match predicate-argument relations, and ten could not be converted to feature structures, because they were sequences of syntactically unconnected phrases rather than sentences (example: *dah FRANKFURT A. M. , 6. November .*). This left us with 184 sentences for evaluation, of which nine did not entirely match

⁵Aoife Cahill (then at Dublin City University) and I decided once by means of a random choice that sentences 8,001 through 10,000 would serve as unseen test section for work done on the basis of the TIGER corpus.

4.7 Matching grammar output against TIGER-derived f-structure charts

due to errors in the original TIGER annotation, one failed to match due to an erroneous lemmatization and nine had not been transferred correctly. This means that 174 out of the 184 sentences were converted correctly, which corresponds to 95%. However, due to the errors in the original TIGER annotation, the proportion of sentences for which an entirely correct f-structure was included in the corresponding f-structure chart was lower, namely about 90%.

4.7 Matching grammar output against TIGER-derived f-structure charts

The f-structure charts automatically derived from TIGER Treebank graphs cannot be used as training data for the estimation of the parameters of a probabilistic disambiguation module, since nothing in them encodes how they relate to the output of the symbolic grammar. It is therefore necessary to match the f-structure charts produced by the grammar against the TIGER-derived f-structure charts.

For this purpose, we developed a Perl program that compares f-structures and, by extension, f-structure charts. It takes as input an f-structure chart produced by the grammar and a reference f-structure chart, which in our case is the f-structure chart derived from the corresponding TIGER graph. Then, it compares each f-structure contained in the former with each f-structure included in the latter. This can result in a very high number of matching operations to be performed, which is why the program can take an optional argument that specifies an upper limit for the number of matching operations that are to be performed between pairs of f-structure charts.

This Perl program can produce two types of output: (i) reference-compatible f-structure charts, which will be used as training data for log-linear models as the ones presented in Chapters 8 and 9, and (ii) OT mark profiles for all readings contained in the f-structure chart produced by the grammar with the reference-compatible profiles marked as target winners. These OT mark profiles are used for the corpus-based trimming of the OT-mark-driven disambiguation module presented in Chapter 7.

The result of matching against the TIGER-derived f-structure charts the 37,546 full parses produced when analyzing the 50,000 TIGER Treebank sentences with the version of the German *ParGram* LFG that we used for the construction of the training data is shown in Table 4.1 (p. 38). It shows that, for 19,660 parses, no reading contained in them could be identified as TIGER-compatible, either because the TIGER-derived f-structure chart is erroneous or because the parse does not include the intended reading. As it is not possible to determine the exact cause without a considerable amount of manual labor, we do not know what the more frequent cause is, although the results of the eval-

no match	19,660
too many readings to be compared	1,553
all readings compatible	7,064
proper subset of readings compatible	9,269
total of sentences with spanning analysis	37,546

Table 4.1: Result of matching grammar output against TIGER-derived f-structure charts

uation of the treebank conversion reported above indicate that most TIGER-derived f-structure charts do include the correct f-structure. For 1,553 sentences, the pairs of f-structure charts could not be matched against each other in a reasonable amount of time because the matching would have involved more than 10,000 f-structure matching operations. For 7,064 out of the 37,546 sentences, all readings included in the parses were compatible with some reading in the corresponding TIGER-derived f-structure chart. These structures cannot be used for discriminative training because the TIGER graph annotations do not provide enough information to distinguish the intended analysis from the unintended analyses. Finally, for 9,269 sentences, a proper subset of the readings contained in the parses could be determined as compatible with the TIGER graph annotations. The reference-compatible f-structure charts associated with these sentences can be used as labeled training data.

The final training set comprises 8,881 pairs of unlabeled and labeled f-structure charts, since some of the 9,269 sentences mentioned above are part of our held-out and test sets. The unlabeled f-structure charts in this training set contain 25.94 readings on average. The labeled f-structure charts contain 3.20 readings on average. Thus, 12.3% of the readings proposed for these sentences by the grammar are TIGER-compatible.

4.8 Summary

In this chapter, we have argued that syntactically annotated corpora are very useful, if not indispensable, for the development of hand-crafted broad coverage grammars. The most important aspect in our context is that they allow for the construction of labeled data for the supervised training of disambiguation modules.

For the construction of labeled LFG-parsed data, we have converted the TIGER Treebank graphs into f-structure charts in several steps, and finally we have matched the analyses produced by the German *ParGram* LFG against these TIGER-derived f-structure charts. This way, we have established 9,269 labeled f-structure charts, out of which 8,881 can be used for training.

Chapter 5

The TiGer Dependency Bank — A Dependency-Based Gold Standard for German Parsers

In this chapter, we document how the TIGER-derived packed f-structure representations were disambiguated and hand-checked for the creation of a gold standard for German parsers, the TiGer Dependency Bank, comparable in granularity and representation format to the f-structures produced by the German *ParGram* LFG and hence usable for the evaluation of our system. We especially address the differences between the TIGER Treebank graphs and the structures annotated in the TiGer Dependency Bank.

5.1 A gold standard for (hand-crafted) German parsers

As noted at the beginning of Chapter 4, syntactically annotated data are indispensable for an informative (and potentially comparative) evaluation of parsers. In other words, in order to determine the quality of a parser's output, we need a gold standard for German parsers, suited both for treebank-induced parsers/grammars and for hand-crafted grammars. Given that there are data collections like the TIGER Treebank, one may be tempted to use these. However, the graphs of the TIGER Treebank themselves are difficult, if not impossible, to use as a gold standard for German parsers that were not induced from this same treebank for a number of reasons: The constituency annotation in the TIGER graphs has the advantage of being fairly theory-neutral, but (i) since it includes discontinuous constituents and secondary edges, it cannot be mimicked by any of the hand-crafted German parsers that we are aware of. Besides, (ii) the tokenization (and lemmatization) of certain multi-word ex-

pressions, compounds etc. differs from the analyses most hand-crafted parsers obtain. The functional annotation in the TIGER graphs is more suitable for the evaluation and comparison of parsers across theoretical frameworks, but (iii) it is intimately tied to the constituency annotation, including edge labels such as ADC¹ (multi-token adjective component), which only exist due to the lemmatization decisions mentioned above, and (iv) it does not encode all information and distinctions that deep parsers are supposed to obtain.

Similar problems arise with other syntactically annotated corpora of German text and speech, such as the NeGra Corpus (Skut et al. 1998, Brants et al. 1999), the Verbmobil Corpus (Wahlster 2000) and the Tübingen Treebank of Written German (Telljohann et al. 2003), since they all encode constituency and dependency information in one structure, the latter being biased by the former. Grammar developers, however, are interested in pure dependency representations, which allow for a much more meaningful evaluation than the bracketing of constituents, and have therefore clearly been moving away from treebanks to dependency banks (Carroll et al. 1999, 2003).

Therefore, it was decided in the TIGER Project to establish a purely dependency-based gold standard for German parsers for a part of the TIGER Corpus. The size of this gold standard was determined to be of 2,000 sentences, and the corpus section from sentence # 8,001 to sentence # 10,000 was randomly selected as to be annotated with the corresponding dependency annotations. This gold standard is called the TiGer Dependency Bank (henceforth TiGer DB), and since 132 sentences in the corpus section mentioned either consist of just one word or of a sequence of syntactically unconnected words, it comprises 1,868 structures. The TiGer DB is annotated with so-called dependency triples, i.e. a functor representing a grammatical relation or feature and two arguments representing the head and the value of this feature respectively. The format of these dependency triples is the same as in the PARC 700 Dependency Bank (King et al. 2003), which makes it possible to use the tools for displaying and pruning structures that are available together with this English dependency bank, which was also constructed for the purpose of parser evaluation. The grammatical relations encoded in the TiGer DB are to a fair extent identical to the edge labels used in the TIGER Treebank; in order to make it more suitable as a basis for the evaluation of deep German parsers, additional distinctions have to be made in the set of grammatical relations, however, which leads to an enlarged set of features compared to the TIGER Treebank.

¹Upper case labels are used for the functional annotation encoded in the TIGER Treebank and lower case labels for the dependencies encoded in the TiGer DB.

5.2 Constructing the TiGer DB

For the construction of the TiGer DB, an approach consisting in a combination of automatic and manual techniques was chosen. The idea was to achieve the most accurate and consistent results in a reasonable amount of time. The basic process is as follows:

1. Convert each TIGER Treebank graph into an f-structure chart (packed representation of one or several f-structures) according to the method presented in Chapter 4.
2. Match the resulting f-structure chart against the output of the German *ParGram* LFG (again according to the method laid out in Chapter 4) and bank the compatible reading(s).
3. For all sentences for which there are either several or no compatible readings, select the correct/best analysis manually.
4. Fully automatically convert the selected f-structure into dependency triples.
5. Manually check/correct each structure using the pretty-printing and validation tools that are distributed with the PARC 700 Dependency Bank.

5.2.1 Automatic derivation of f-structure charts from the TIGER Treebank

The conversion of TIGER graphs into f-structure charts is described in Chapter 4. It takes the TIGER graphs encoded in TIGER XML as input and produces f-structure charts. The ambiguity in the mapping from TIGER graphs to f-structure charts is because of missing information in the TIGER Treebank, such as information concerning the decomposition of compounds, the argument vs. adjunct status of phrases labeled as MOs (modifiers), etc. In the conversion process it can be dealt with by means of optional rules, but in order to be used as a gold standard, the resulting output has to be disambiguated, of course. How this can be done is discussed in Subsections 5.2.2 and 5.2.3.

Apart from changes due to the shift from one representation to another, we decided to perform some changes to the analyses chosen by the TIGER Treebank annotators as well. These latter changes are motivated by the fact that the treatment given to these phenomena by all German parsers that we are aware of differs from the analysis in the TIGER Treebank in a systematic way. One of these changes concerns PPs that are extracted from NPs, such as *statt dessen* in (5.1).

- (5.1) Statt dessen gestand ihnen die Regierung eine Entschädigung zu:
Instead this-GEN conceded them the government an indemnity to.
'Instead (of this), the government conceded them an indemnity.'²

In the TIGER Treebank, this PP is attached as an MNR (noun modifier) to the NP *eine Entschädigung*. Current hand-crafted parsers for German, however, would attach this PP to the verb, since the attempt to attach it to the NP would result in a massive increase in ambiguity. For a gold standard for German parsers, we consider it reasonable to encode the latter attachment rather than one that no parser would be able to achieve and which, moreover, is semantically debatable.

5.2.2 Automatic disambiguation of TIGER-derived f-structure charts

As a first step towards disambiguating the f-structure charts resulting from the fully automatic treebank conversion, these are matched against the output of the German *ParGram* LFG, as described in Section 4.7 of the last chapter, and the compatible reading(s) are saved. Of course, this matching can only be performed for sentences that are assigned a full parse by the German *ParGram* LFG, and although the information both in the TIGER graphs and in the LFG parses is relatively detailed, it can be impossible to fully disambiguate. Typical remaining ambiguities are due to the decomposition of compounds and to person and number ambiguities of possessive determiners and pronouns, these pieces of information not being included in the TIGER Treebank.

Moreover, it has to be kept in mind that the matching against the LFG output does not always retain the correct analysis, although most ambiguities can be resolved correctly in this way. This is particularly true for the *mo* (modifier) vs. *op* (prepositional object) distinction; an incomplete lexicon entry in the LFG can lead to the selection of the *mo* reading, even if the *op* reading is more adequate.

Nevertheless, the automatic matching of the TIGER-derived f-structure charts against the output of the German *ParGram* LFG is extremely useful. Not only does it help to eliminate, or at least reduce, the ambiguity in the representations, but it also helps to increase consistency in the gold standard, since every time a match between the TIGER-derived f-structure chart and the grammar output is expected but cannot be achieved, the human annotators can pay special attention to the phenomenon that caused the match to fail.

²s14841

5.2.3 Manual disambiguation of TIGER-derived f-structure charts

All ambiguous TIGER-derived f-structure charts that cannot be fully disambiguated in the previous step have to be disambiguated manually. This is performed by visualizing the structures in the grammar development tool XLE (Maxwell & Kaplan 1993, Crouch et al. 2006). It displays the packed representation of all f-structures encoded, the currently selected f-structure and an additional window, where the alternatives with the information differing among them are visualized. This allows human annotators to choose and save the correct reading. When none of the readings can be considered correct, the best analysis is selected and the annotator puts the sentence number on a list of structures to be reconsidered in the validation step.

For mildly ambiguous structures, this manual disambiguation step can be carried out relatively comfortably with the help of XLE. However, when disambiguating highly ambiguous structures, the annotators found it extremely hard to select exactly the correct f-structure, since XLE displays the discriminants between readings in a fashion that is difficult to read in the case of highly ambiguous structures and it does not fix choices that the annotator has made. In conclusion, XLE, which is *not* and has never been conceived as a treebanking tool, can reasonably be used for treebanking when the structures involved are mildly ambiguous; for handling more complex representations, however, it proved to be cumbersome to use. For future annotation efforts similar to the construction of the TiGer DB, the tools developed in the TREPIL Project of the University of Bergen (Rosén et al. 2005), will probably provide an interesting alternative.

5.2.4 Conversion into dependency triples and validation

The conversion from f-structures to dependency triples is fully automatic and unambiguous. It is carried out in basically the same way as it was done for the PARC 700 Dependency Bank (King et al. 2003). It mainly involves a certain amount of “flattening”, i.e. articulate f-structures without a PRED have to be restructured, but this can be done without any loss of information. In addition to the flattening, a certain amount of renaming and reorganizing has to be carried out in order to make the structures meet the annotation principles outlined in Subsection 5.3.

In a final (and very important) step, each TiGer DB structure is manually evaluated by an expert. If the structure is not correct, changes are made in the text-based representation of the structure. For this step, it is essential to use the pretty-printing and validation tools that are distributed with the PARC 700 Dependency Bank, as the visualization they facilitate speeds up the validation process considerably. Nevertheless, the manual effort required in this final step

was more than we had expected initially. To give a rough estimate, it probably involved more than six person months.

5.3 Grammatical relations and features encoded in the TiGer DB

The choice of the format and the dependencies encoded in the TiGer DB is crucial for its possible uses. Therefore the contents of the TiGer DB structures themselves are discussed in this subsection. First we discuss indices, reentrancies and lemmatization. We then present the grammatical relations we have decided to encode in the TiGer DB and finally the atomic features chosen.

5.3.1 Indices, Reentrancies and Lemmatization

Just as in the PARC 700 Dependency Bank, all predicates in a given TiGer DB structure are assigned a unique index. For displaying reasons, the matrix predicate is always assigned the index 0. All other predicates are assigned the index corresponding to the ID of the terminal node in the TIGER Treebank that it relates to. Predicates which do not clearly relate to a terminal node in the TIGER Treebank are given a ‘new’ arbitrary index. (This is the case for the compound non-head *privat~1001* in Figure 5.2 on p. 46, for example.)

The use of indices has a number of advantages: First, they help to distinguish two instances of the same word. Second, they permit the expression of reentrant structures, i.e. structures in which a single item is related to more than one predicate. This occurs with controlled infinitives and with predicative constructions.

Consider the sentence in (5.2) as well as its TIGER graph representation (in TIGER XML) and its representation as dependency triples, which are shown in Figures 5.1 and 5.2 (p. 46) respectively.

- (5.2) Privatmuseum muß weichen
Private museum must leave
‘Private museum must leave’³

The matrix predicate of the sentence is the verb *müssen*; this verb is thus assigned the index 0. All other predicates are assigned the indices corresponding to the terminal nodes they relate to, which are 1 for *(Privat)Museum* and 3 for *weichen*. The ‘new’ predicate *privat*, whose existence is due to the decomposition of compounds in the TiGer DB, is assigned a new unique index calculated on the basis of the index of its head and the position of the compound non-head

³s8597

5.3 Grammatical relations and features encoded in the TiGer DB

```
<s id="s8595">
<graph root="s8595_500">
  <terminals>
    <t id="s8595_1"
      word="Privatmuseum"
      pos="NN" morph="Nom.Sg.Neut"/>
    <t id="s8595_2" word="muß"
      pos="VMFIN" morph="3.Sg.Pres.Ind"/>
    <t id="s8595_3" word="weichen"
      pos="VVINF" morph="--" />
  </terminals>
  <nonterminals>
    <nt id="s8595_500" cat="S">
      <edge label="SB" idref="s8595_1"/>
      <edge label="HD" idref="s8595_2"/>
      <edge label="OC" idref="s8595_3"/>
    </nt>
  </nonterminals>
</graph>
</s>
```

Figure 5.1: TIGER XML representation of (5.2)

within the compound, which turns out to be 1001 in the present example. The fact that the subject of the embedded verb *weichen* shares its structure with the subject of the top verb *müssen* is expressed by the two triples *sb(müssen~0, Museum~1)* and *sb(weichen~3, Museum~1)*.

Concerning the grammatical relations encoded, the dependency triple representation shows both similarities and differences to the TIGER XML graph representation. Just as the latter, it annotates *(Privat)Museum* as the *sb* (subject) of *müssen*, but in contrast to the graph, it also annotates it as the *sb* of *weichen*. In addition, the *OC* (clausal object) edge label from the graph is reinterpreted as an *oc_inf*, since the related phrase is an infinite clausal object.

The above example also demonstrates the lemmatization applied in the TiGer DB. Verb forms are lemmatized to the infinitive, nominal forms to the nominative singular etc. Compounds are split up into their components, of which the head is used in the predicate name and the others are *mod* dependents of this head. In order to keep track of the original compound form, the lemma of the compound is encoded as the value of the feature *cmpd_form*, as can be seen in Figure 5.2 (p. 46).

The TiGer Dependency Bank

```
case(Museum~1, nom),
compd_form(Museum~1, Privatmuseum),
gend(Museum~1, neut),
mod(Museum~1, privat~1001),
mood(müssen~0, indicative),
num(Museum~1, sg),
oc_inf(müssen~0, weichen~3),
pers(Museum~1, 3),
sb(müssen~0, Museum~1),
sb(weichen~3, Museum~1),
tense(müssen~0, pres)
tiger_id(müssen~0, 2)
```

Figure 5.2: Dependency triple representation of (5.2)

5.3.2 Grammatical relations

The most difficult decisions in creating the TiGer DB involve choosing the grammatical relations to be encoded. Which dependencies are needed in the final application differs from framework to framework, and the names they are given vary from grammar to grammar. As a guideline, it has been decided to stick to the functional annotation in the TIGER Treebank, i.e. the edge labels in the TIGER graphs. Additional distinctions were introduced where the TIGER Treebank annotations seemed not to make all the distinctions current deep parsers of German make. The most striking example of a TIGER Treebank edge label which is treated in a number of different ways by German parsers is *MO*, which can be a truly optional modifier (still labeled as *mo* in the TiGer DB), but also a predicative argument (labeled as *pd* in the TiGer DB) or a (more or less) obligatory directional or locative argument (labeled as *op_dir* and *op_loc* respectively).

Grammatical relations defined like in the TIGER Treebank

The grammatical relations that are encoded identically in the TIGER Treebank (or in a former version of it) and in the TiGer DB are:

- *cj* – conjunct of a coordination
- *da* – objects in the dative
- *gr* – genitive attribute on the right of its head noun
- *oa* – direct objects in the accusative
- *oa2* – secondary objects in the accusative

- og – objects in the genitive
- op – prepositional objects
- pg – *von*-PPs considered as pseudo-genitives
- rc – relative clauses
- sbp – logical subjects of verbs in the passive
- vo – vocatives

Subjects

Although all SBs (subjects) of the relevant TIGER Treebank graphs appear as *sbs* in the corresponding TiGer DB annotations, there is one difference with respect to subjects: The TiGer DB encodes more subjects than the TIGER Treebank. In addition to subjects of inflected verb forms, these are the following:

- **sbs within *oc_infs*, infinitival sbs and *mos*:** *oc_infs* (infinite clausal objects, see page 51), as well as infinitival *sbs* (subjects) and *mos* (modifiers, see page 50) *always* contain a *sb*. In the case of *oc_infs* of raising verbs, this *sb* is indicated by coindexation, as in Figure 5.2, corresponding to (5.2), where the *sb* of the verb *weichen* is coindexed with the *sb* of the modal verb *müssen*, and Figure 5.3 (p. 48),⁴ corresponding to (5.3), where the *sb* of the verb *festnageln* is coindexed with the *oa* (accusative object) of the verb *lassen*. In the case of *oc_infs* of equi verbs and of infinitival *sbs* and *mos*, the *sb* is filled by a null pronoun, as in Figure 5.3, where the *sb* of the verb *lassen* is a null pronoun, and Figure 5.4 (p. 49), corresponding to (5.4), where this is the case for the *sb* of the verb *zurückhalten*.

- (5.3) ... , ohne sich auf deren Umfang festnageln zu lassen.
 ... without himself on these-GEN amount nail down to let.
 ‘... without letting himself be nailed down to the amount of these.’⁵
- (5.4) ... hat ... vorgeworfen, wichtige Informationen über
 ... has ... accused important informations about
 Kriegsverbrechen in Bosnien zurückzuhalten.
 war crimes in Bosnia to withhold.
 ‘... has accused ... of withholding important information about war
 crimes in Bosnia.’⁶

```

| mo | pred 'ohne'
    | obj | pred 'lassen'
        | oa [16] | pred 'pro'
            | case acc
            | num sg
            | pers 3
            | pron_type refl
        | oc_inf | pred 'fest#nageln'
            | pass_asp modal
            | op | pred 'auf'
                | obj | pred 'Umfang'
                    | case acc
                    | gend masc
                    | num sg
                    | gl | pred 'pro'
                        | case gen
                        | gend fem
                        | num sg
                        | pron_type demon
                | sb [16: pro]
            | sb | pred 'pro'
                | pron_type null

```

Figure 5.3: Dependency triple representation of (5.3)

- **sbs within pds:** Whether a predicative argument (pd) is subject-controlled or object-controlled is *always* indicated by a coindexed sb. This is illustrated in Figure 5.5 (p. 50), corresponding to (5.5), where the dependency triple `sb(mitverantwortlich~10, Regierung~9)` encodes that the pd of the verb *machen* is object-controlled.

(5.5) Der DIHT macht die Regierung für die eingetrübte Stimmung
 The DIHT makes the government for the tarnished vibes
 mitverantwortlich.
 co-responsible.
 ‘The DIHT holds the government for co-responsible for the tarnished
 vibes.’⁷

⁴For better readability, we show all following sample TiGer DB structures as they are displayed by the pretty-printing tool that comes with the PARC 700 DB.

⁵s9966

⁶s9034

⁷s9992

5.3 Grammatical relations and features encoded in the TiGer DB

```

| pred 'vor#werfen'
| mood ind
| perf +
| tense pres
| oc_inf | pred 'zurück#halten'
          | oa | pred 'Information'
            | case acc
            | gend fem
            | num pl
            | mo | pred 'wichtig'
              | degree pos
            | op | pred 'über'
              | obj | pred 'Verbrechen'
                | case acc
                | compd_lemma Kriegsverbrechen
                | gend neut
                | num pl
                | mod | pred 'Kriegs'
                | mo | pred 'in'
                  | obj | pred 'Bosnien'
                    | case dat
                    | gend neut
                    | num sg
          | sb | pred 'pro'
            | pron_type null

```

Figure 5.4: Dependency triple representation of (5.4)

Grammatical relations whose definition diverges from the one in the TIGER Treebank

Grammatical relations that can also be found in the TIGER Treebank, but whose definition diverges from the one there, are:

- *app* – close appositions, opposed to wide appositions in the TIGER Treebank; the latter are shifted to *mo*.
- *cc* – comparative (and equative) complements; in contrast to *CC* in the TIGER Treebank, this no longer comprises *wie*-PPs that are not triggered by an equative context; not being subcategorized, these are treated as *mos* in the TiGer DB.
- *gl* – genitive attribute on the left of its head noun *or* possessive determiner.

```

| pred 'machen'
| oa [9] | pred 'Regierung'
      | det | pred 'die'
| pd | pred 'mitverantwortlich'
      | op | pred 'für'
          | obj | pred 'Stimmung'
              | det | pred 'die'
                  | mo | pred 'ein#trüben'
| sb [9: Regierung]
| sb | pred 'DIHT'
      | det | pred 'die'

```

Figure 5.5: Predicate-argument triples of (5.5)

- *mo* – optional modifiers; in contrast to *MO* in the TIGER Treebank, *mo* in the TiGer DB no longer comprises (more or less) obligatory directional, local and modal arguments; the definition is enlarged with respect to the TIGER Treebank in that it now includes AMSs (measure complements of adjectives), APPs (wide appositions) and CCs (comparative complements) that are not triggered by a comparative or equative context.
- *pd* – all predicative arguments, not only those of the copular verbs *bleiben*, *sein* and *werden*.
- *rs* – reported speech, in sentences like (5.6), where the RS clause is not regularly subcategorized for by the matrix verb; note that all other RS constructions of the TIGER Treebank are reinterpreted as *oc_fins* (finite clausal objects, see 51).

- (5.6) Technisch sei dies machbar, widersprach Starzacher den Skeptikern
 Technically is this feasible, contradicted Starzacher the sceptics
 in der Verwaltung.
 in the administration.
 ‘Starzacher contradicted the skeptics in the administration, saying
 that, technically, this was feasible.’

‘New’ grammatical relations

Finally, there are a number of ‘new’ grammatical relations in the TiGer DB, which arise from the more fine-grained distinctions that are made in the dependency bank with respect to the TIGER Treebank edge labels.

- *app_cl* – appositive clauses, occurring with *es* and pronominal adverbs

5.3 Grammatical relations and features encoded in the TiGer DB

- `det` – articles, demonstrative and interrogative determiners
 - `measured` – measured entity in constructions such as (5.7)
- (5.7) `zwei Flaschen Wein`
`two bottles wine`
`‘two bottles of wine’`
- `mod` – non-head components of compounds (see, e.g., Figure 5.4 on p. 49)
 - `name_mod` – non-head in complex name
 - `number` – numbers in specifier position
 - `obj` – argument of a preposition or a subordinating conjunction
 - `oc_fin` – finite clausal objects (*dass/ob*-clauses, indirect wh-questions)
 - `oc_inf` – infinite clausal objects
 - `op_dir` – directional oblique arguments
 - `op_loc` – local oblique arguments
 - `op_manner` – modal oblique arguments
 - `quant` – quantifying specifiers

In general, determining these grammatical relations is relatively straightforward. There are exceptions to this rule, however, such as the distinction between `mos` (modifiers) and the different `ops` (prepositional arguments), as well as certain constructions where a given PP could be analyzed either as a `mo` or a `pd` (predicative argument).

5.3.3 Atomic features

The atomic features included in the TiGer DB correspond mostly to the morphological information encoded in the TIGER Treebank, but also to information from the part-of-speech tags. Furthermore, some of them encode the form of words that do not introduce a predicate themselves. Generally the atomic features further specify the predicates that relate to the terminal nodes in the TIGER Treebank where the information is encoded. An exception to this rule is the person/number agreement information given for finite verb forms, which ends up in the features `num` and `pers` of the subject of the verb under consideration, as well as agreement information provided by determiners and inflected adjectives, which is attached to their head noun. The purpose of this is to avoid the doubling of information.

It was decided to encode the following atomic features in the TiGer DB:

The TiGer Dependency Bank

- case with the values acc (accusative), dat (dative), gen (genitive) and nom (nominative)
- *compd_form* (see Subsection 5.3.1)
- *comp_form* (complementizer form) provided by *dass* or *ob*
- *coord_form* (form of coordinating conjunction) provided by *aber*, *oder*, *und* etc.
- degree with the values pos (positive), comp (comparative) and sup (superlative)
- *det_type* (type of determiner) with the values def (definite), demon (demonstrative), indef (indefinite) and int (interrogative)
- fut (future) with the value + (otherwise unspecified)
- *gend* (gender) with the values fem (feminine), masc (masculine) and neut (neuter)
- mood with the values imp (imperative), ind (indicative) and subj (subjunctive)
- num (number) with the values sg (singular) and pl (plural)
- *pass_asp* (dynamic vs. stative passive) with values *dynamic* and *stative*
- perf (perfect) with the value + (otherwise unspecified)
- pers (person) with values 1, 2 and 3
- *precoord_form* (first part of composite coordinating conjunction) provided by *entweder*, *sowohl* etc.
- *pron_form* (form of indefinite pronoun) with values like *etwas*, *jemand*, *man*, *nichts* etc.
- *pron_type* (type of pronoun) with the values demon (demonstrative), indef (indefinite), pers (personal), recip (reciprocal) and refl (reflexive)
- tense with the values pres (present) and past

5.4 Uses of the TiGer DB

The TiGer DB is designed as a gold standard for the dependency-based evaluation of German parsing systems. Since it uses the fairly theory-independent dependency structures (even though they are labelled), we expect them to be of use for a number of linguistic theories and, hence, to allow cross-framework comparisons. Concerning the concrete possibilities of matching parser output against the TiGer DB dependency triples, we have considered this task both for an LFG and for an HPSG parser.

Given the resemblance between the TiGer DB representations and f-structures, the mapping from LFG f-structures to dependency triples is straight-forward. It involves some renaming and reorganizing of the structures, but this is basically the same as in the final step of the construction of the TiGer DB (see Subsection 5.2.4).

The mapping from HPSG feature structures to the TiGer DB is less trivial, since the representations differ more. Nevertheless, the TiGer DB was constructed in close collaboration with the HPSG developers at Saarland University, so that it should, at least in principle, be a useful resource for the HPSG community as well, although a direct mapping from the (R)MRSs produced by the German HPSG developed in Saarbrücken seems to be difficult. Spreyer & Frank (2005a,b) therefore investigated ways of converting the TiGer DB structures consisting of dependency triples into RMRs by applying a further transfer step to them.

Due to its relatively limited size, the TiGer DB has so far only been used for evaluation purposes and, at least in as far as our grammar development activities are concerned, it will continue to be considered evaluation data and thus not be inspected during grammar development.

In order to enable us to use TIGER-derived dependency annotations for corpus sections other than sentence # 8,001 to sentence # 10,000 in regression tests during grammar development, a fully automatic conversion of TIGER Treebank graphs into dependency triples has recently been developed (Kountz 2006). The grammatical relations used in these new representations correspond directly to TIGER Treebank edge labels, so that the representations are more coarse-grained than the TiGer DB structures. However, they are available for almost the whole TIGER Corpus, and thanks to a recent extension of the XLE output representations, we will be able to map the f-structures produced by the German *ParGram* LFG onto this new type of dependency structure. For the first time, we will then be able to closely observe the effects of grammar modifications on the accuracy of the grammar's analyses during grammar development. Final evaluations will be carried out both on the more fine-grained TiGer DB and on the new dependency structures.

5.5 Summary

The TiGer Dependency Bank is a dependency bank containing both grammatical relations between predicates and arguments and a number of other grammatical features. In this, it is closely related to the PARC 700 Dependency Bank (King et al. 2003). It has been produced semi-automatically on the basis of the TIGER Treebank annotations, partly cross-validated by means of the German *ParGram* LFG and finally validated by human annotators. The automation and cross-validation allow for a rapid construction of the TiGer Dependency Bank compared to the creation of such a resource from scratch. Nevertheless, the manual effort involved in resolving ambiguities introduced during lemmatization and the reinterpretation of ambiguous TIGER edge labels was considerable.

The TiGer DB is intended to be used for the evaluation of German parsers. As the grammatical relations encoded in it are close to the ones in the TIGER Treebank and the related NeGra Treebank, we hope that, apart from the German *ParGram* LFG, all kinds of parsing systems for German, both hand-crafted ones and systems that were induced from the treebanks mentioned, will be evaluated on it. In order to facilitate this, this chapter has given a relatively detailed account of how the TiGer DB structures relate to the corresponding TIGER Treebank graphs.

Part III

OT-inspired Disambiguation

Chapter 6

Manually Defined OT Constraint Rankings for Disambiguation

This chapter discusses the Optimality-Theoretically inspired disambiguation mechanism implemented in XLE in detail and the way this mechanism is used in the initial German *ParGram* LFG. The functionality of various categories of optimality marks is explained, and all the marks used in the original grammar are presented.

6.1 Optimality-Theoretically inspired disambiguation in XLE

Theory-driven grammar development typically leads to grammars that overgenerate only mildly, since lexical subcategorization information is taken into account and the grammatical constructions can be restricted by rich feature constraints. In other words, most of the parses that a grammar assigns to a string are linguistically justified. Nevertheless, due to the underspecified nature of natural language and due to the fact that broad-coverage grammars necessarily include rules for rather rare phenomena that are then — erroneously — used in analyzing sentences that have nothing to do with these phenomena, ambiguity rates for non-trivial sentences are considerable. This justifies the need for disambiguation techniques to complement symbolic broad-coverage grammars in parsing.

Motivated by the observation that languages exhibit (dis)preferences or tendencies for or against certain constructions and interpretations and that grammar writers are often well aware of these (dis)preferences when working on a linguistic construction, the XLE grammar development platform, which processes the German *ParGram* LFG as well as all other *ParGram* LFGs, has been integrated with a soft constraint mechanism inspired by the constraint ranking

system of Optimality Theory (OT) (Frank et al. 2001). It makes it possible, for instance, that “rare” constructions are only appealed to in analysis when there is no “canonical” analysis of a string. Awareness of this type of linguistic interaction goes back to Panini, and in recent years, ways of including soft-constraint mechanisms in formal grammar formalisms have been explored, particularly in the framework of Optimality Theory (Prince & Smolensky 1993) or in probabilistic grammar models (Manning 2003).

The mechanism is conceptually quite simple: For particular structural configurations in the linguistic representation, an optimality mark (OT mark) can be introduced (e.g., for the occurrence of a topicalized object). The introduction of OT marks is realized via the projection of such a mark to a separate representation (in addition to the c-structure and the f-structure), the so-called o-structure, which is basically just a set of sets of OT marks. The constraints that project OT marks to the o-structure have the following form:¹

```
... Mark1 $ o::* ...
```

Each optimality mark is assigned a polarity, so that the corresponding structure is defined as preferred or dispreferred. Preference OT marks are marked by a ‘+’; dispreference OT marks are the default case. Furthermore, all marks used in a grammar are ordered in a relative ranking, but groups of several marks can also be given the same rank position with the help of parentheses. This ranking is specified in the `config` section of the grammar as follows:

```
FOO GERMAN CONFIG (1.0)
...
OPTIMALITYORDER Mark1 ( Mark2 +Mark3 ) Mark4 +Mark5.
...
```

When the parser is applied, OT mark instances are collected from the o-structure and can then be used as a filter on the readings produced by the system. Following the ranking order, each mark filters out readings that have fewer instances than the reading with the maximal number of instances (for preference marks) or more instances than the reading with the fewest instances (for dispreference marks). The readings of a sentence that pass all marks and are still left in the end are called “optimal”, the readings that are filtered out are called “suboptimal”.

For example, the parser may assign four readings to a sentence. Reading 1 has the multiset { *Mark2*, *Mark2*, *Mark3*, *Mark3* }, Reading 2 has the multiset { *Mark1*, *Mark2*, *Mark3* }, Reading 3 { *Mark2*, *Mark2*, *Mark3*, *Mark4* }, and Reading 4 { *Mark2*, *Mark2*, *Mark3*, *Mark3*,

¹\$ stands for \in and `o::*` stands for the o-structure projected from the annotated c-structure node.

Mark5 }. Let us furthermore assume that the marks are ranked and defined as preference vs. dispreference marks according to the sample OPTIMALITYORDER given above. In evaluation, *Mark1* is considered first. Reading 2 has one occurrence of *Mark1*, whereas all the other readings have none. Therefore Reading 2 is filtered out at this step. For the remaining readings, *Mark2* and *Mark3* are considered next. They all contain the same number of occurrences of *Mark2*, but Readings 1 and 4 have more occurrences of *Mark3* than Reading 3, so that the latter is filtered out. Finally, Reading 4 has one occurrence of the preference mark *Mark5*, whereas Reading 1 has none. Hence, the optimal reading is Reading 4.

Apart from ‘general’ OT marks, which are used for ranking analyses after parsing proper in the way just presented, the OPTIMALITYORDER can comprise other types of OT marks that actually interact with the parse process. These are mainly the so-called NOGOOD OT marks and STOPPOINT OT marks. How exactly they interact with the parse process is presented in the following section with the help of concrete examples. Furthermore, XLE now allows OT marks to be specified as CSTRUCTURE OT marks, which means that they are evaluated after the construction of all possible c-structures and, hence, before the resolution of f-annotations, and OT marks can be defined as ‘local’ rather than ‘global’. Since these options were not yet available for the original version of the German *ParGram* LFG, since they are only marginally used in the final system and since they cannot be ranked and evaluated according to the method proposed in Chapter 7, we do not present these here. The interested reader is referred to Crouch et al. (2006).

6.2 Optimality marks in the original German grammar

In the original German grammar, this OT-inspired mechanism is the only mechanism for disambiguation. Therefore, 65 OT marks, which is more than in the revised grammar, are used there and they are organized in a relatively elaborate hierarchy. In the following, I present the marks used in the original German grammar and explain which kinds of ambiguities they are supposed to solve. For most OT marks, I provide examples from the TIGER Corpus that can be correctly disambiguated with their help. Working with authentic data seems important to me, as all approaches presented subsequently are data-driven and, hence, rely on corpus data.

(6.3) [...] dürften sich Christen nicht davon abhalten lassen.
 [...] should themselves Christians not from this refrain let.
 ‘[...] should Christians not let themselves be refrained from this.’⁴

- `FUWithArgs` is projected in the `DPfunc`, `PPfunc`, `ADVPfunc` and `PREDPfunc` templates in `vp.tmp1.lfg`. It comments out the functional uncertainty path `XCOMP*` for arguments.
- `NoGood` is projected in the `NACHFELD-SIMPLE` macro in `cp.gram.lfg`. It comments out *zu* infinitives in the Nachfeld that are not separated by a comma, such as *zu brennen* in (6.4).

(6.4) [...], wann es anfängt zu brennen.
 [...] when it starts to burn.
 ‘[...] when it will start to burn.’⁵

6.2.2 STOPPOINT OT marks

8 out of the 65 OT marks are so-called STOPPOINT OT marks. These OT marks are special in that they are not used for ranking analyses after the parsing process proper, but that they interact with the parsing process. In fact, they mark (parts of) grammar rules and lexical entries that are believed to cover rather infrequent constructions that are expensive to process. The idea then is not to consider these (parts of) rules and lexical entries in a first attempt to parse a sentence, but to include them only once the attempt with the reduced and hence more efficient grammar version has failed. It is possible to use more than one STOPPOINT marker in the OT mark hierarchy, which causes XLE to make, in addition to the second attempt caused by the lowest STOPPOINT marker, a third, fourth or even fifth attempt to parse a given sentence, each time with a more relaxed grammar version. In practice, however, this possibility is rarely made use of, and in the original grammar, there is only one STOPPOINT marker in the OT mark hierarchy. Since all STOPPOINT OT marks are ranked equally high, I present them in alphabetical order:

- `CPCommaCoord` is projected in the `PUNCT-COORD` macro in `cp.gram.lfg`. It marks coordinations of matrix sentences where, instead of a coordinating conjunction, there is only a comma that “coordinates” the two sentences. `CPCommaCoord` is a STOPPOINT OT mark because the disjunct of the `ROOT` rule which it marks has a serious impact on efficiency. Nevertheless, comma-separated coordinations of sentences are too frequent not

⁴s6008

⁵s1832

to cover them at all. (6.5), whose c-structure is given in Figure 6.1, is an example of such a coordination.

- (6.5) Am Tag schlage das Herz schneller, der Blutdruck sei höher.
 At the day beat the heart faster, the blood pressure was higher.
 higher.
 ‘During the day, the heart beat faster, blood pressure was higher.’⁶

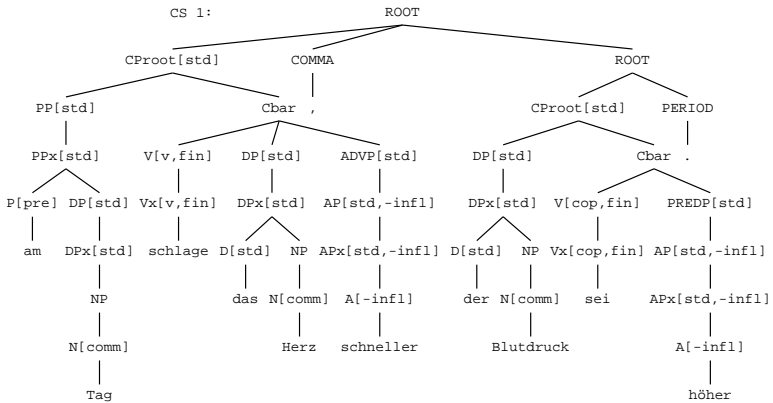


Figure 6.1: C-structure built for (6.5) in second parsing attempt

- EinAsCard is projected in the lexical entry of *eine_card* in *dp.lex.lfg*. It marks the use of the forms *ein*, *eine* etc. as a cardinal number, as it occurs in the c-structure given in Figure 6.2. EinAsCard is a STOPPOINT OT mark because these forms are identical to the very frequent indefinite article.

- (6.6) einen Tag Urlaub
 one day vacation
 ‘one day off’⁷

- EmptyHead is projected in the NP rule in *np.gram.lfg*. It marks NPs that lack a nominal head, such as *ein kurzfristiger* in (6.7). The c-structure corresponding to this kind of headless NP is illustrated in Figure 6.3.

⁶s814

⁷s1759

6.2 Optimality marks in the original German grammar

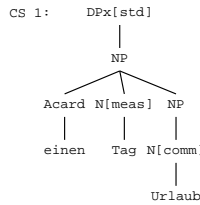


Figure 6.2: C-structure built for (6.6) in second parsing attempt

- (6.7) Ist das ein langfristiger Prozess oder ein kurzfristiger?
 Is this a long term process or a short term?
 ‘Is this a long term or a short term process?’⁸

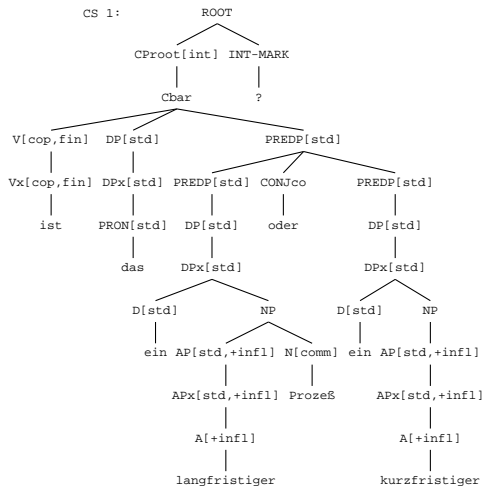


Figure 6.3: C-structure built for (6.7) in second parsing attempt

- ParenthInMittelfeld is projected in the VP-PARENTHETICAL macro in `vp.gram.lfg`. It marks the attachment of parentheticals, which include comma-separated adverbial clauses like the one in (6.8) in this grammar version, to the Cbar or the VP that represents the Mittelfeld. How exactly the parenthetical is attached to the Mittelfeld is illustrated in Figure 6.4.

⁸s166

- (6.8) Es geht, wenn wir von Erfolg in der Politik reden, um
 It goes, when we of succes in the politics talk, around
 Kompetenzzuschreibung.
 competence attribution.
 ‘When we talk about success in politics, it is actually about the attri-
 bution of competences.’⁹

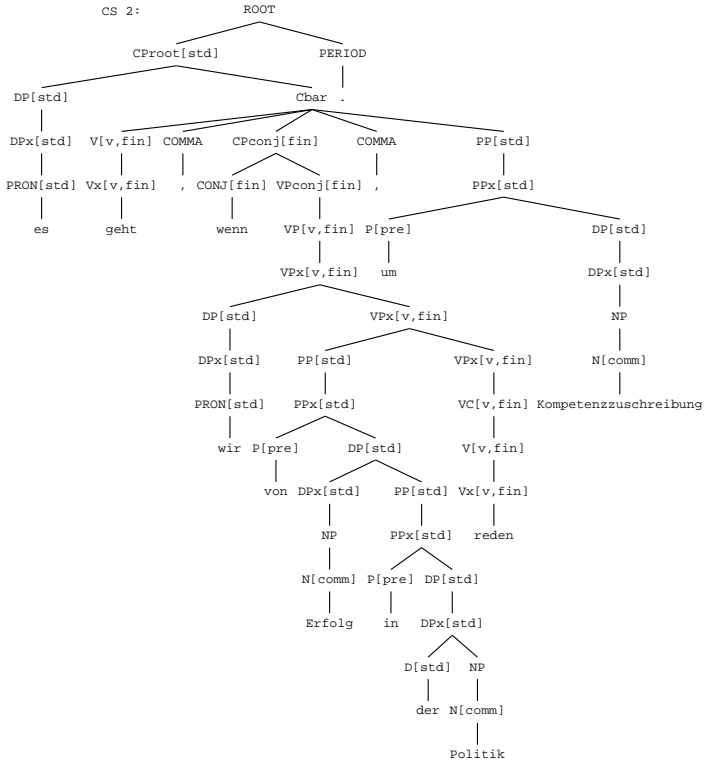


Figure 6.4: C-structure built for (6.8) in second parsing attempt

- PersAppos is projected in the PRON-APPOS macro in `dp.gram.lfg`. It marks close appositions to personal pronouns, such as *drei Frauen* in (6.9). The corresponding c-structure is given in Figure 6.5.

⁹s17847

6.2 Optimality marks in the original German grammar

(6.9) wir drei Frauen
we three women
'we three women'¹⁰

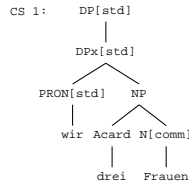


Figure 6.5: C-structure built for (6.9) in second parsing attempt

- PunctInImp is projected in the PUNCT-IMP-V1 and PUNCT-IMP-VLAST macros in cp.gram.lfg. In the former, PunctInImp marks the use of a period or a colon after a verb-initial imperative sentence, such as (6.10); in the latter, it marks the use of an exclamation mark or a colon after a verb-final (i.e. infinitival) imperative sentence, such as (6.11). This OT mark was probably introduced with generation in mind; although we want to obtain the analyses in Figures 6.6 and 6.7 when parsing, we would not want to generate these c-structures from the corresponding f-structures.

(6.10) Laßt Honecker in Frieden. (6.11) Bloß jetzt nicht schlappmachen!
Leave Honecker in peace. Just now not break down!
'Leave Honecker alone.'¹¹ Just do not break down now!¹²

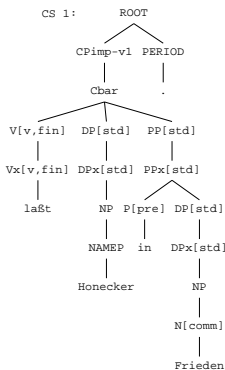


Figure 6.6: C-structure built for (6.10) in second parsing attempt

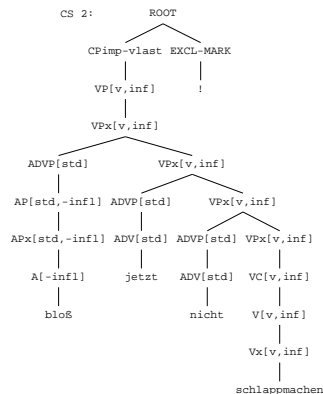


Figure 6.7: C-structure built for (6.11) in second parsing attempt

¹⁰s4981

- SentArgInAP is projected in the APconst-POST macro in ap.gram.lfg. It marks the occurrence of clausal and infinitival constituents in the Nachfeld of an AP, as illustrated in Figure 6.8.

(6.12) unfähig, sich selbst und die Wähler zu mobilisieren,
 unable themselves themselves and the voters to mobilize
 ‘unable to mobilize themselves and the voters’¹³

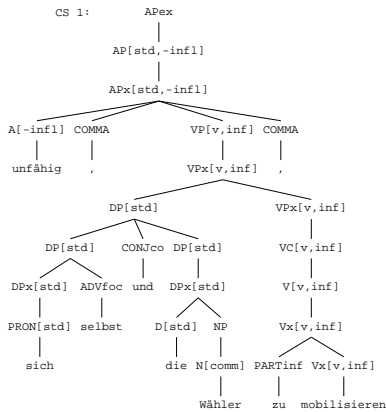


Figure 6.8: C-structure built for (6.12) in second parsing attempt

- VPTopicalization is projected in the disjunct of the CProot rule in cp.gram.lfg that covers sentences with a partially fronted VP, such as (6.13). The corresponding c-structure is given in Figure 6.9.

(6.13) Vorbereitet werden effektivere Mittel der Gewalt.
 Prepared are more efficient means of force.
 ‘What is prepared are more efficient means of force.’¹⁴

6.2.3 General OT marks

The remaining 51 OT marks of the original grammar are ‘normal’ OT marks in the sense that they are used for ranking analyses after the parsing process

¹³s22275

¹⁴s395

6.2 Optimality marks in the original German grammar

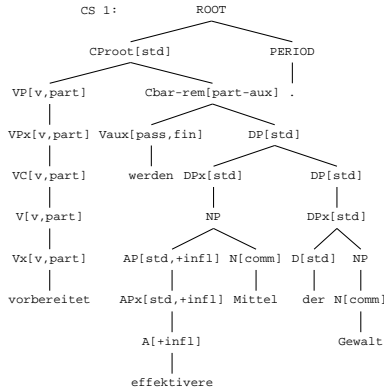


Figure 6.9: C-structure built for (6.13) in second parsing attempt

proper; they thus do not interact with the parsing process. These marks are organized in 16 equivalence classes, i.e. groups of OT marks that are ranked equally high. I present the equivalence classes in descending order, ordering the OT marks within each equivalence class alphabetically:

- The equivalence class with the highest rank among the general OT marks comprises *GussedMassNoun*, *MarkedPunct* and *VlastImp*.
 - *GussedMassNoun* is a dispreference mark that serves mainly robustness purposes. It is projected in the template *DEFAULT-COUNT-NOUN* in *np.tmpl.lfg* and marks the mass noun reading of a noun that is not explicitly listed as a mass noun or a count noun in the grammar's lexicon.

As an example, take the noun *Story* in (6.14). It is listed neither as a count noun nor as a mass noun in the grammar's lexicon of noun stems; it can thus be guessed as both a count and a mass noun, which gives rise to the two competing analyses illustrated in Figure 6.10 (p. 68), the latter option being dispreferred by this OT mark.

(6.14) Die Story geht so.
 The story goes so.
 'The story goes like this.'¹⁵
 - *MarkedPunct* is a dispreference mark that marks a question mark or an exclamation mark after a declarative sentence. It is projected in

¹⁵s23

Manually Defined OT Constraint Rankings for Disambiguation

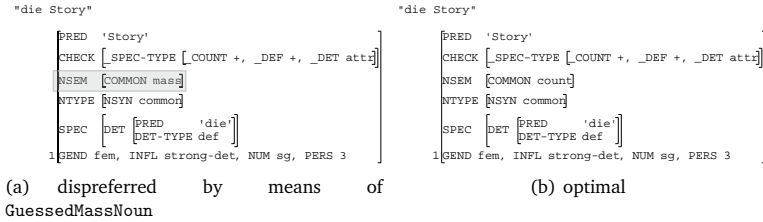


Figure 6.10: F-structures of mass noun and count noun readings of *Story* in (6.14)

the macro PUNCT-STD in `cp.gram.lfg`. Similarly to `PunctInImp`, this OT mark was probably introduced with generation in mind; although we want to parse the question marks and exclamation marks that sometimes occur after declarative sentences, we usually do not want to generate them.

- `VlastImp` is a dispreference mark which is projected in the `ROOT` rule in `cp.gram.lfg`. It disprefers the interpretation of a sentence as a verb-final, i.e. infinitival, imperative in cases where alternative analyses are available. An example of a sentence for which both a declarative and an imperative reading are available is (6.15). For this sentence, `VlastImp` correctly treats the imperative reading (see Figure 6.11(a)) as suboptimal with respect to the declarative reading (see Figure 6.11(b)).

(6.15) Kommunalpolitiker fragen nach der Möglichkeit von
Municipal politicians ask after the possibility of
Koalitionsbildungen [...].
coalition formations [...].
'Municipal politicians ask/Ask municipal politicians about the
possibility of coalition formations [...].'¹⁶

- The next-highest equivalence class comprises `ComplexWord` and `DerivedWord`.
 - `ComplexWord` is projected in the `COMPOUND` macro in `misc.gram.lfg`. For each compound non-head which is covered by this macro, one

¹⁶s162. When the competing analyses have different meanings that can be rendered in English at least approximately, we provide translations for both of them. In order not to confront the reader with implausible sentences directly, we give the the translation of the intended reading first and then a translation of the competing reading(s). In the corresponding structures, however, we provide the suboptimal reading(s) first and show the intended reading last.

6.2 Optimality marks in the original German grammar

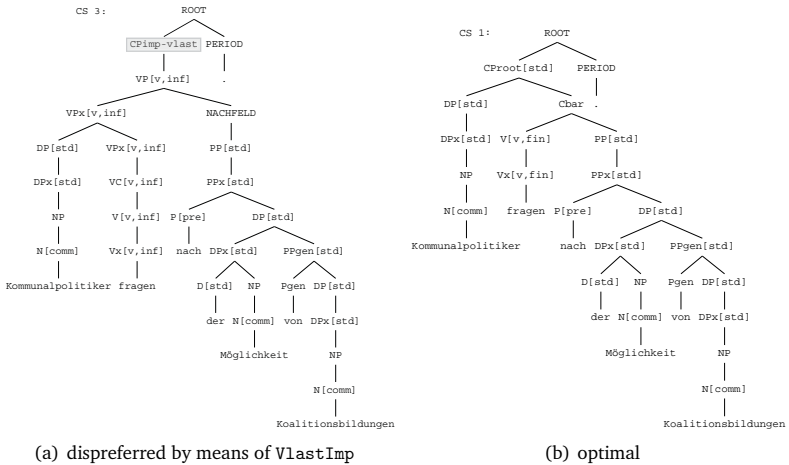


Figure 6.11: C-structures of imperative and declarative readings of (6.15)

occurrence of ComplexWord is projected. As a consequence, morphological analyses of compounds involving many components are dispreferred with respect to analyses involving fewer components. For (6.16), the simplex analysis of *gleichzeitig*, illustrated in Figure 6.12(b), is preferred over the competing compound analysis, illustrated in Figure 6.12(a) (both p. 70).

- (6.16) Gleichzeitig wurde über eine neue Verfassung
 Simultaneously was about a new constitution
 abgestimmt.
 voted.
 ‘Simultaneously, a referendum about a new constitution was held.’¹⁷

- DerivedWord is projected in the lexical entry of [^]VINF in *dmor1.lex.lfg*. It disprefers the analysis of nouns as nominalized infinitives with respect to alternative analyses. Figure 6.13 (p. 70) shows the competing analyses of the noun *Bündnispflichten* in (6.17).

¹⁷s19722

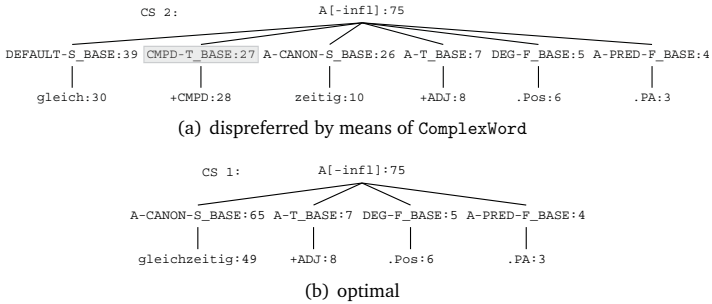


Figure 6.12: Competing c-structures for *gleichzeitig* in (6.16)

(6.17) ausgenommen Landesverteidigung und Bündnispflichten
 except country defense and alliance obligations
 ‘except homeland defense and alliance obligations’¹⁸

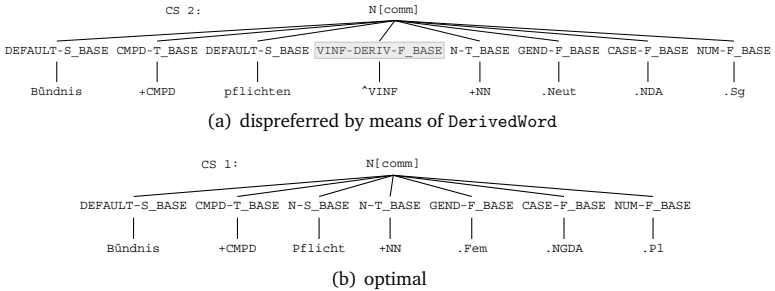


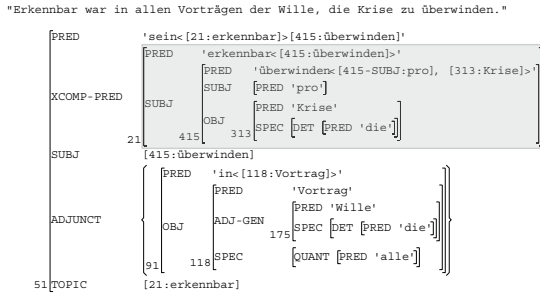
Figure 6.13: Competing c-structures for *Bündnispflichten* in (6.17)

- The third equivalence class in the hierarchy of the general OT marks comprises AdjSententialSubj, DPAppositive, LabelP and PersWithPP.
 - AdjSententialSubj is projected in the ADJ template in `ap_tmpl.lfg`. It marks readings where an adjective takes a sentential or infinitival subject and is intended to disprefer them when a reading is available where the adjective takes a nominal subject. An example sentence where these kinds of readings are both available is given in (6.18) and the competing analyses are shown in Figure 6.14.

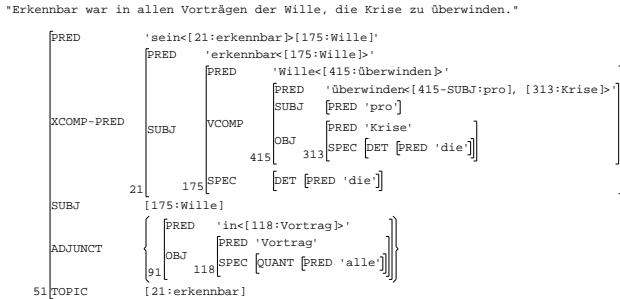
¹⁸s532

6.2 Optimality marks in the original German grammar

- (6.18) Erkennbar war in allen Vorträgen der Wille, die Krise
 Recognizable was in all presentations the will/Wille the crisis
 zu überwinden.
 to overcome.
 'The will to overcome the crisis was recognizable in all presenta-
 tions./To overcome the crisis was recognizable in all presenta-
 tions of Wille.'¹⁹



(a) dispreferred by means of AdjSententialSubj



(b) optimal

Figure 6.14: Competing c-structures for (6.18)

- DPAppositive is projected in the DP-APPOS-HEAD macro in `dp.gram.lfg`. It marks so-called wide appositions, which occur at the right edge of DPs and which are DPs in their own right. It is intended to disprefer the interpretation of DPs as wide appositions with respect to their interpretation as, e.g., conjuncts in a coordination. Figure 6.15 (p. 72) illustrates competing analyses of this kind.

¹⁹s25374

(6.19) Und Coca Cola, IBM, Siemens oder BMW sind keine Schimpf-
 And Coca Cola, IBM, Siemens or BMW are no insult
 Bezeichnungen mehr.
 designations more.
 'And Coca Cola, IBM, Siemens or BMW are no insults any
 longer.'²⁰

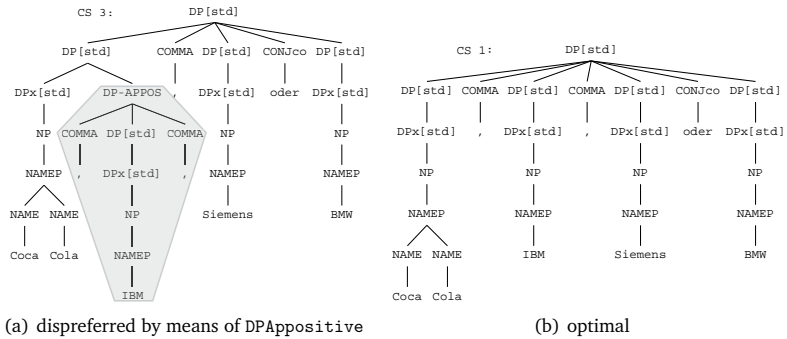


Figure 6.15: Competing c-structures for *Coca Cola, IBM, Siemens oder BMW* in (6.19)

- LabelP is projected in the N-APPOS-LABEL macro in `np.gram.lfg`. It marks so-called close appositions and is intended to disprefer them with respect to alternative interpretations. In (6.20), it disprefers the interpretation of *Welayatis* as a close apposition, illustrated in Figure 6.16(a) with respect to the correct interpretation as a genitive attribute, illustrated in Figure 6.16(b).

(6.20) Er habe gefordert, an der Ausladung
 He had asked to the cancellation of the invitation
 Welayatis festzuhalten [...]
 Welayati-GEN/Welayatis adhere [...]
 'He had asked for adherence to the cancellation of the invitation
 [of Welayati]/Welayatis.'²¹

- PersWithPP is projected in the DPpost-PP template in `dp.tmpl.lfg`. It marks postnominal PPs in DPs whose head is a personal pronoun.

²⁰s65
²¹s20904

6.2 Optimality marks in the original German grammar

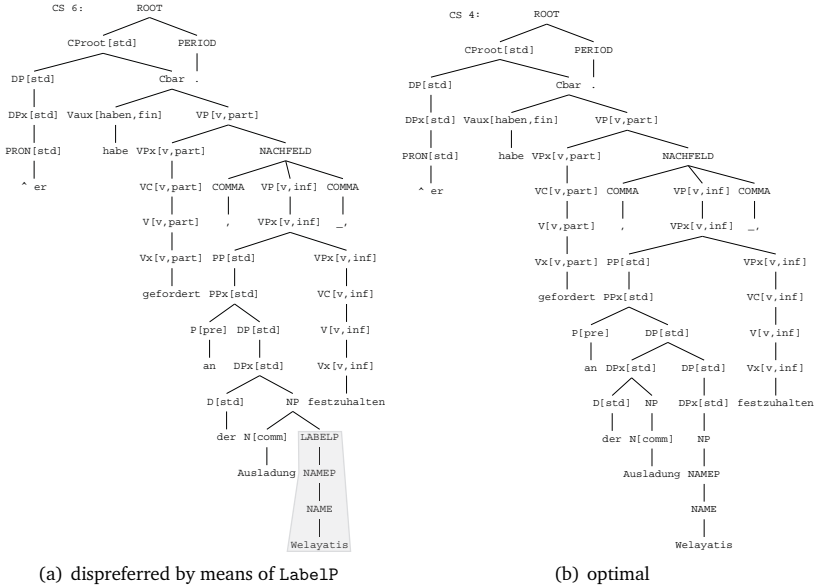


Figure 6.16: Competing c-structures for (6.20)

It is intended to resolve PP attachment ambiguities by means of the heuristic that personal pronouns are rarely modified by PPs. In (6.21), it correctly disprefers the attachment of the PP *am Freitag* to the DP headed by *er*, illustrated in Figure 6.17(a), favoring thus its attachment to the clause level, illustrated in Figure 6.17(b) (both p. 74).

- (6.21) Spekulationen [...] wies er am Freitag zurück.
 Speculations [...] rejected he on the Friday.
 'He rejected speculations [...] on Friday./He on Friday rejected speculations [...]'²²

- NominalizedAP is projected in the ADJ-DEVERB-NOMINALIZED and ADJ-NOMINALIZED macros in *ap.gram.lfg*. It marks the analysis of nominalized adjectives and participles via the AP rule, illustrated in Figure 6.18(a), thus dispreferring this analysis with respect to the analysis via the NAdj rule, illustrated in Figure 6.18(b) (both p. 75).

²²s19370

Manually Defined OT Constraint Rankings for Disambiguation

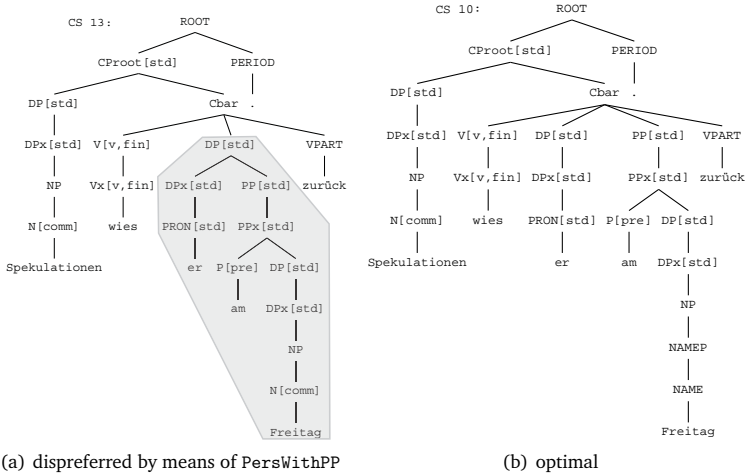


Figure 6.17: Competing c-structures for (6.21)

(6.22) Eine solche Liberalisierung lehnen die Grünen ab.
 A such liberalization disapprove the Greens of.
 ‘The Greens disapprove of such a liberalization.’²³

- The fifth equivalence class comprises the OT marks *MitAsAdv*, *Ppost*, *RelAdv*, *SoAsDet*, *WasAsQuant*, *WelcheAsQuant*, *WelcheAsRelPron* and *WieAsCONJsub*.
 - *MitAsAdv* is projected in the lexical entry of *mit* as an ADV-S in *advp.lex.lfg*. It disprefers the adverbial use of *mit*, making it sub-optimal with respect to its potential use as a preposition or a verb particle. It thus correctly prefers the analysis in Figure 6.19(b) over the analysis in Figure 6.19(a) (both p. 76).

²³s23131

6.2 Optimality marks in the original German grammar

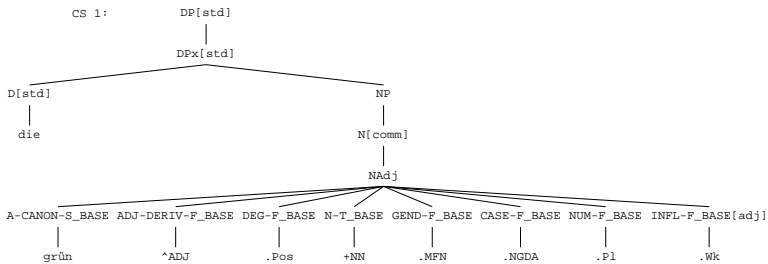
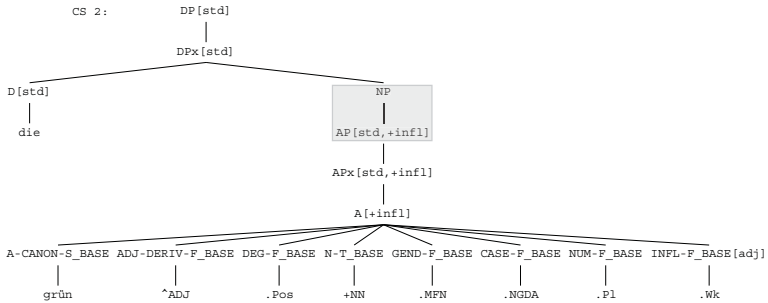


Figure 6.18: Competing c-structures for *die Grünen* in (6.22)

(6.23) Stattdessen werden mit dem Januar-Entgelt [...] 1000
 Instead are with the January allowance [...] 1,000
 Mark ausgezahlt [...]
 marks paid [...]
 'Instead, 1,000 marks are paid with the January allowance [...]/
 Instead, 1,000 marks are paid among others to the January al-
 lowance [...]'²⁴

- Ppost is projected in the lexical entry of *+POSTP* in *dmor1.lex.lfg*. It disprefers analyses containing postpositions with respect to alternative analyses where the potential postpositions are analyzed as, e.g., prepositions or verb particles. Figure 6.20 (p. 77) shows that it disprefers the interpretation of *über* in (6.24) as a postposition with respect to its interpretation as a preposition.

²⁴s24259

Manually Defined OT Constraint Rankings for Disambiguation

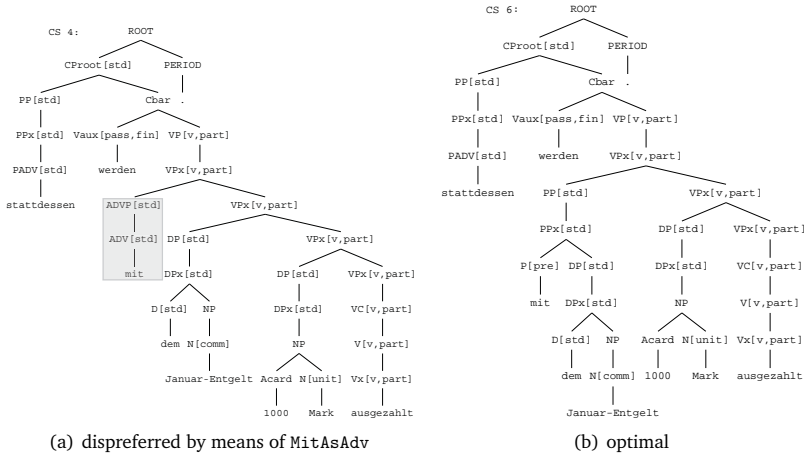


Figure 6.19: Competing c-structures for (6.23)

(6.24) Dieser übernimmt die alleinige Kontrolle über den Konzern.
 This one takes over the sole control over the company.
 ‘This one takes over the sole control of the company/takes over the company during the sole control.’²⁵

- RelAdv is projected in the lexical entry of +RELADV in dmor1.lex.lfg. It marks analyses containing relative adverbs and is intended to disprefer these analyses with respect to alternative analyses, containing, e.g., interrogative adverbs instead. In (6.25), it correctly disprefers the interpretation of the clause introduced by *wie* as a relative clause, making the interpretation as an interrogative argument clause optimal. Figure 6.21 (p. 78) illustrates these competing analyses.

(6.25) die Frage, wie Arbeitsplätze zu schaffen seien,
 the question how jobs to create are
 ‘the question of how jobs are to be created/the question as jobs are to be created’²⁶

²⁵s22726

²⁶s12833

6.2 Optimality marks in the original German grammar

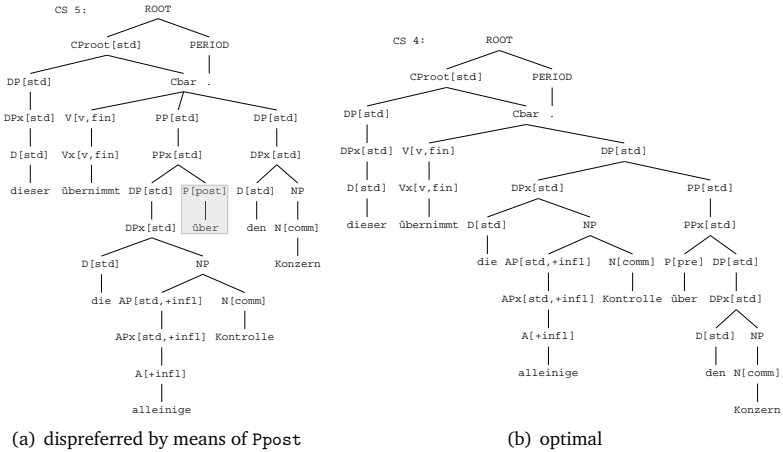


Figure 6.20: Competing c-structures for (6.24)

- SoAsDet is projected in the lexical entry of *so* as an INDEF-S in *dp.lex.lfg*. It disprefers the analysis of *so* as a determiner, illustrated in Figure 6.22(b) (p. 78), with respect to alternative analyses.

(6.26) Er hat so Leute kennengelernt, die [...]

He has so people got to know who [...]

'This way, he has got to know people who [...] / He has got to know people like this, who [...]

- WasAsQuant is projected in the lexical entry of *was.indef* in *dp.lex.lfg*. It disprefers the interpretation of *was* as the shortened version of the indefinite pronoun *etwas*, illustrated in Figure 6.23(b) (p. 79). If alternative interpretations, e.g. as an interrogative pronoun, are available, these are favored.

(6.27) [...], daß die Republikaner nicht sahen, was

[...] that the Republicans not saw what/something geschah.

happened.

'[...] that the Republicans did not see [what happened] / [something happened].'²⁷

²⁷s30024

Manually Defined OT Constraint Rankings for Disambiguation

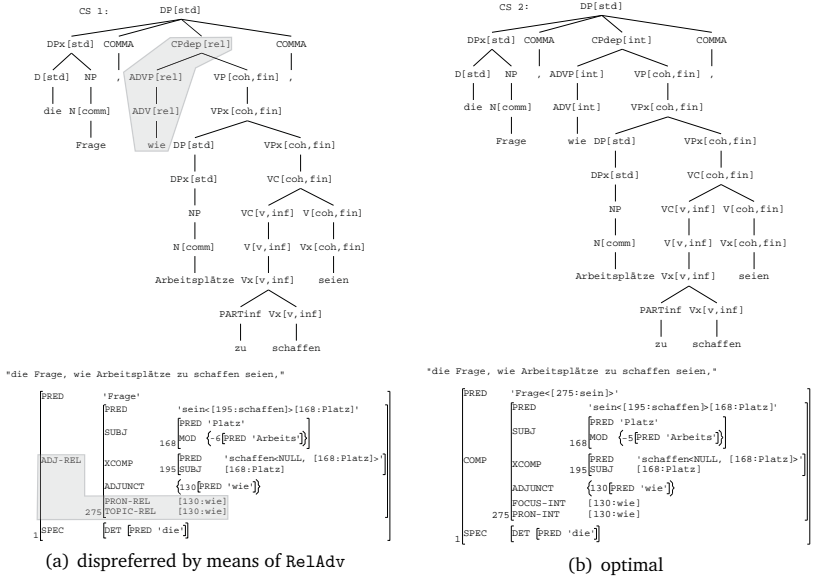


Figure 6.21: Competing c- and f-structures for (6.25)

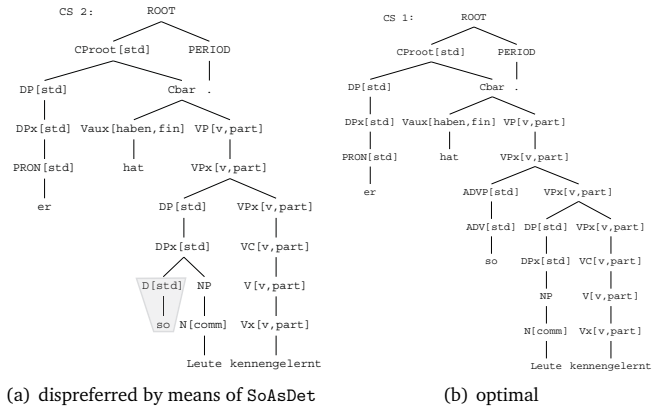


Figure 6.22: Competing c-structures for (6.26)

6.2 Optimality marks in the original German grammar

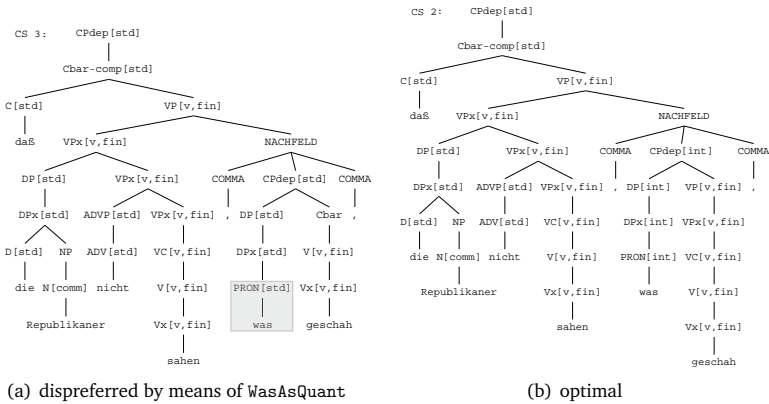


Figure 6.23: Competing c-structures for (6.27)

- *WelcheAsQuant* is projected in the lexical entry of *welche_indef* in *dp.lex.lfg*. It marks the use of *welche* as an indefinite pronoun as in (6.28), potentially dispreferring it to alternative interpretations.

(6.28) Außerdem gebe es welche, die noch viel schlimmer dran
 Moreover gives it some who even much worse at it
 seien [...] were
 'Moreover, there are some who are in a much worse situation.'²⁸
- *WelcheAsRelPron* is projected in the lexical entry of *welche_rel* in *dp.lex.lfg*. It marks the use of *welche* as a relative pronoun as in (6.29), potentially dispreferring it to alternative interpretations.

(6.29) Das Ehepaar Bonk, welches sich seit sechs Jahren
 The married couple Bonk, who themselves for six years
 Kinder wünschte, [...] children wished, [...]
 'The Bonks, who had had the desire to have children for six
 years, [...]'²⁹
- *WieAsCONJsub* is projected in the lexical entry of *wie* as a *CONJsub-S* in *cp.lex.lfg*. It marks the use of *wie* as a subordinating conjunction and disprefers its interpretation as such a conjunction, illustrated in Figure 6.24(a) (p. 81), with respect to alternative analyses.

²⁸s43687

²⁹s15333

(6.30) Hier herrscht Unklarheit, wie stark einzelne Branchen
Here reigns vagueness how strongly individual sectors
betroffen sein werden.
concerned be will.
'There is vagueness [as to how strongly individual sectors will be
concerned]/[as individual sectors will be concerned strongly].'³⁰

- The sixth equivalence class comprises the OT marks *OldDat* and *WeakGen*.

- *OldDat* is projected in the lexical entry of *^OLDDAT* in *dmor1.lex.lfg*. It marks slightly archaic dative forms ending in *e*, which are dispreferred with respect to alternative readings of the corresponding forms. For (6.31), *OldDat* correctly leads to the preference of the interpretation of the form *Reise* as the dative of *Reise* over its interpretation as the dative of *Reis*. The two competing analyses are illustrated in Figure 6.25 (p. 82).

(6.31) auf Asien-Reise
on Asia trip/rice
'on a trip to Asia/on Asia rice'³¹

- *WeakGen* is projected in the lexical entry of *^WEAKGEN* in *dmor1.lex.lfg*. It is used in a very way similar to *OldDat* and marks so-called 'weak' genitive forms as in (6.32). As these forms generally do not lead to ambiguities, unlike *OldDat*, this is an OT mark which is useful mainly for surface realization, where it can prevent the simultaneous generation of both weak and strong forms.

(6.32) zur Umsetzung solchen gewerkschaftlichen Wollens
to the realization such-WEAKGEN of the unions wanting
'to the realization of such a will on the unions' side'³²

- The seventh equivalence class comprises the OT marks *DefinitePREDP* and *PPdirAsPREDP*. The purpose of these OT marks is mainly the resolution of *SUBJ* vs. *XCOMP-PRED* ambiguities.

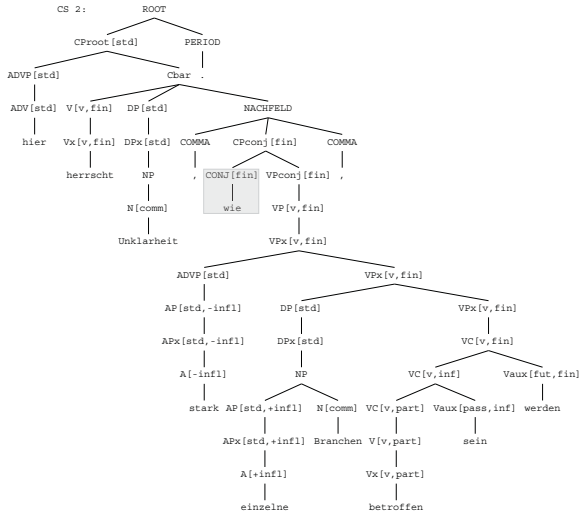
- *DefinitePREDP* is projected in the *PREDP-DP-RESTR* template in *vp.tmp1.lfg*. It disprefers the interpretation of definite DPs as *PREDPs*. For (6.33), it correctly attributes the grammatical function *XCOMP-PRED* to the sentence-initial DP, and the function *SUBJ* to the subsequent one. This analysis as well as the competing interpretation are illustrated in Figure 6.26 (p. 83).

³⁰s20203

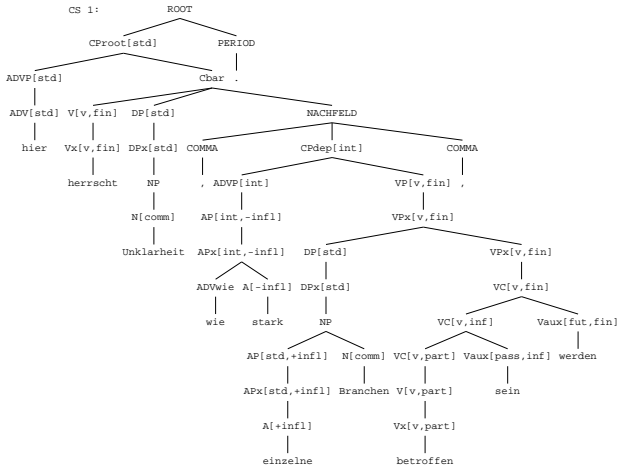
³¹s23083

³²s21987

6.2 Optimality marks in the original German grammar



(a) dispreferred by means of *WieAsCONJsub*



(b) optimal

Figure 6.24: Competing c-structures for (6.30)

Manually Defined OT Constraint Rankings for Disambiguation

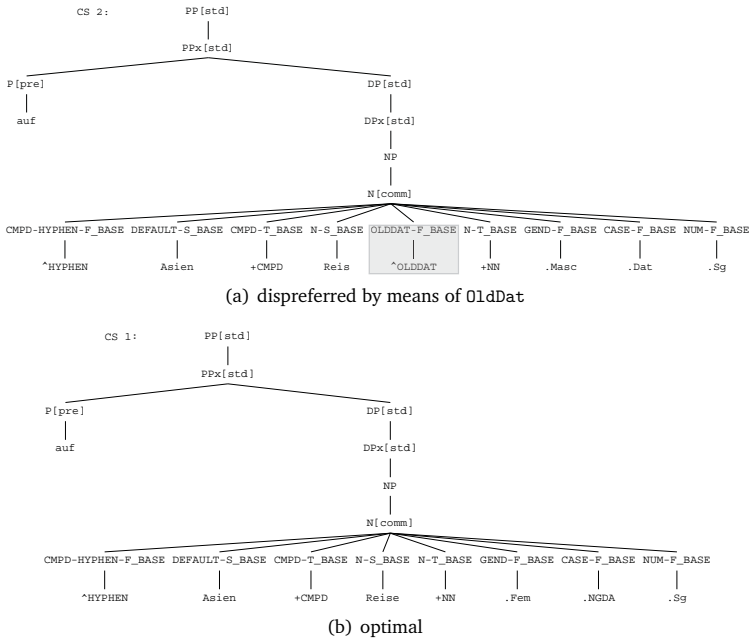


Figure 6.25: Competing c-structures for (6.31)

(6.33) Kernpunkt der Reform ist die Einführung einer
 Crucial point the-GEN reform is the introduction a-GEN
 Sonderabgabe auf alle Einkommen.
 special tax on all incomes.

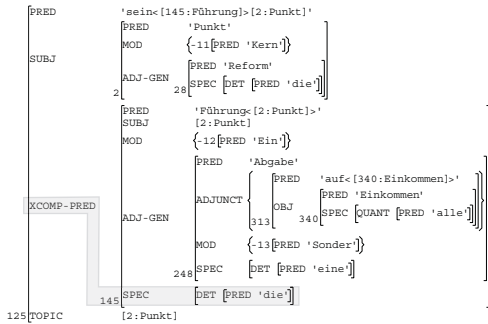
'The crucial point of the reform is the introduction of a special tax on all sorts of income.'³³

- PPdirAsPREDP is projected in the PREDP rule in *vp.gram.1fg*. It marks the attribution of the function XCOMP-PRED to a directional PP and disprefers this interpretation to alternative readings. For (6.34), it correctly disprefers the reading shown in Figure 6.27(a) with respect to the reading shown in Figure 6.27(b) (both p. 84).

³³s25931

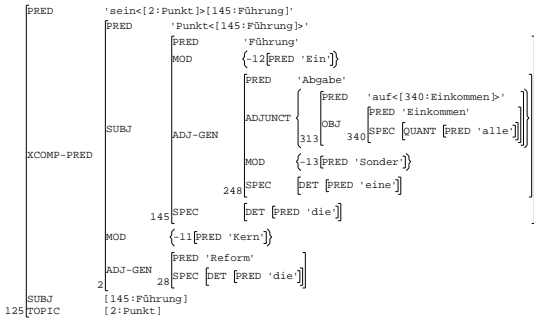
6.2 Optimality marks in the original German grammar

"Kernpunkt der Reform ist die Einführung einer Sonderabgabe auf alle Einkommen."



(a) dispreferred by means of DefinitePREDD

"Kernpunkt der Reform ist die Einführung einer Sonderabgabe auf alle Einkommen."



(b) optimal

Figure 6.26: Competing f-structures for (6.33)

(6.34) Schlechter sind die Nachrichten aus der Bauwirtschaft;
 Worse are the news from the constr. industry;
 'Worse is the news from the construction industry;/The news is
 worse from the construction industry.'³⁴

- The eighth equivalence class comprises the OT marks AdvPAdjunct, AdvPInAP and PPAjunct. These OT marks are intended to resolve ADVP and PP attachment ambiguities and help in the assignment of argument or adjunct status to ADVPs and PPs.

³⁴s10013

Manually Defined OT Constraint Rankings for Disambiguation

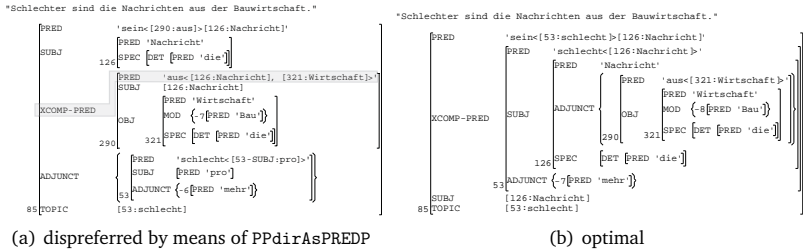


Figure 6.27: Competing f-structures for (6.34)

- AdvPAdjunct is projected in the ADVPfunc_desig template in `vp.tmpl.lfg`. It marks the attribution of the function ADJUNCT to an ADVP and is intended to disprefer ADJUNCT readings of ADVPs to argument readings. For (6.35), it correctly causes the OBL-MANNER reading of *skeptisch*, illustrated in Figure 6.28(b) to be preferred over the ADJUNCT reading, illustrated in Figure 6.28(a).

(6.35) Ost-SPD begegnet Lafontaine skeptisch
 East SPD meets Lafontaine skeptically
 'Eastern SPD sees Lafontaine skeptically/Eastern SPD meets Lafontaine skeptical(ly)'³⁵

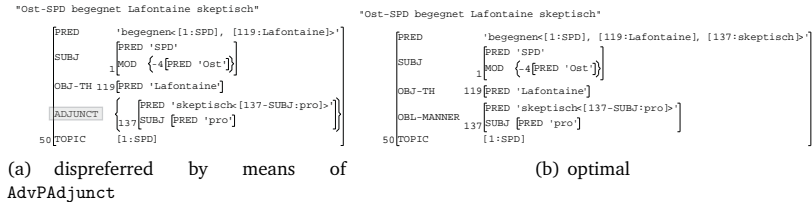


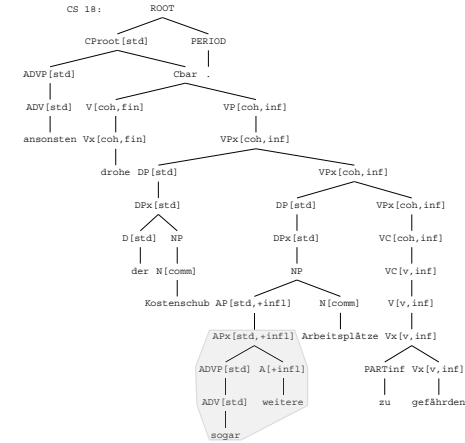
Figure 6.28: Competing f-structures for (6.35)

- AdvPInAP is projected in the APconst-PRE macro in `ap.gram.lfg`. It marks ADVPs that are attached to an AP node and disprefers this option with respect to alternative readings. For (6.36), it correctly disprefers the attachment of the adverb *sogar* to the AP headed by *weitere*, illustrated in Figure 6.29(a).

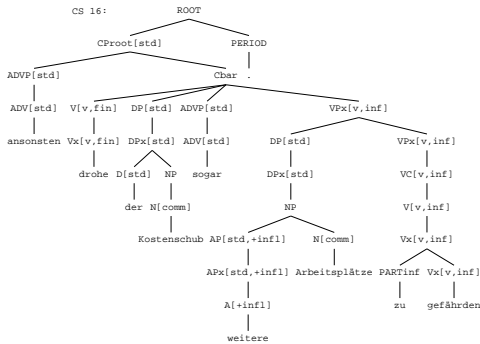
³⁵s33309

6.2 Optimality marks in the original German grammar

- (6.36) Andernfalls drohe der Kostenschub sogar weitere
 Otherwise threatened the increase of costs even further
 Arbeitsplätze zu gefährden.
 jobs to endanger.
 'Otherwise, the increase of costs threatened to endanger even
 more jobs.'³⁶



(a) dispreferred by means of AdvPInAP



(b) optimal

Figure 6.29: Competing c-structures for (6.36)

³⁶s15950

- PPA_{adjunct} is projected in the DP_{post-PP} template in `dp.tmpl.lfg` and in the PP_{func_desig} template in `vp.tmpl.lfg`. It works similarly to AdvP_{adjunct} and is intended to cause argument readings of PPs to be preferred over ADJUNCT readings. In (6.37), it correctly causes the OBL-LOC reading of the PP *im Mittelpunkt der Reise* to be preferred over the ADJUNCT reading.
- (6.37) Die wirtschaftliche Zusammenarbeit und Kohls umstrittener
 The economic cooperation and Kohl's controversial
 Besuch [...] stehen im Mittelpunkt der Reise.
 visit [...] stand in the center the-GEN trip.
 'The economic cooperation and Kohl's controversial visit [...] are in the center of this trip.'³⁷
- The ninth equivalence class comprises MassInP1 and Name. These OT marks are intended to resolve lexical ambiguities involving nouns.
 - MassInP1 is projected in the MASS-NOUN template in `np.tmpl.lfg`. It disprefers readings where mass nouns occur in the plural. In (6.38), it correctly causes the analysis of the form *Kosten* as the invariably plural noun *Kosten*, illustrated in Figure 6.30(b), to be preferred over its analysis as the plural of the mass noun *Kost*, illustrated in Figure 6.30(a).
- (6.38) Um die Kosten zu drücken, [...]
 In order the costs/foods to reduce, [...]
 'In order to reduce the costs/foods, [...]'³⁸
- Name is projected in the NAMEP rule in `np.gram.lfg`. It marks simple proper names and is intended to disprefer them with respect to alternative readings. For (6.39), it correctly causes the common noun reading of *Richter*, illustrated in Figure 6.31(b), to be preferred over the proper name reading, shown in Figure 6.31(a).
- (6.39) Die Zahl der Richter sei nur um 15 Prozent
 The number the-GEN judges/Richter was only by 15 percent
 gestiegen.
 risen.
 'The number of judges/(Mz.) Richter had only risen by 15 percent.'³⁹

³⁷s20445

³⁸s10003

³⁹s20964

- The tenth equivalence class comprises `ObjInVorfeld`, `PREDPInVorfeld` and `SpecNotDistributed`. The first two are intended to resolve `SUBJ` vs. `OBJ` and `SUBJ` vs. `XCOMP-PRED` ambiguities and the like, which arise due to the relatively free order of constituents in German. `SpecNotDistributed` is intended to resolve determiner attachment ambiguities in coordinated DPs.
 - `ObjInVorfeld` is projected in the `DPfunc_desig` template in `vp.tmpl.lfg`. It marks the coindexation of DPs as both `OBJ` or `OBJ-TH` and `TOPIC` and thus disprefers the interpretation of `Vorfeld` constituents in standard declarative clauses as `OBJS` or `OBJ-THs`. This OT mark is inspired by the observation that the default order of the typically nominal grammatical functions is `SUBJ-OBJ-TH-OBJ`. For (6.40), it correctly causes the identification of *spanische Klöster* as the `OBJ` of the sentence (see Figure 6.32(a)) to be dispreferred with respect to the attribution of the grammatical function `SUBJ` to this `Vorfeld` DP (see Figure 6.32(b)).

(6.40) *Spanische Klöster kaufen Novizinnen in Indien*
 Spanish monasteries buy novices in India
 ‘Spanish monasteries buy novices in India/It is Spanish monasteries that novices buy in India’⁴⁰

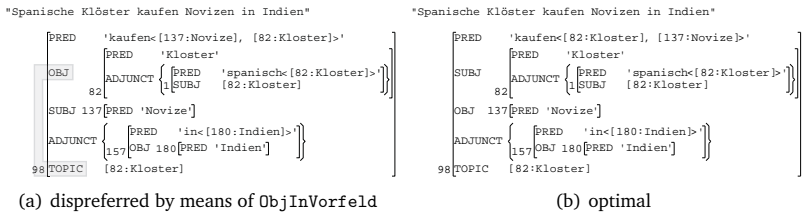


Figure 6.32: Competing f-structures for (6.40)

- `PREDPInVorfeld` works similarly to `ObjInVorfeld`. It marks the attribution of the grammatical function `XCOMP-PRED` to a `Vorfeld` DP and is intended to resolve `SUBJ` vs. `XCOMP-PRED` ambiguities. In (6.41), `PREDPInVorfeld` correctly disprefers the interpretation of the `Vorfeld` DP *eine Neuwahl oder der Gang in die Opposition* as the `XCOMP-PRED` of the sentence, making the alternative reading with the `Vorfeld` DP as the `SUBJ` optimal.

⁴⁰s21532

6.2 Optimality marks in the original German grammar

(6.41) Eine Neuwahl oder der Gang in die Opposition seien keine
 A new election or the way in the opposition were no
 vernünftigen Alternativen, sagte er.
 reasonable alternatives, said he.
 'New elections or moving into the opposition were no reasonable
 alternatives, he said.'⁴¹

- SpecNotDistributed is projected in the disjunct of the sublexical P rule in `pp.gram.lfg` that covers merged prepositions and articles like *am*, *im* etc. It marks the non-distribution of the SPEC DET contribution of these forms in cases where the form has a coordinated argument. In (6.42), it correctly disprefers the reading where the argument of *im* is the coordination of *Umbau der Regeln* and *Strukturen* with respect to the one where its argument is just headed by *Umbau*. Figure 6.33 illustrates these competing analyses.

(6.42) im Umbau der Regeln und Strukturen
 in the reorganization the-GEN rules and structures
 'in the reorganization of the rules and structures'⁴²

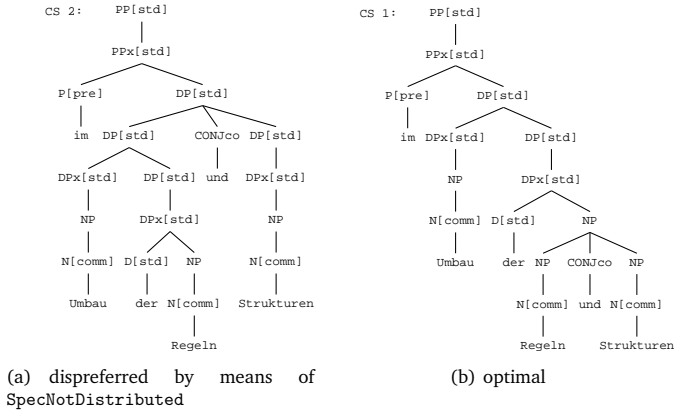


Figure 6.33: Competing c-structures for (6.42)

- The eleventh equivalence class comprises the OT marks `BridgeVerb` and `PPextraposed`. These OT marks serve the purpose of suppressing vacuous ambiguities.

⁴¹s10635

⁴²s6925

6.2 Optimality marks in the original German grammar

- (6.44) Schering leidet unter Rötung
 Schering suffers under redness
 ‘Schering suffers from redness’⁴⁴

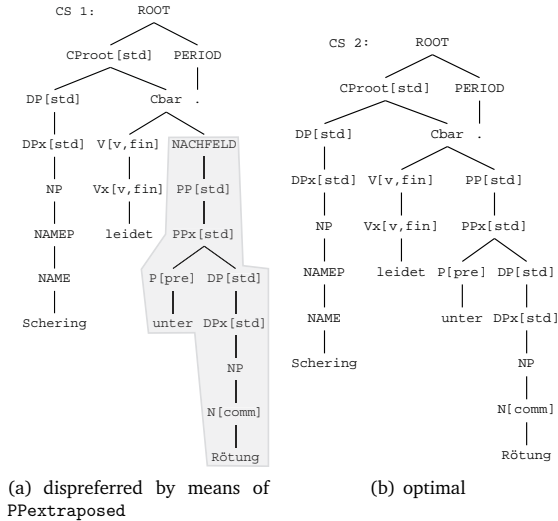


Figure 6.35: Competing c-structures for (6.44)

- +MultiWord is the first preference mark in the hierarchy (although preference marks are **not** inherently ranked lower than dispreference marks). It is projected in the MWE template in `misc.tmpl.lfg`, which is called in full form entries of multi-word entities. It prefers multi-word readings of given strings to potential alternative analyses. For (6.45), it correctly prefers the multi-word analysis of *mehr als*, shown in Figure 6.36(a) (p. 92), over the ‘regular’ analysis.

- (6.45) Das Unternehmen produzierte mehr als 20000 Fahrzeuge.
 The company produced more than 20,000 vehicles.
 ‘The company produced more than 20,000 vehicles.’⁴⁵

⁴⁴s21249

⁴⁵s13734

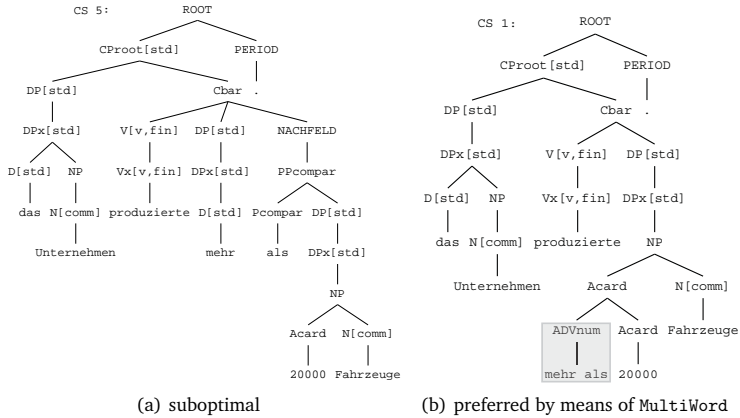


Figure 6.36: Competing c-structures for (6.45)

- The thirteenth equivalence class comprises the preference marks *DieAsDet*, *EinAsDet*, *HabenAsAux*, *NoPred*, *SeinAsAux*, *VerbParticle* and *WerdenAsAux*. What they have in common is that they are intended to resolve ambiguities that arise at the word level, due to part-of-speech ambiguities.
 - +*DieAsDet* is projected in the lexical entry of *die_art* in *dp.lex.lfg*. It prefers readings in which the forms *der*, *die*, *das* etc. are analyzed as definite articles over potential alternative readings in which they are analyzed as demonstrative or relative pronouns. Figure 6.37 illustrates these kinds of alternative analyses for (6.46).

(6.46) [...], setze die EKD auf Stellenstreichungen.
 [...], sets the/this one EKD on job cancellations.
 '[...] sets the EKD/this one EKD on job cancellations'⁴⁶

 - +*EinAsDet* is projected in the lexical entry of *eine_art* in *dp.lex.lfg*. It prefers readings in which the forms *eine*, *einem*, *einen* etc. are analyzed as indefinite articles over potential alternative readings where they are analyzed as cardinal numbers or indefinite pronouns. Figure 6.38 (p. 94) illustrates these kinds of alternative analyses for (6.47).

⁴⁶s10410

6.2 Optimality marks in the original German grammar

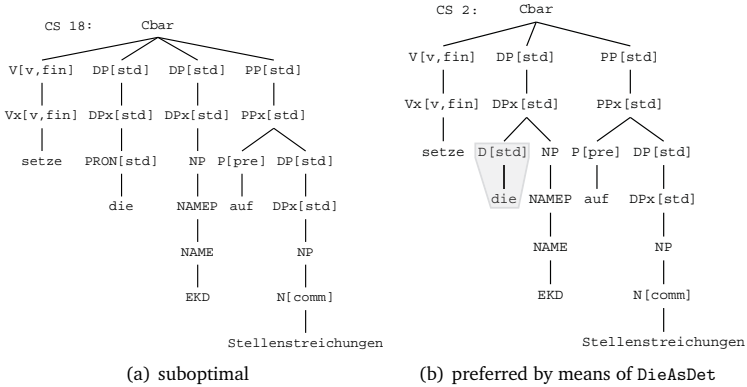


Figure 6.37: Competing c-structures for (6.46)

(6.47) Er sprach von einem bewegenden Moment.
 He spoke of a moving moment.
 ‘He spoke of a moving moment/He spoke moving moment from one.’⁴⁷

- +HabenAsAux is projected in the lexical entry of *haben* as a Vaux-S in *vp.lex.lfg*. It prefers analyses where *haben* is an auxiliary, such as the one shown in Figure 6.39(b), over alternative analyses, such as the one shown in Figure 6.39(a) (both p. 95).

(6.48) Die Mehrheit hat einen sofortigen Teststopp gefordert.
 The majority has an immediate test stop called for.
 ‘The majority has called for an immediate stop of tests./The majority has an immediate stop of tests in a requested way.’⁴⁸

- +NoPred is projected in the PRON-NOPRED template in *dp.tmpl.lfg* as well as in the DUMMY template in *vp.tmpl.lfg*. In the former, it marks the inherently reflexive reading of the reflexive pronouns; in the latter, it marks the expletive reading of *es*. It thus prefers analyses where a reflexive pronoun or *es* is a non-thematic argument, such as the one shown in Figure 6.40(b) (p. 95), over alternative readings where it is a thematic argument that obligatorily has a PRED.

⁴⁷s28038

⁴⁸s31453

Manually Defined OT Constraint Rankings for Disambiguation

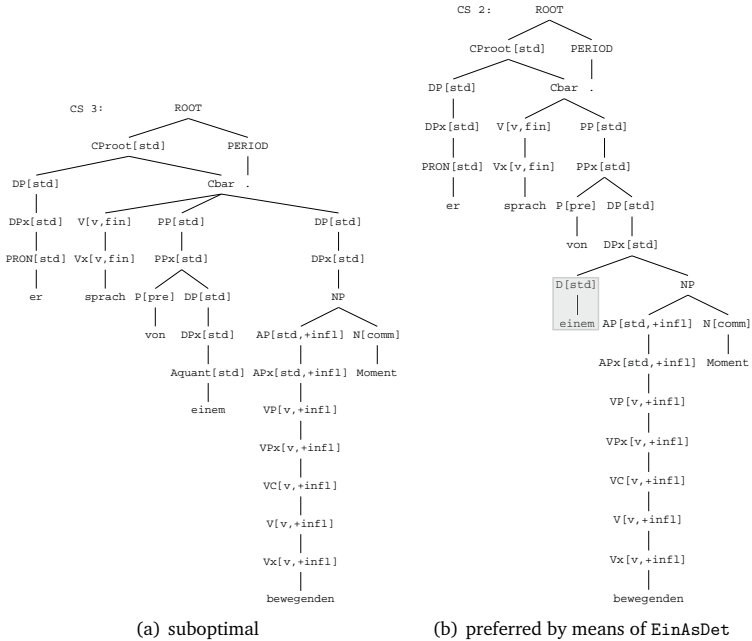


Figure 6.38: Competing c-structures for (6.47)

(6.49) Wenn Kongreß und Präsident sich bis dahin nicht
 If Congress and President themselves until then not
 geeinigt haben, [...] agreed have, [...]
 'If Congress and President have not agreed until then/If
 Congress and President have not unified themselves until then,
 [...]'⁴⁹

- +SeinAsAux is projected in the lexical entry of *sein* as a Vaux-S in vp.lex.lfg. It prefers analyses where *sein* is an auxiliary, such as the one shown in Figure 6.41(b) (p. 96), over alternative analyses.

⁴⁹s21483

6.2 Optimality marks in the original German grammar

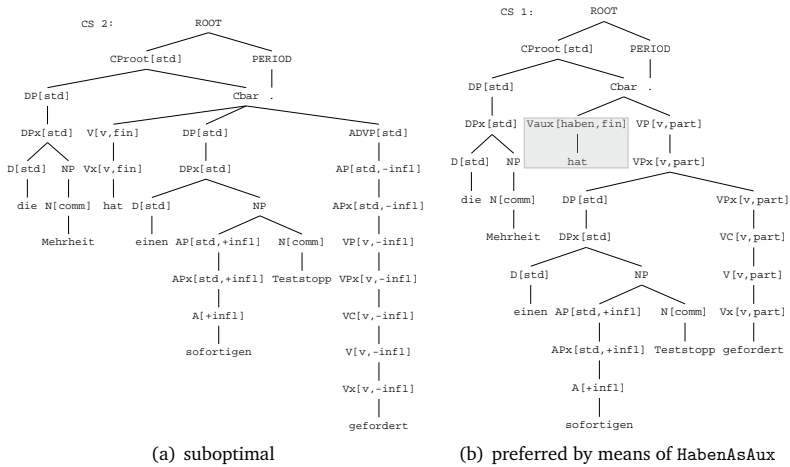


Figure 6.39: Competing c-structures for (6.48)

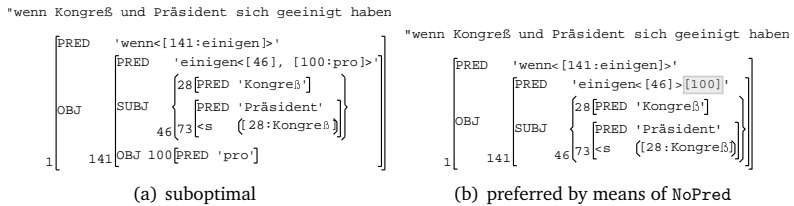


Figure 6.40: Competing f-structures for (6.49)

(6.50) Für die nächsten Tage sind weitere deutsche Hilfsflüge geplant.
 For the next days are more German aid flights planned.
 'For the next days, more German aid flights are planned./More
 German aid flights are for the next days in a planned way.'⁵⁰

- +VerbParticle in the VERB-PARTICLE template in vp.tmp1.lfg. It prefers analyses where a given form is a verb particle, such as the one shown in Figure 6.42(b) (p. 97), over alternative analyses where this same form is an adverb, an adjective etc.

⁵⁰s1304

Manually Defined OT Constraint Rankings for Disambiguation

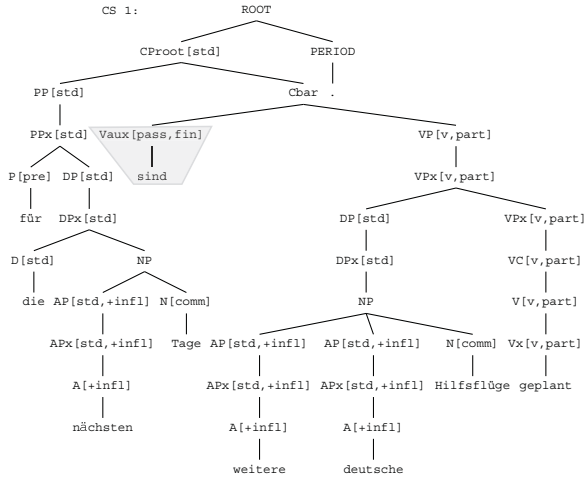
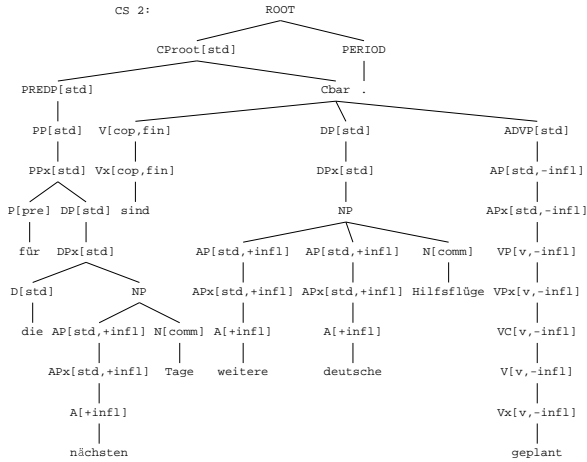


Figure 6.41: Competing c-structures for (6.50)

6.2 Optimality marks in the original German grammar

- (6.51) Die Importe aus der EG [...] gingen dabei zurück.
 The imports from the EC [...] went meanwhile back.
 ‘Meanwhile, imports from the EC diminished/went back.’⁵¹

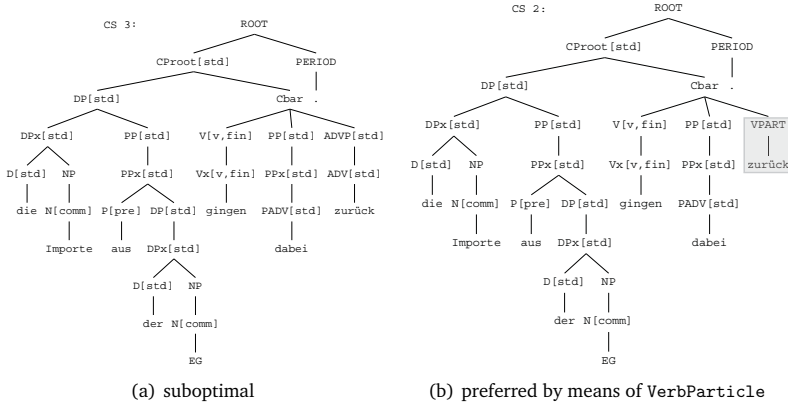


Figure 6.42: Competing c-structures for (6.51)

- +WerdenAsAux is projected in the lexical entries of *werden* and *werden.aux* as *Vaux-Ss* in *vp.lex.lfg*. It prefers analyses where *werden* is an auxiliary, such as the one shown in Figure 6.43(b) (p. 98), over alternative analyses.

- (6.52) Wie die Polizei mitteilte, wurden acht Protestanten
 As the police reported, were eight Protestants
 festgenommen.
 arrested.
 ‘As the police reported, eight Protestants were arrested/became
 arrested.’⁵²

- The fourteenth equivalence class comprises the preference marks *ComplexNum*, *DieAsRelPron*, *LexDeverbalAdj*, *LexDeverbalAdj*, *SupportVerb* and *UnitNoun*. Similarly to the preference marks in the thirteenth equivalence class, these OT marks are mainly intended to resolve lexical ambiguities.

⁵¹s1098

⁵²s1403

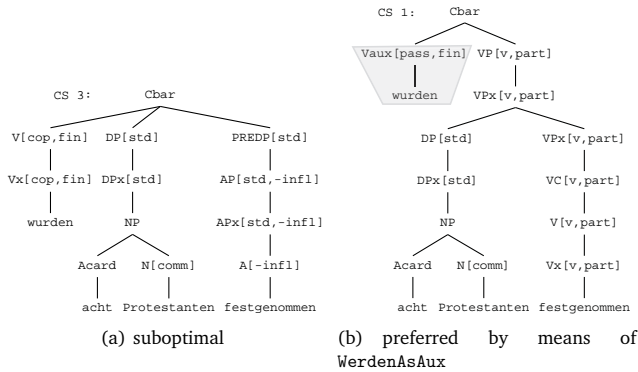


Figure 6.43: Competing c-structures for (6.52)

- +ComplexNum is projected in the disjunct of the Acard rule in ap.gram.lfg that allows for the analysis of complex numbers, such as *50 Millionen*, as Acards. It prefers this analysis over alternative analyses, which most often involve the analysis of the cardinal noun as a measure noun. The two competing analyses of (6.53) are given in Figure 6.44.

(6.53) für 50 Millionen Mark
for 50 million marks
'for 50 million marks'⁵³

- +DieAsRelPron is projected in the lexical entry of *die_rel* in dp.lex.lfg. It prefers analyses where *der*, *die*, *das* etc. are relative pronouns over alternative readings.
- +LexDeverbalAdj is projected in the LEXICALIZED-ADJ template in ap.tmpl.lfg. This template is called in all lexical entries of so-called lexicalized participles, such as *dringend* in (6.54). LexDeverbalAdj causes readings where a given form is analyzed as a lexicalized participle, such as the one shown in Figure 6.45(b) (p. 100) to be preferred over 'regular' analyses.

(6.54) ohne dringenden Tatverdacht
without urgent deed suspicion
'without strong/irrupting suspicion'⁵⁴

⁵³s24864

⁵⁴s219

6.2 Optimality marks in the original German grammar

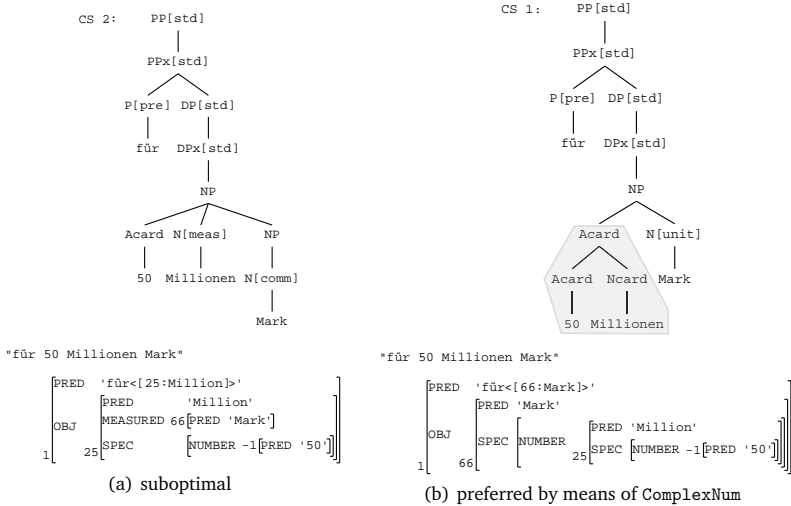


Figure 6.44: Competing c- and f-structures for (6.53)

- +SupportVerb is projected in the subcategorization templates for support verbs in `support-verb.lex.lfg`. It prefers support verb readings to 'normal' verb readings in cases where they give rise to alternative solutions. Figure 6.46 (p. 100) shows such alternative solutions for (6.55).

(6.55) Offizielle Gesprächspartner stünden zur Verfügung.
 Official interlocutors stand to the disposal.
 'Official interlocutors were available.'⁵⁵

- +UnitNoun is projected in the UNIT-NOUN-FULL template in `np.tmpl.lfg`. It is a robustness OT mark intended to avoid vacuous ambiguities due to unit nouns that have both a full form entry and a stem entry as unit nouns. Figure 6.47 (p. 101) shows alternative readings for (6.56) that are due to redundant lexical entries of this type.

(6.56) [...], die 27,4 Meter lang ist.
 [...], which 27.4 metres long is.
 '[...], which is 27.4 metres long.'⁵⁶

⁵⁵s24801

⁵⁶s848

Manually Defined OT Constraint Rankings for Disambiguation

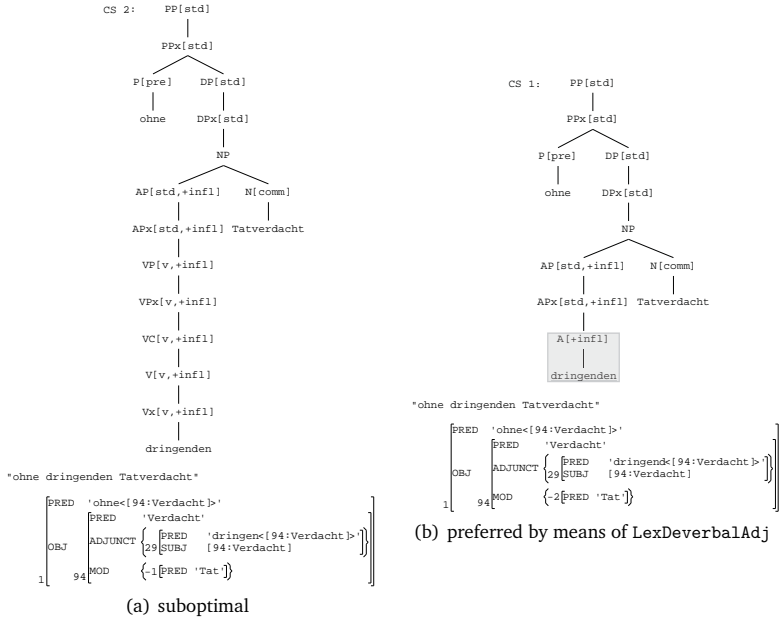


Figure 6.45: Competing c- and f-structures for (6.54)

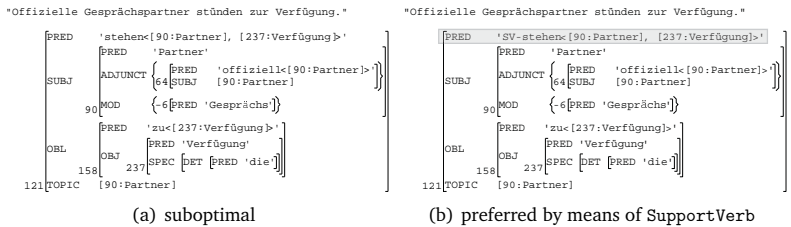


Figure 6.46: Competing f-structures for (6.55)

- The fifteenth equivalence class comprises the preference marks CanonPREDP, DPagreement, FirstNameInNAMEP and UnmarkedForm. Whereas CanonPREDP and FirstNameInNAMEP are intended to resolve structural ambiguities, DPagreement and UnmarkedForm are supposed to resolve potential ambiguities that arise because, with appositions and with the arguments of the prepositions *als* and *wie*, there is variation in

6.2 Optimality marks in the original German grammar

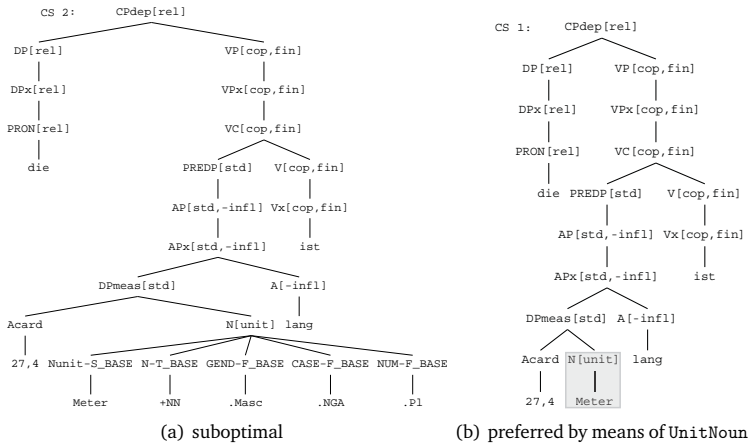


Figure 6.47: Competing c-structures for (6.56)

the use of agreement vs. the use of the nominative as a default case.

- +CanonPREDP is projected in the DP and AP disjuncts of the PREDP rule in `vp.gram.lfg`. It prefers the interpretation of DPs and APs as XCOMP-PREDS to alternative readings. For (6.57), for example, it favors the analysis shown in 6.48(b) over the one illustrated in 6.48(a).

(6.57) Sie sind Menschen geblieben.
 They have human beings stayed.
 'They stayed human beings./They were left to human beings.'⁵⁷

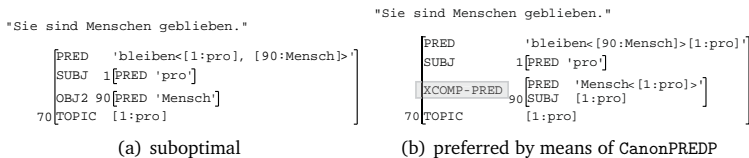


Figure 6.48: Competing f-structures for (6.57)

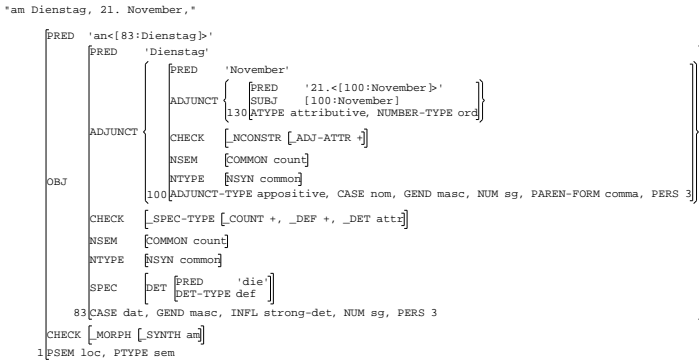
- +DPAgreement is projected in the PREFER-AGR template in `vp.tmpl.lfg`. It prefers analyses where XCOMP-PREDS agree in case,

⁵⁷s26481

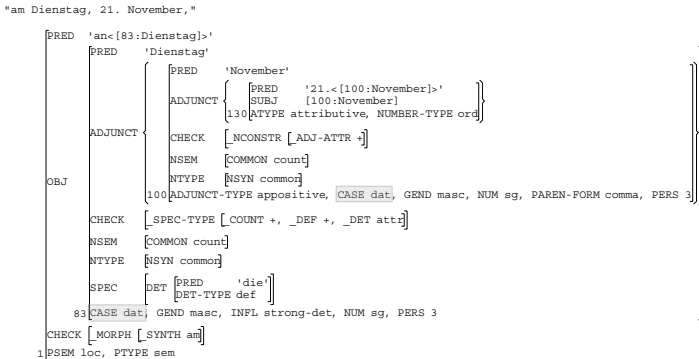
Manually Defined OT Constraint Rankings for Disambiguation

gender and number with their SUBJS or appositions agree with their heads. For (6.58), DPagreement causes the analysis with case agreement between *November* and *Dienstag*, illustrated in Figure 6.49(b) to be preferred over the analysis without case agreement, illustrated in Figure 6.49(a).

(6.58) am Dienstag, 21. November,
 on the Tuesday, 21st November,
 ‘on Tuesday, November 21st’⁵⁸



(a) suboptimal



(b) preferred by means of DPagreement

Figure 6.49: Competing f-structures for (6.58)

⁵⁸s35470

6.2 Optimality marks in the original German grammar

- +FirstNameInNAMEP is projected in the NAME-MOD-FIRST-NAME template in `np.tmpl.lfg`. It prefers analyses where a known first name is part of a NAMEP rather than a separate DP. In (6.59), it correctly prefers the analysis where *Reinhard Frieling* is analyzed as a single NAMEP (and hence as a single DP) rather than as two separate DPs. The two competing c-structures are illustrated in Figure 6.50.

(6.59) [...], sagt zum Auftakt Reinhard Frieling, [...]
 [...], says to the beginning Reinhard Frieling, [...]
 ‘[...], Reinhard Frieling, [...], says in the begin-
 ning./[...],Reinhard says to Friedling, [...], in the beginning.’⁵⁹

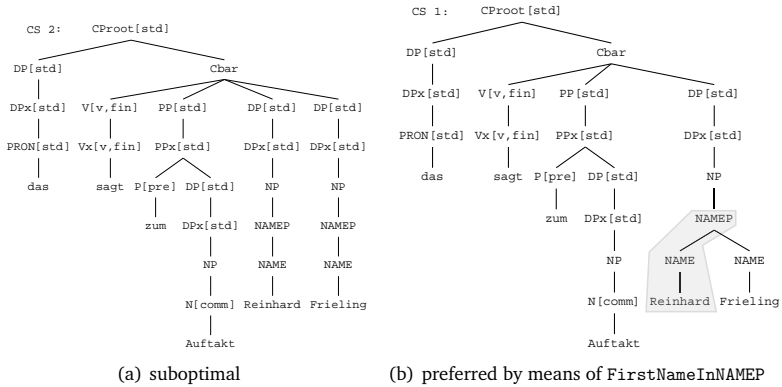


Figure 6.50: Competing c-structures for (6.59)

- +UnmarkedForm is projected in the PREFER-OBJ-NOM template in `pp.tmpl.lfg`. As its name indicates, this template, which is called in the lexical entries of the prepositions *als* and *wie*, prefers analyses where its object is in the nominative as the ‘unmarked’ form, such as the one in Figure 6.51(b) (p. 104), over alternative readings.

(6.60) Wer geht als nächster baden?
 Who goes as next bath?
 ‘Who’s going to take a bath next?’⁶⁰

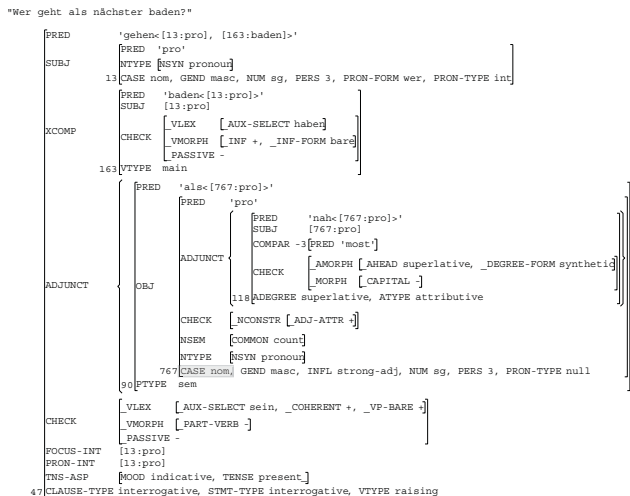
⁵⁹s14769

⁶⁰s19026

Manually Defined OT Constraint Rankings for Disambiguation



(a) suboptimal



(b) preferred by means of UnmarkedForm

Figure 6.51: Competing f-structures for (6.60)

6.2 Optimality marks in the original German grammar

- The last equivalence class of general OT marks comprises ClauseExtraposited and QuantAsDet. These preference marks serve the purpose of suppressing certain kinds of vacuous ambiguities, in a way similar to BridgeVerb and PPextraposited.
 - +ClauseExtraposited is projected in the ADVP-SENT-RESTR template in `advp.tmpl.lfg` as well as in the NACHFELD-CPcompar, NACHFELD-CPre1, NACHFELD-ARG and NACHFELD-MODlast macros in `cp.gram.lfg`. It marks adverbial clauses, comparative clauses, relative clauses and argument clauses and VPs in the Nachfeld. It is intended to prefer analyses where these clauses are analyzed as (part of) the Nachfeld as opposed to (part of) the Mittelfeld. For (6.61), it prefers the c-structure in Figure 6.52(b) over the c-structure in Figure 6.52(a) (both p. 106), the corresponding f-structures being identical.

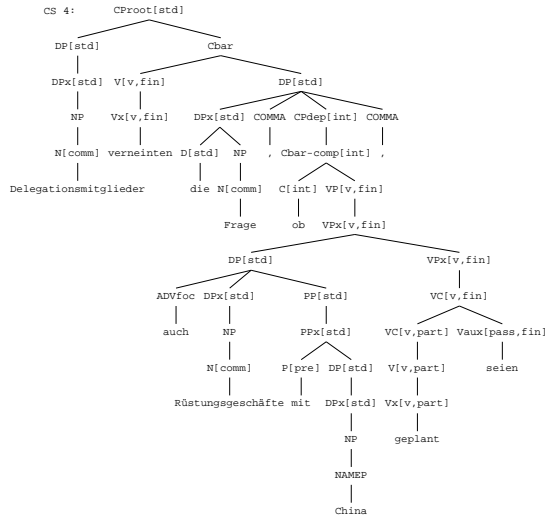
(6.61) Delegationsmitglieder verneinten die Frage, ob auch
Delegation members negated the question whether also
Rüstungsgeschäfte mit China geplant seien.
armament deals with China planned were.
'Members of the delegation answered in the negative to the question whether they planned armament deals with China.'⁶¹
 - +QuantAsDet is projected in the lexical entry of *weniger* as INDEF-S in `dp.lex.lfg` as well as in the AMBIG-INFL template in `dp.tmpl.lfg`. It prefers analyses of forms that can be both D and Aquant, such as *beide*, as Ds. For (6.62), it causes the c-structure in Figure 6.53(b) to be preferred over the c-structure in Figure 6.53(a) (both p. 107), the corresponding f-structures being identical.

(6.62) Auch von daher seien beide Bewerber gut geeignet.
Also from there were both candidates well suited.
'Also for this reason, both candidates were well suited.'⁶²

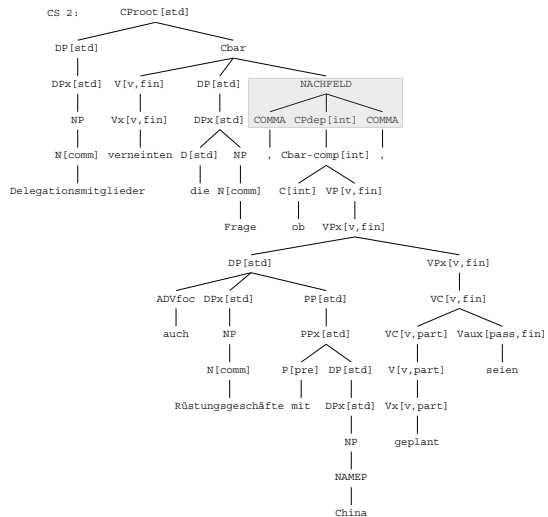
⁶¹s22701

⁶²s11893

Manually Defined OT Constraint Rankings for Disambiguation



(a) suboptimal



(b) preferred by means of ClauseExtrposed

Figure 6.52: Competing c-structures for (6.61)

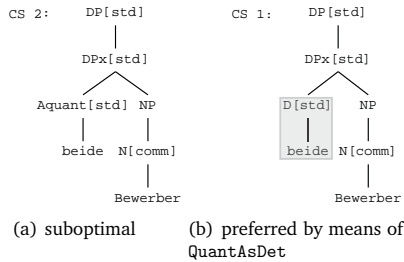


Figure 6.53: Competing c-structures for (6.62)

6.3 Summary

In this chapter, we have motivated the use of the manually defined Optimality-Theoretically inspired disambiguation mechanism proposed by Frank et al. (2001) and implemented in XLE, and we have presented a detailed documentation of the OT marks that are used with this mechanism in the initial German *ParGram* LFG. Among these OT marks, three major categories can be distinguished: the so-called NOGOOD OT marks, which deactivate certain (parts of) grammar rules or lexical entries; the so-called STOPPOINT OT marks, which interact with the parsing process by dividing the grammar into a core grammar, used in a first parsing attempt, and one or several marginal layers that are sequentially taken into account when prior parsing attempts have failed; and finally, the general OT marks, which act as a filter on all analyses found by the grammar that is applied after parsing proper.

Chapter 7

Corpus-Based Learning of OT Constraint Rankings

In this chapter, we present a method for adjusting the OT-mark-driven disambiguation used in the German *ParGram* LFG, and actually any grammar that uses the OT-mark-driven disambiguation mechanism, on the basis of corpus data. It is important to note in this context that the task of the OT-mark-driven disambiguation module changes from being the only means of ranking analyses with respect to each other to being a pre-filter, as presented in Figure 2.5 (p. 11) already. This shift has important consequences for the evaluation of the module, since recall (or filter fidelity) and precision (or filter efficiency) are no longer equally important, but recall (or filter fidelity) is of higher importance. Neither in putting together our training data nor in applying the model do we want to lose an intended reading due to the pre-filter. In other words, filter fidelity should be as close as possible to 100% (say, above 95%), while filter efficiency is less important.

7.1 Disambiguation: a two-stage approach

In Chapter 6, we saw that XLE provides a disambiguation mechanism that can, to a large extent, be easily controlled by the grammar writer. Given the extremely high number of analyses that a broad-coverage grammar, even if it is linguistically precise, necessarily produces for average sentences, and considering that this high number can very efficiently be reduced by the use of an OT mark ranking as a filter, we would like to continue to take advantage of this mechanism. That is why, in Chapter 2 already, the OT-mark-driven disambiguation module was presented as part of our two-stage disambiguation architecture.

This being said, we are aware of the fact that the purely manually specified OT mark ranking presented in Chapter 6 sometimes evaluates the intended

reading of a given sentence as suboptimal. This has historic reasons. The OT-mark-driven disambiguation was the only disambiguation device for several years and the grammar was developed by several people in several stages, so that interactions between OT marks were not always foreseen. But it also has a more technical reason, which is that it is very difficult, if not impossible, for a grammar writer to foresee the exact effect of a large number of OT marks that, moreover, have to be ranked relatively to each other. In the construction of the training corpus for the log-linear model that performs the final disambiguation step, the fact that intended readings are sometimes evaluated as suboptimal implies that the corresponding sentences are lost as potential training data. In the application of our system to unseen text, it means that the desired reading may actually not even be among the candidates among which the log-linear model makes the selection that is evaluated in the end.

In order to reconcile the desires of, on the one hand, maintaining the possibility for the grammar writer to contribute to disambiguation decisions in a linguistically informed way by specifying OT mark annotations in the grammar and of, on the other hand, guaranteeing, to the largest extent possible, that the OT-mark-driven pre-filter does not eliminate intended readings, we have developed a methodology for trimming the pre-filter on the basis of corpus data. It thus allows us to take advantage of the strength of the OT-mark-based approach, while controlling its main weakness.

Apart from this conceptual motivation, there is an essentially technical motivation for the use of a pre-filter in our architecture: The training procedure of the log-linear model used for the final disambiguation step is a discriminative technique. It relies on data for which a proper subset of the readings has been determined as compatible with some sort of annotation (in our case, the TIGER Treebank annotation). Determining the annotation-compatible subset of readings by means of the Perl programme presented in Chapter 4 is only feasible in a reasonable amount of time for moderately ambiguous data, i.e. f-structure charts that contain maximally thousands of readings, but not hundreds of thousands or even millions. The average number of optimal readings in the f-structure charts produced for the sentences of our test set that receive a spanning analysis is around 140, while the average number of all readings produced by the grammar for these sentences is in the tens of millions. The OT-mark-based filter thus makes an important contribution to keeping the construction of training data feasible. Furthermore, the parameter estimation for the log-linear model is an iterative process that passes the training data multiple times. Hence, to keep the process tractable on medium-size to large training corpora, the set of competing analyses evaluated by the log-linear model has to be limited. Using only the analyses that pass a linguistically motivated pre-filter is a very desirable setup.

7.2 Methodology

In this section, we discuss our experimental methodology at a conceptual level, a more detailed description and the results follow in Section 7.3.

7.2.1 Trimming the linguistic pre-filter

In past work on the *ParGram* grammars, both the introduction of OT marks and the specification of their relative ranking was done manually. This is problematic since the various marks affect phenomena that were integrated into the grammar at different development stages, and often the appropriate relative ranking can only be determined empirically. Moreover, the question of whether or not a particular mark should be active in the two-stage filter architecture we described is also hard to answer in isolation. (But note that the structural specification and the polarity of the candidate marks *are* aspects about which the grammar writer can make an informed decision.)

We have therefore conducted the first systematic study applying empirical methods in order (i) to determine the ranking of the OT marks, and (ii) to decide which marks should be left out of the first disambiguation stage. It has been part of this study to explore measures for the quality of a particular specification of the pre-filter. We present results for the German *ParGram* LFG. Some of the results of our experiments are surprising and provide some interesting insights in the workings of the two-stage filter architecture.

While the technical results we report in the following sections make reference to project-specific details of our system architecture, we believe that many of the higher-level observations will carry over very well to other projects involving a linguistically motivated core module that is applied in a broader context of empirically tuned system components.

7.2.2 Measuring the quality of the pre-filter

The fact that the component we are interested in here is a pre-filter in the context of a two-stage system has special consequences for quality assessment. It is not necessary that the pre-filter remove *all* incorrect readings – since it is followed up by a sophisticated second disambiguator. On the other hand, it is highly undesirable if the pre-filter accidentally removes the correct reading, since this would make data unusable for the second stage. One might describe this as a task in which recall is of greatest importance, and precision should be traded off for recall; but in order to do justice to the special setting, we call the relevant measures ‘filter fidelity’ and ‘filter efficiency’. Filter fidelity is defined as the proportion of sentences for which the OT mark ranking under consideration keeps the correct reading among the optimal reading(s). The

intuition behind filter efficiency, on the other hand, is to measure the proportion of readings among all incorrect readings of a sentence which are discarded by the OT mark ranking as suboptimal. Concretely, we calculate it as the number of readings discarded by the filter divided by the total number of readings minus one.¹ Filter fidelity is our main criterion and it should be as close as possible to 100%, but filter efficiency does have a certain importance as well, of course, since filtering a maximum of bad readings while losing a minimum of good readings is the goal of this whole enterprise. As a combined quality measure, we therefore provide a weighted F-score where filter fidelity is weighted more strongly than filter efficiency, namely

$$F_{0.5} = (1 + 0.5^2) \times \frac{FE \times FF}{FE + 0.5^2 \times FF}.$$

7.2.3 Corpus-based learning of a ranking

Our experiments start out with the manually specified OT mark ranking in the initial version of the German *ParGram* LFG documented in Chapter 6. An obvious technique to try out is to learn a ranking automatically from corpus data for which the readings that are compatible with the treebank annotations have been labeled. The filter quality with the learned ranking can then be compared against the manual ranking and a uniform ranking, which consists in giving all marks the same rank.

For corpus-based learning of the OT mark ranking, one could in theory apply the classical Constraint Demotion Algorithm from the OT literature (Tesar & Smolensky 1998); however, due to the variation in the data the algorithm might not converge. Therefore we transform the classical discrete constraint ranking into a continuous numerical ranking for the purpose of learning. This allows us to apply robust learning algorithms like the Gradual Learning Algorithm (GLA) proposed by Boersma (1998). In learning, the system's current numerical ranking (with some noise added to determine each constraint's particular rank) is used in order to disambiguate a sentence from the training data. When the predicted solution does not match any of the treebank-compatible analyses, all constraints ranked too low are promoted by a small increment (controlled by the so-called plasticity parameter) and all constraints ranked too high are demoted. The noise added in application has the effect that constraints with a similar ranking can "swap" their relative rank, which leads to variation in the data, as it is often observed. This variant of OT is thus often called Stochastic OT.

¹At the LFG Conference in Bergen we presented figures that were based on a slightly different definition of filter efficiency, namely the number of readings discarded by the filter divided by the total number of readings. Since this initial definition prevents filter efficiency from taking 1.0 as a value and it is highly dependent on the total number of readings for a given sentence, our new definition is more appropriate.

7.2.4 Augmenting the set of OT marks

We also performed an additional experiment besides learning a ranking for just the OT marks specified by the grammar writers: we explored how pre-filter quality is affected if we systematically augment the existing set of OT marks to ensure that for common disambiguation decisions, sufficiently fine-grained distinctions in the OT marks are available. It is conceivable that for certain decisions, the OT mark set is too “sparse” to produce a reliable result, whereas a richer OT mark set might behave in a more balanced way. This is because in Stochastic OT, competing marks may form clusters in the numerical ranking, and the addition of new constraints may have the effect of making such a cluster more stable.²

In a pilot study, we thus established OT tableaux containing the OT marks employed in the German *ParGram* LFG and ran the GLA on these. This allowed us to identify OT marks which were reranked particularly often and/or which were regularly both demoted and promoted. Two such marks were *ObjInVorfeld*³ and *LabelP*.⁴ After inspection of a certain number of sentences where these OT marks made the correct reading(s) suboptimal, we introduced new, more fine-grained OT marks such as *ObjPersPronoun* (which disprefers the interpretation of personal pronouns as objects) and *SubjIndef* (which disprefers the interpretation of indefinite noun phrases as subjects), hoping these would result in making the correct reading(s) optimal for more sentences.

In order to be able to control whether this is effectively the case, we established two sets of tableaux: the first one, henceforth the ‘all marks’ set, contains both the 59 original⁵ and the 54 additional, newly introduced OT marks; the second one, henceforth the ‘original marks’ set, contains only the original marks. Both sets were in turn split up into a training and a test set, so that we can examine how well rankings learned from the training sets generalize to unseen data.

We then ran the GLA on the training portions of both the ‘all marks’ set and the ‘original marks’ set. For training, we used a “traditional” GLA setting, i.e. a setting where the effective numerical rank of an OT mark diverts from its

²As we show in Subsection 7.3.2 however, we did not obtain a more relaxed filter by providing a larger set of interacting OT marks.

³*ObjInVorfeld* disprefers the interpretation of case-ambiguous noun phrases in the Vorfeld, i.e. the position in front of the finite verb in verb-second clauses, as objects in sentences such as [NP-SUBJ *Hans*] *sieht Maria* ‘John sees Mary’ vs. [NP-OBJ *Hans*] *sieht Maria* ‘John, Mary sees’.

⁴*LabelP* disprefers the interpretation of a noun phrase as a close apposition to another noun phrase in sentences such as *Hans stellt* [NP-OBJ *das Auktionshaus Ebay*] *vor* ‘Hans presents the auction house Ebay’ vs. *Hans stellt* [NP-OBJ *das Auktionshaus*] [NP-OBJ-TH *Ebay*] *vor* ‘Hans presents the auction house to Ebay’.

⁵These are the 51 general OT marks presented in the previous chapter plus 8 OT marks that had been introduced by the grammar writers when we started the experiments described in this chapter.

grammatical rank within a normal distribution due to added noise and where marks making wrong predictions are demoted or promoted on the numerical scale by a constant called plasticity (see Boersma (1998)).

In order to evaluate the resulting rankings, we used a variant of the GLA without any noise intervening at evaluation time. This allowed us to evaluate the resulting numerical rankings as if they were strict relative rankings, which is the type of ranking used in XLE. Moreover, the application of this variant of the GLA to the data allows us to identify marks that, even with an “optimal” OT mark ranking, cause correct readings to be evaluated as suboptimal. In this sense, it is not only a tool for the evaluation of OT mark rankings as a whole, but it can also be used to evaluate how reliable individual OT marks are.

7.2.5 Relaxing the filter

An important additional step in our experiments (based both on the ‘original marks’ and based on ‘all marks’) was the attempt to modify an existing set of marks and ranking in order to increase filter fidelity – without decreasing filter efficiency too much. Besides learning a more adequate ranking, this could be achieved in the following ways: (1) deactivating certain OT marks, such that their filtering effect is removed, and (2) grouping together constraints with a very similar rank. For step (1), it is important to identify appropriate marks for deactivation. In the pre-filter scenario, marks that are typically involved in “highly contingent disambiguation decisions” (i.e., decisions that may turn out one way or the other) should be excluded from the set, since they eliminate the correct solution in relatively many cases. To identify marks for deactivation we explored two strategies: (a) manually inspecting the results obtained with the GLA variant without noise given a ranking obtained through a training run and deactivating the marks that still caused wrong predictions and (b) automatically deactivating a certain proportion of marks being associated with ranks at the lower side of the numerical scale.

For step (2) – grouping of similarly ranked constraints – we used various threshold values representing the minimal distance that two marks have to be away from each other in order to be attributed to distinct groups with distinct ranks. Such a grouping can “relax” the pre-filter in the following way: assume two dispreference marks, $Mark_1$ and $Mark_2$, end up with similar, but distinct ranks, where the numeric rank of $Mark_1$ is slightly higher than the one of $Mark_2$. Without grouping, all readings with a $Mark_1$ mark would be filtered out, regardless of their $Mark_2$ marking, whereas, after grouping, a reading with a $Mark_1$ and no $Mark_2$ mark is treated like one with a $Mark_2$ and no $Mark_1$ mark.

7.3 Experiments

7.3.1 Data

For our experiments, we parsed the 40,020 sentences of Release 1 of the TIGER Treebank⁶ with a variant of the German *ParGram* LFG (Dipper 2003) into which we had integrated the new, more fine-grained OT marks and in which the evaluation of almost all OT marks had been deactivated.⁷ Out of 40,020 TIGER sentences, 23,962 received a full parse.⁸ The resulting f-structure charts (packed f-structure representations) were matched against the f-structure charts previously derived from the TIGER graph annotation (see Chapter 4, as well as Forst (2003a,b)); OT mark profiles corresponding to the f-structure charts produced by the grammar were established, the TIGER-compatible readings being marked as target winners in them. Since the matching of the grammar output against the TIGER-derived representations is very time-intensive when the number of different analyses contained in the two f-structure charts involved (or at least in one of them) is very high, we had to limit the maximum number of matches performed between individual analyses to 10,000.⁹ By doing this, we obtained 6,418 OT mark profiles, associated with the sentences for which a proper subset of all parses is compatible to the TIGER-derived f-structure charts. (The granularity of the TIGER annotation is not sufficient to always determine one single parse as the correct one.) Sentences for which all analyses were compatible with the TIGER-derived representations (and thus ‘target winners’) had to be discarded, since there would be nothing to be learned from OT tableaux associated with sentences of this kind; the same is true for sentences for which no analysis is TIGER-compatible, since there would be no target winner in the corresponding tableau.


An example of an OT mark profile obtained this way is the one associated with the following sentence:¹⁰

⁶<http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERCorpus/>

⁷The OT mark *GuessedMassNoun* was kept active, as its deactivation would have led to such an enormous increase in the numbers of readings produced by the grammar that the matching mentioned below would not have been feasible for most sentences.

⁸The grammar version employed was not chosen for its coverage but for its adherence to *ParGram* f-structure decisions which are reflected in the packed f-structure representations derived from the TIGER Treebank. Moreover, the newly introduced OT marks caused a slight slow-down of the grammar, which caused additional timeouts compared to other grammar versions.

⁹This means that sentences for which the product of the number of analyses in the grammar output and the number of analyses in the TIGER-derived representations was greater than 10,000 were discarded.

¹⁰Note that the ‘’ in the OT mark profile does not indicate the optimal candidate according to the ranking displayed, but the candidate that should be determined as the winner by the ranking to be learned from data of this kind. Actually, the order in which the OT marks are displayed in this table just follows their alphabetical order and does not say anything about

- (7.1) Anlaß für all das gab aus Schweizer Sicht das neue österreichische
 rise for all that gave from Swiss view the new Austrian
 Abfallwirtschaftsgesetz.
 waste management law
 ‘From the Swiss point of view, it was the new Austrian waste management
 law that gave rise to all that.’

reading	AdvAttach	Obj	ObjCommon	ObjDef	ObjNoSpec	...
A1	0	1	2	1	1	...
A2	0	1	2	0	2	...
A3	0	1	2	0	2	...
A4	0	1	2	1	1	...
A5-B1	1	1	1	0	1	...
A5-B2	0	1	1	0	1	...
A6-B1	1	1	1	1	0	...
A6-B2	0	1	1	1	0	...

Table 7.1: Sample OT mark profile for sentence (7.1)

The directly resulting 6,418 OT mark profiles, which correspond to the ‘all marks’ set, were randomly split up into a training set of 5,755 and a test set of 663. Then we created the ‘original marks’ set, by replacing all values in the columns of the newly introduced OT marks by zero, and split it up into a training set of 5,755 profiles and a test set of 663 along the same lines as the ‘all marks’ set.

7.3.2 Training and first results

The 5,755 OT mark profiles of both the ‘all marks’ training set and the ‘original marks’ training set were input to an implementation of the GLA that allows for multiple target winners. The learning was performed with a plasticity of 0.2 and in 10 iterations over the whole training set, each datum being considered 5 times. The result of these training runs were two different numerical OT mark rankings, one for the ‘all marks’ set and another one for the ‘original marks’ set. The results of these two training runs are summarized in Table 7.2.

As can be seen from the evaluation figures obtained on the unseen test set, the ranking of the OT marks does not play a major role. Although the automatically learned ranking performs better than the manually determined ranking originally used in the German *ParGram* LFG, both in terms of filter fidelity

their relative ranking, so that, strictly speaking, the table is not an OT tableau.

ranking employed	original marks			all marks		
	filter fidelity	filter efficiency	weighted F-score	filter fidelity	filter efficiency	weighted F-score
after 10 iterations GLA	81.9	84.9	82.5	78.3	87.2	79.9
uniform ranking	82.7	83.3	82.8	77.2	84.1	78.5
original manual ranking	80.5	84.8	81.3	–	–	–

Table 7.2: Results of GLA learning on test sets

(81.9% vs. 80.5%) and filter efficiency (84.9% vs. 84.8%), the improvement from the latter to the former is slight, in particular for filter efficiency. Also, we have to state that, for the ‘original marks’ set, the automatically learned ranking performs worse than a uniform ranking, i.e. a ranking where all marks are equally strong, in terms of filter fidelity (81.9% vs. 82.7%), even if filter efficiency is better (84.9% vs. 83.3%). The weighted F-score we employ confirms this picture (82.8% vs. 82.5%).

Comparing the results for the ‘original marks’ set and the ‘all marks’ set, the observation is that, although the additional marks allow for a better filter efficiency, they have a negative effect on filter fidelity. This result is a bit disappointing, because initially, we had hoped to improve both filter efficiency and filter fidelity by providing the new marks. At the same time, it is not all that surprising, since the more OT marks are used for disambiguation, the more difficult it is, of course, to maximize filter fidelity.

As to the ability of the rankings to generalize from the training data to the unseen test data, we can see in Table 7.3 that both the evaluation figures themselves and the patterns observed in them are comparable between the training sets and the test sets.

ranking employed	original marks			all marks		
	filter fidelity	filter efficiency	weighted F-score	filter fidelity	filter efficiency	weighted F-score
after 10 iterations GLA	80.2	85.4	81.2	78.0	87.3	79.7
uniform ranking	81.8	82.7	82.0	76.7	83.7	78.0
original manual ranking	79.6	85.2	80.7	–	–	–

Table 7.3: Results of GLA learning on training sets

7.3.3 Relaxing the filter

Given that the filter fidelity we achieved with the learned ranking hardly exceeded 80%, we thought of ways of relaxing the OT filter in order to increase this value. At the same time, filter efficiency was not supposed to be affected too badly.

Inspecting (and deactivating) “problematic” OT marks

The first approach we took was to inspect the OT marks that, even with the automatically learned ranking, caused correct readings to be evaluated as sub-optimal. Examples of these were, as in the pilot study mentioned in 7.2.4, *ObjInVorfeld* and *LabelP*. Apparently, even the newly introduced OT marks did not allow us to counterbalance them in cases where they caused wrong predictions, which leads us to the opinion that these OT marks, instead of being evaluated in the pre-filter step, should be integrated as properties into the log-linear model. As such, they can contribute to choosing the correct reading in the final disambiguation step, where, moreover, they can interact with other properties, such as the ones that weight competing subcategorization frames against each other.

Another category of OT marks that still made wrong predictions were robustness OT marks such as *AdvAttach* and *MassInPl*. The purpose of these OT marks is mainly to disprefer fallback rules that are implemented for cases where lexical information is lacking and, as a consequence, they interact tightly with this kind of information. Due to missing or erroneous information in the lexicons, it can happen that they make wrong predictions, although they are fairly reliable in all other cases. We deactivated most of these OT marks, i.e. those which caused relatively many wrong predictions, but keep in mind that they can potentially be reactivated once the lexicons they interact with have been improved.

As a reaction to the inspection of the “problematic” OT marks, a variant of the data was created where these marks are deactivated. We henceforth call this set of data the ‘unproblematic marks’ set.

“Translating” the numerical rankings into strict rankings

For use in XLE, the numerical rankings obtained from GLA learning have to be “translated” into strict rankings in which marks may be grouped as equally strong. The easiest way of doing this is, of course, to have one group for each distinct numerical ranking. However, this may not be the most appropriate method of “translating” a numerical ranking into a strict ranking, because it completely ignores the information contained in the distance between two rankings. A possible alternative is to group all OT marks whose ranking have a distance smaller than a given threshold. This way, the number of groups of equally ranked OT marks is reduced, which should allow for better generaliza-

tion, and, more importantly, some of the information contained in the distance between rankings is taken into account. We experimented with groupings of this kind with thresholds 2.0 and 5.0. The resulting rankings were then applied to both the ‘original marks’ data set and the ‘unproblematic marks’ set. The results are shown in Table 7.4.

ranking employed	original marks			unproblematic marks		
	filter fidelity	filter efficiency	weighted F-score	filter fidelity	filter efficiency	weighted F-score
after 10 iterations GLA	81.9	84.9	82.5	95.9	62.2	86.5
grouped with threshold 2.0	82.4	84.8	82.9	96.2	62.1	86.7
grouped with threshold 5.0	82.5	84.7	82.9	96.2	62.1	86.7
uniform ranking	82.7	83.3	82.8	96.1	60.3	85.9

Table 7.4: Results of disambiguation on test sets with ‘original marks’ and ‘unproblematic marks’, marks being grouped according to different methods

Just as in our first results (cf. subsection 7.3.2), we observe that the ranking has only a rather small influence on the results. Nevertheless, filter fidelity can be improved slightly by grouping marks whose ranks are not very distant, without filter efficiency being affected considerably. Taking the figures from the training data into account (which are not displayed here), the conclusion could be that a grouping with a threshold value of 5.0 performs best, since it basically achieves the same filter fidelity as the uniform ranking, while allowing for a slightly higher filter efficiency.

More importantly, Table 7.4 shows that the deactivation of “problematic” marks can increase the filter fidelity considerably. We achieve a filter fidelity of more than 95%, while still discarding more than 60% of the readings as suboptimal.¹¹ This set-up also yields the highest weighted F-score of all our experiments: 86.7%.

Deactivating portions of the OT marks according to their ranks

An alternative approach to deactivating “unreliable” OT marks we experimented with was to discard a certain proportion of the marks corresponding to the ranks at the lower end of the numerical scale. We ran this experiment for both the ‘original marks’ set and the ‘all marks’ set, deactivating the lower

¹¹This can arguably be considered an underestimation, because the effect of the OT mark *GuessedMassNoun*, mentioned in subsection 7.3.1 as well, is not taken into account here, although it cuts down the number of readings considerably.

50% of the OT marks.¹² The resulting variants of the data are henceforth called ‘upper 50% original’ and ‘upper 50% all’ respectively. Tables 7.5 and 7.6 show the effect of discarding the lower 50% of the two OT mark sets. (The ranking used is the one obtained after 10 iterations of the GLA.)

original marks			upper 50% original		
filter fidelity	filter efficiency	weighted F-score	filter fidelity	filter efficiency	weighted F-score
81.9	84.9	82.5	99.5	41.3	77.6

Table 7.5: Results of disambiguation with ‘original marks’ and ‘upper 50% original’

all marks			upper 50% all		
filter fidelity	filter efficiency	weighted F-score	filter fidelity	filter efficiency	weighted F-score
78.3	87.2	79.9	97.0	50.5	81.9

Table 7.6: Results of disambiguation with ‘all marks’ and ‘upper 50% all’

Filter fidelity is greatly improved by this strategy, but unfortunately, there is a high price to be paid in terms of filter efficiency. In both settings, it drops to 50% or even less. Given that the filter fidelity for the ‘upper 50% all’ set is comparable to the filter fidelity for the ‘unproblematic marks’ set, but that the filter efficiency for it is considerably lower than for the ‘unproblematic marks’ set, we conclude that it is a better strategy to semi-automatically identify problematic marks and then deactivate them than just to deactivate a certain proportion of the lower ranked marks.

7.4 Summary

In this chapter, we have presented a sequence of experiments exploring ways of empirically tuning the first stage of a disambiguation architecture for linguistically precise grammars. This pre-filter is triggered by configurations that the grammar writer specifies as OT marks and uses a relative ranking among the marks. A somewhat surprising result is that training the constraint ranking on corpus data does not lead to a noticeable improvement over the use of a uniform ranking. However, it is very effective for identifying marks that are to

¹²The choice of this proportion was an arbitrary one.

7.4 Summary

be deactivated because they tend to exclude the correct readings. Both results are, of course, directly related to the somewhat unusual application context of the disambiguation routine as a pre-filter: if it were used as the only filter, one should certainly rely on a learned ranking to maximize filter efficiency, and the OT marks that are problematic in the pre-filter scenario might well play an important role. For the given two-stage scenario however, our systematic empirical exploration showed that filter fidelity can be maximized most effectively by removing unreliable marks. With the resulting system, we achieve a filter fidelity of 96%, while filter efficiency stays above 60%.

Part IV

A Log-linear Model for Disambiguation

Chapter 8

A first model

In this chapter, we present the properties used in the first log-linear model for disambiguation that we built and briefly explain the training scheme that we applied. The model is designed and trained along the lines of Riezler et al. (2002) and Riezler & Vasserman (2004), following the latter, more recent, paper wherever the two approaches differ. The chapter concludes with an evaluation of this first model, some observations about the results and an outlook to steps that are necessary to improve the model.

8.1 Properties based on XLE property templates

The first set of properties with which we conducted experiments was built on the model of the property set used for the disambiguation of English *ParGram* LFG parses. These properties are defined with the help of fourteen property templates, which are all explained in the following sections. The templates are hardwired in XLE, which allows for a very efficient extraction of properties based on them from packed c-/f-structure representations. The downside of the templates being hardwired, however, is that, at least at first sight, the property developer is confined to what the developers of the property templates anticipated as potentially relevant for disambiguation or, more precisely, for the disambiguation of English LFG analyses.

For the model presented in this chapter, properties based on all fourteen property templates are used. In the final model presented in the subsequent chapters, we use ten out of these fourteen property templates. For each of the four property templates that are discarded, it is briefly argued why it is not used.

8.1.1 C-structure-based properties

The template `cs_right_branch` is in fact a property, since it is not parameterized. According to Crouch et al. (2006), it counts the number of right children in the c-structure of a parse. Although it is relevant for disambiguation in English, it does not contribute anything (or, at most, very little) to the disambiguation of German *ParGram* LFG parses.

A further template for c-structure-based properties is `cs_conj_nonpar`, which is parameterized for the depth from which onwards the conjuncts of a coordinated structure are not parallel any more. This definition implies that the property `cs_conj_nonpar n+1` is counted if the property `cs_conj_nonpar n` is counted. The properties based on the template `cs_conj_nonpar` make it possible to disambiguate the analyses of sentences or phrases where constituents can be coordinated in multiple ways, as it is the case in (8.1).

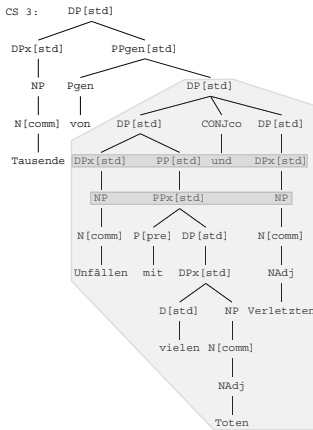
- (8.1) Tausende von Unfällen mit vielen Toten und Verletzten
thousands of accidents with many dead and injured
'thousands of accidents with many deaths and injuries'¹

Figures 8.1(a), 8.1(b) and 8.1(c) illustrate the relevant ways of coordinating constituents in (8.1). In the reading illustrated in Figure 8.1(a), the conjuncts are parallel on the first level of embedding; in other words, the same node label occurs both to the left and the right of the `CONJco` node. On the next level (and all subsequent levels), however, the conjuncts are no longer parallel, since the left side branches into a `DPx[std]` node and a `PP[std]` node and the right side only has one node. This means that `cs_conj_nonpar 2` is counted once (as are all other `cs_conj_nonpar` properties that are passed a parameter greater than 2). In the reading illustrated in Figure 8.1(b), `cs_conj_nonpar 2` is not counted because the conjuncts are parallel at embedding levels 1 and 2 under the coordinated constituent, but the branching of the left `DPx[std]` node makes them no longer parallel at level 3, so that all `cs_conj_nonpar` properties that are passed a parameter greater than or equal to 3 are counted once. In the reading illustrated in Figure 8.1(c), finally, the conjuncts are parallel all the way until level 4, where the different terminal nodes cause them to be non-parallel, so that only `cs_conj_nonpar` properties that are passed a parameter greater than or equal to 4 are counted. As both `cs_conj_nonpar 2` and `cs_conj_nonpar 3` are associated with negative weights, their occurrence is penalized, which makes the readings in Figures 8.1(a) and 8.1(b) less probable than the intended reading in Figure 8.1(c).

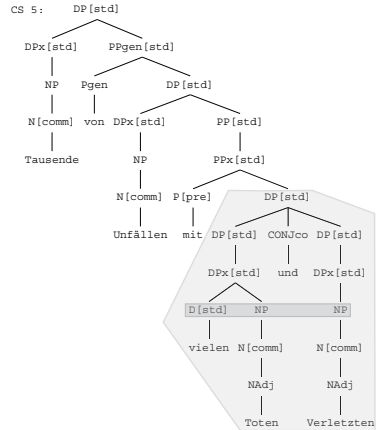
Another property template that captures c-structural characteristics of analyses is `cs_label`, which is parameterized for a c-structure category. An example of a property defined with the help of this template is `cs_label`

¹s4314

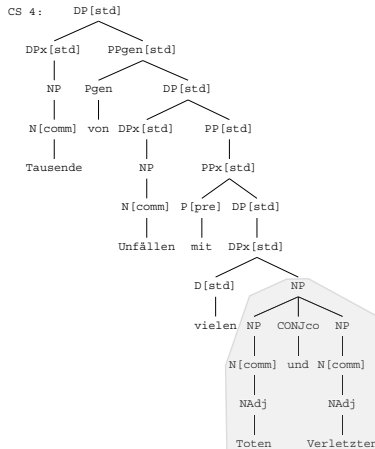
8.1 Properties based on XLE property templates



(a) evaluated as least probable due to `cs_conj_nonpar 2` and `cs_conj_nonpar 3`



(b) evaluated as less probable due to `cs_conj_nonpar 3`



(c) evaluated as probable

Figure 8.1: Competing c-structures for (8.1)

A first model

ADVP-VORF [std]. Properties based on the template `cs_label` simply count the number of c-structure nodes that are of the category passed as a parameter. `cs_label ADVP-VORF [std]` thus counts the number of ADVP-VORF [std] nodes in the c-structure of a given reading.

(8.2) is a sentence where `cs_label ADVP-VORF [std]` contributes to the identification of the intended analysis. Being associated with a negative weight, the property makes the c-structure shown in Figure 8.2(a), with one occurrence of `cs_label ADVP-VORF [std]`, less probable than the c-structure shown in Figure 8.2(b), where there is no ADVP-VORF [std] node.

- (8.2) Fünf Jahre nach der Vereinigung sieht er jedoch eine glänzende
Five years after the reunification sees he however a brilliant
Zukunft.
future.

‘Five years after the reunification, however, he sees a brilliant future.’²

A related property template is `cs_num_children`, which is also parameterized for a c-structure category. As its name indicates, it counts the number of nodes that are immediately dominated by a node of that category. An example of a property defined with the help of this template is `cs_num_children ADVP-VORF [std]`, which counts the number of children the ADVP-VORF [std] nodes of a parse’s c-structure have.

(8.3) is a sentence where `cs_num_children ADVP-VORF [std]` contributes to the identification of the intended reading, which is illustrated in Figure 8.3(c) (p. 130). Being associated with a negative weight, `cs_num_children ADVP-VORF [std]` causes parses without ADVP-VORF [std] nodes to be preferred over parses with ADVP-VORF [std] nodes, and among parses with the same number of ADVP-VORF [std] nodes, it picks the one where the fewest nodes are dominated by an ADVP-VORF [std] node. As the preference for parses without ADVP-VORF [std] nodes is also captured by `cs_label ADVP-VORF [std]`, the two properties are highly correlated.

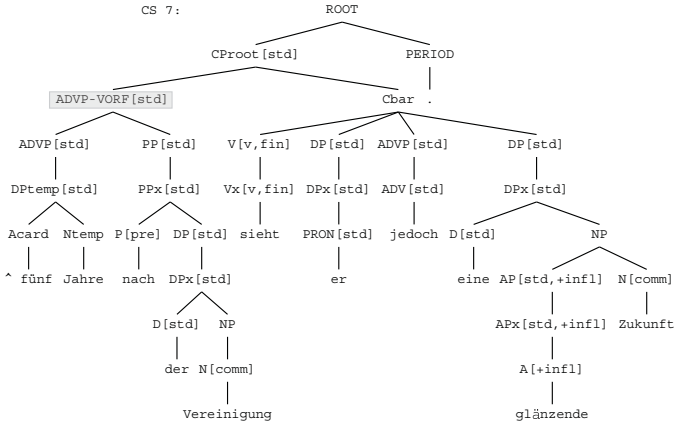
- (8.3) Auch vor dem Sprung ins dritte Jahrtausend ist das nicht
Also before the jump into the third millenium is that not
anders.
different.

‘This is not any different before the jump into the third millenium.’³

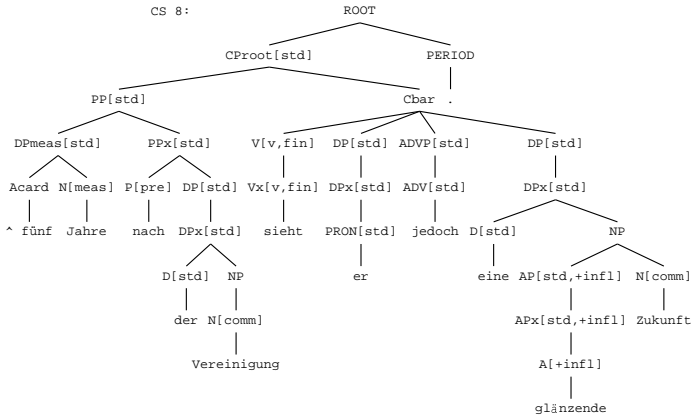
²s7170

³s32817

8.1 Properties based on XLE property templates



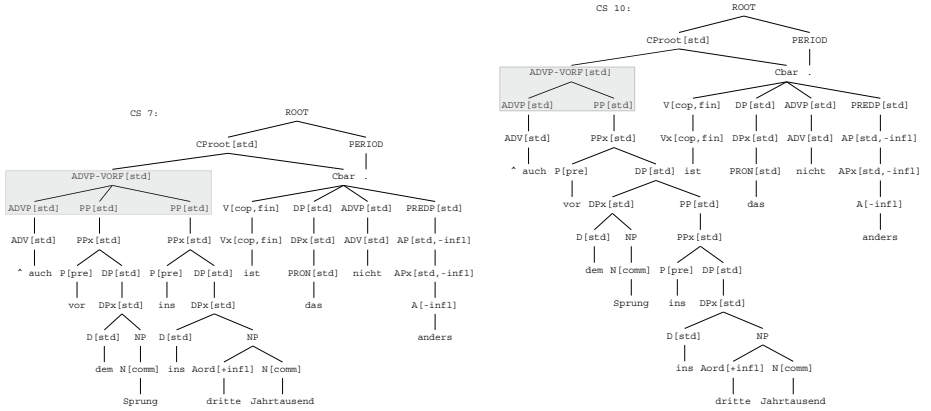
(a) evaluated as relatively improbable due to `cs_label ADVP-VORF[std]`



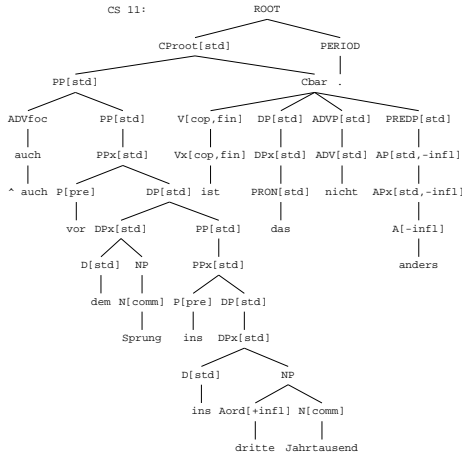
(b) evaluated as more probable

Figure 8.2: Competing c-structures for (8.2)

A first model



(a) evaluated as least probable due to `cs_num_children` (b) evaluated as less probable due to `cs_num_children` `ADVP-VORF[std]`



(c) evaluated as probable

Figure 8.3: Competing c-structures for (8.3)

8.1 Properties based on XLE property templates

Another c-structure-based property template is `cs_adjacent_label`.⁴ It takes two parameters, which have to be c-structure categories, and counts all nodes of the first category that immediately dominate a node of the second category. An example of a property based on this template is `cs_adjacent_label DPx[std] CONJco`, which counts the number of `DPx[std]` nodes that immediately dominate a `CONJco` node.

(8.4) is a sentence for which `cs_adjacent_label DPx[std] CONJco` contributes to the identification of the intended analysis. As the weight associated with this property is negative, the analysis illustrated in Figure 8.4(a) (p. 132), where `cs_adjacent_label DPx[std] CONJco` yields 1, is evaluated as less probable than the analysis illustrated in Figure 8.4(b) (also p. 132), where it yields 0.

- (8.4) Der 50. Jahrestag der Zwangsvereinigung von SPD und KPD
The 50th anniversary of the forced unification of SPD and KPD
zur SED steht im nächsten Jahr bevor.
into the SED is in the next year in store.

‘The 50th anniversary of the forced unification of SPD and KPD into the SED is in store for next year.’⁵

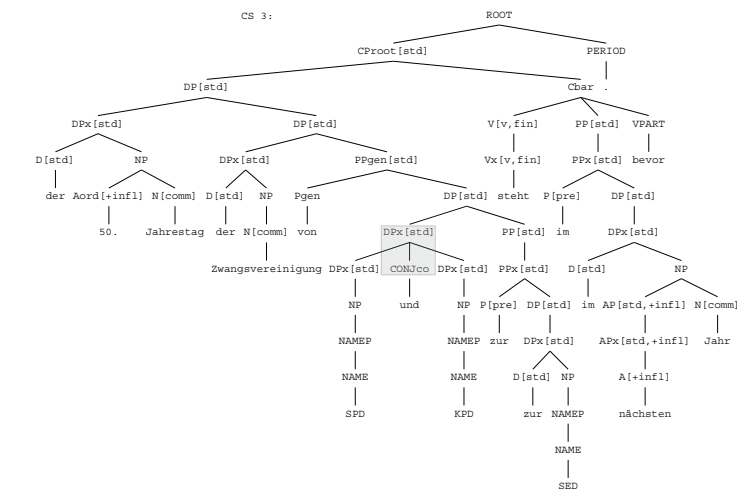
In addition to the template `cs_adjacent_label`, there is the template `cs_sub_label`, the latter being a non-local version of the former. Just like `cs_adjacent_label`, it is parameterized for two c-structure categories. The difference between the two templates then is that `cs_adjacent_label` only counts local subtrees where a node of the first c-structure category immediately dominates a node of the second c-structure category, whereas `cs_sub_label` counts subtrees where a node of the first c-structure category dominates a node of the second c-structure category at arbitrary depth. Since we find non-local properties difficult to relate to systematic linguistic preferences or dispreferences and given that there is an extremely high correlation between the local properties based on `cs_adjacent_label` and their counterparts based on `cs_sub_label`, we do not make use of the property template `cs_sub_label` in the final model.

Another non-local c-structure-based property template is `cs_embedded`, which is parameterized for a c-structure category c and a natural number n and counts nodes of category c that dominate n other nodes of category c , the dominance relation not needing to be immediate. Please note that, although the template `cs_embedded` looks reasonably different from `cs_sub_label`, these two templates give rise to properties that are equivalent if instantiated mechanically, namely properties of the types `cs_sub_label XP XP` and `cs_embedded XP 1`. In

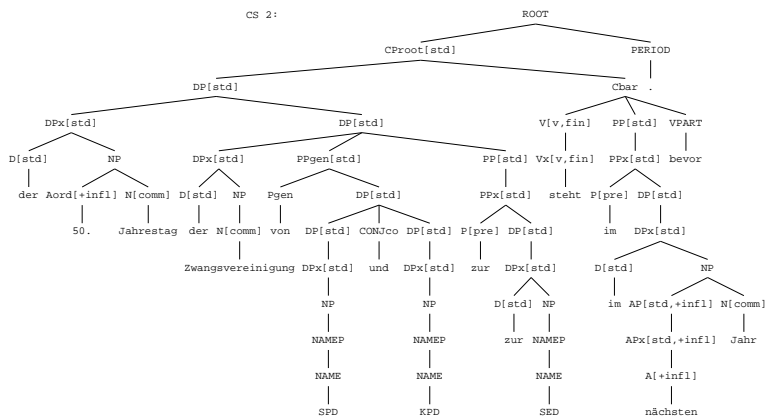
⁴Contrary to what the name that the developers of XLE’s stochastic disambiguation machinery chose for this template suggests, the corresponding features capture dominance relations, not adjacency relations.

⁵s35288

A first model



(a) evaluated as relatively improbable due to `cs_adjacent_label DPx[std] CONJco`



(b) evaluated as more probable

Figure 8.4: Competing c-structures for (8.4)

order to avoid this type of redundancy in the set of properties and on the basis of the non-local character of `cs_embedded`, we do not make use of properties based on this template in the final model.

8.1.2 F-structure-based properties

Besides c-structure-based property templates, XLE also provides a number of property templates that refer to f-structure characteristics. Many of them work similarly to their c-structure-based counterparts, but there are, of course, differences in the property templates that have to do with the differences between the representations.

`fs_attrs` is an f-structure-based property template that counts the number of occurrences of a given attribute in an f-structure. An example for a property based on `fs_attrs` is `fs_attrs FOCUS-INT`, which helps to discriminate between the two readings of (8.5) illustrated in Figures 8.5(a) and 8.5(b) (p. 134). Being associated with a positive weight, `fs_attrs FOCUS-INT` contributes to an analysis that contains the attribute `FOCUS-INT` being preferred over an alternative analysis that does not.

- (8.5) Alle Kollegen wissen noch aus der Wendezeit, was die Bevölkerung
 All colleagues know still from the change time what the population
 ihnen an Mißtrauen entgegenbrachte.
 them of mistrust showed.

‘All colleagues still remember from times of the change how much mistrust the population showed towards them./what the population showed towards them at mistrust.’⁶

`fs_attrs` is in a sense the f-structural equivalent of `cs_label`. Unlike `cs_label`, however, it can take a series of f-structure attributes as a parameter, not just one. The attributes are then treated disjunctively. As we do not have clear intuitions on how attributes should be grouped in order to form informative parameters for `fs_attrs`, we do not make use of this possibility so far, but rather instantiate the template with one f-structure attribute as a parameter at a time.

Another f-structure-based property template is `fs_attr_val`, which takes two parameters, namely an f-structure attribute and a potential value of this attribute. One instantiation of this template is `fs_attr_val PCASE gen`, which counts how often the atomic attribute `PCASE` has the value `gen`.

(8.6) is a sentence for which `fs_attr_val PCASE gen` contributes to the identification of the intended analysis. As the property is associated with a positive weight and it yields a count 0 for the reading illustrated in Figure 8.6(a) (p. 135) and a count 1 for the reading illustrated in Figure 8.6(b) (also p. 135), it contributes to making the latter more probable than the former.

⁶s4968

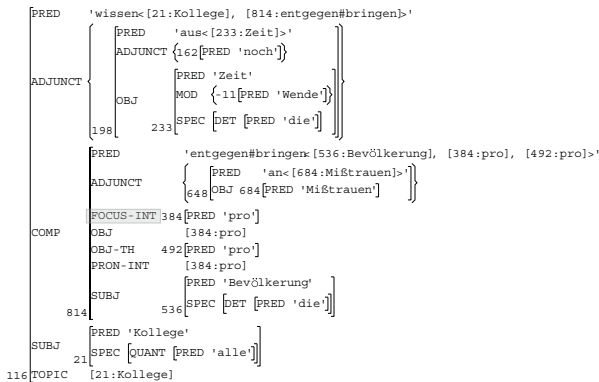
A first model

"Alle Kollegen wissen noch aus der Wendezeit, was ihnen die Bevölkerung an Mißtrauen entgegenbracht"



(a) evaluated as less probable

"Alle Kollegen wissen noch aus der Wendezeit, was ihnen die Bevölkerung an Mißtrauen entgegenbracht"



(b) evaluated as relatively probable due to *fs_attr*s FOCUS-INT

Figure 8.5: Competing f-structures for (8.5)

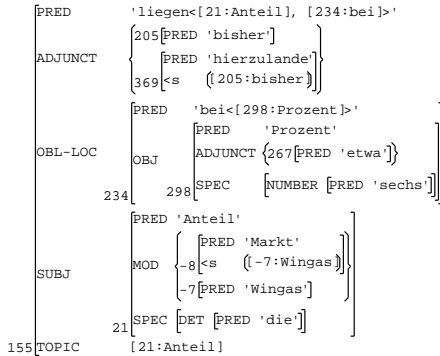
- (8.6) Der Chef von Gesamtmetall geht.
The boss of Gesamtmetall goes.

‘The boss of/from Gesamtmetall [employers’ association for the metal and electrical industry] will leave.’⁷

⁷s10092

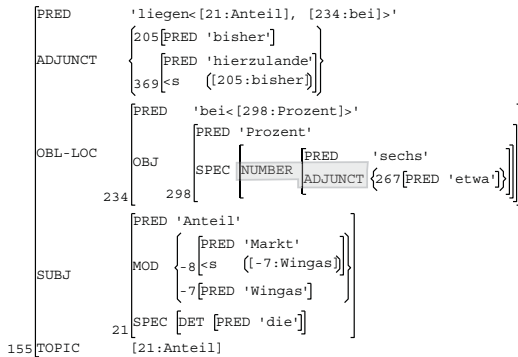
A first model

"Der Wingas-Marktanteil liegt bisher bei etwa sechs Prozent hierzulande



(a) evaluated as less probable

"Der Wingas-Marktanteil liegt bisher bei etwa sechs Prozent hierzulande



(b) evaluated as relatively probable due to fs_adj_attr's NUMBER ADJUNCT

Figure 8.7: Competing f-structures for (8.7)

8.1 Properties based on XLE property templates

[PRED	‘widersprechen(SUBJ, OBJ-TH, COMP)’]
[SUBJ	[...]]
[OBJ-TH	[...]]
[COMP	[...]]
[PASSIVE -]

[PRED	‘widersprechen(OBL-AG, OBJ-TH)SUBJ’]
[SUBJ	[...]]
[OBJ-TH	[...]]
[OBL-AG	[...]]
[PASSIVE +]

[PRED	‘widersprechen(OBJ-TH)SUBJ’]
[SUBJ	[...]]
[OBJ-TH	[...]]
[PASSIVE +]

(8.8) is a sentence whose intended reading is correctly identified thanks to this property. Being associated with a highly negative weight, the property contributes to making the reading illustrated in Figure 8.8(a) (p. 138) less probable than the intended reading, illustrated in Figure 8.8(b) (also p. 138).

- (8.8) Lafontaine widersprach der Darstellung, daß seine Wahl einen Lafontaine contradicted the statement that his election a Linksruck bedeute.
left shift means.

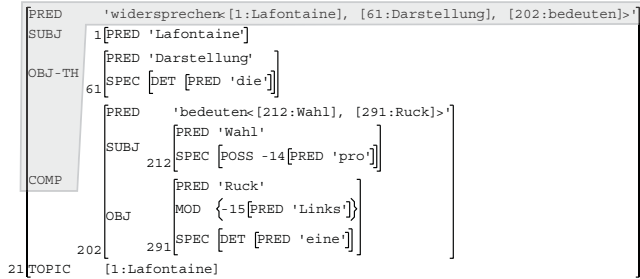
‘Lafontaine contradicted the statement (by saying) that his election meant a shift to the left.’⁹

Another lexicalized property template is *verb_arg*. As parameters, it takes the lemma of the frame-evoking element and an argument of this element. *verb_arg an#vertrauen OBJ-TH* is an example of a property based on this template. It counts the number of PREDs with the value *an#vertrauen* that subcategorize for an OBJ-TH. Other arguments of *an#vertrauen* may or may not occur. Properties based on *verb_arg* are used in the model presented in this chapter. In the final model, they are not used, however, since they are (almost) equivalent to properties of the DEP21 property family (see Subsection 9.1.8) that is used there.

⁹s33320

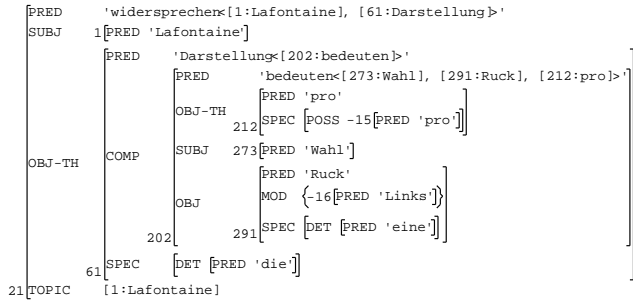
A first model

"Lafontaine widersprach der Darstellung, dass seine Wahl einen Linksruck bedeute



(a) evaluated as relatively improbable due to lex.subcat widersprechen SUBJ,OBJ-TH,COMP,PASSIVE=- ...

"Lafontaine widersprach der Darstellung, dass seine Wahl einen Linksruck bedeute



(b) evaluated as more probable

Figure 8.8: Competing f-structures for (8.8)

(8.9) is a sentence that can be correctly disambiguated with the help of verb_arg an#vertrauen OBJ-TH. As the property is associated with a positive weight, it contributes to making the reading illustrated in Figure 8.9(b) more probable than the reading illustrated in Figure 8.9(a).

- (8.9) [...] weil sich ein 47jähriger Mann der Polizei anvertraute.
 [...] because himself a 47-year-old man the police confided.
 '[...] because a 47-year-old man confided in the police./because a 47-year-old man of the police confided (in someone).'¹⁰

¹⁰s16182

8.2 Estimation of property weights

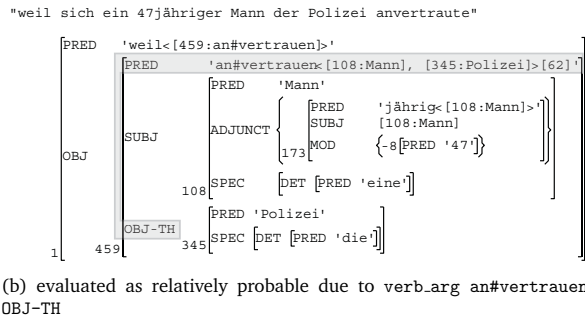
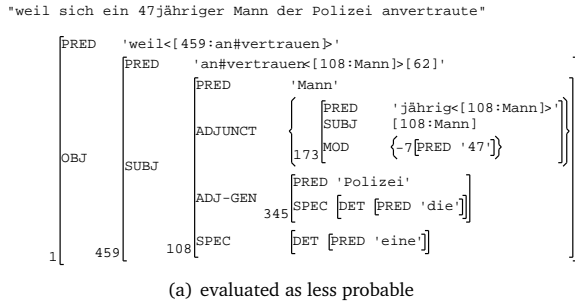


Figure 8.9: Competing f-structures for (8.9)

Further f-structure-based property templates that XLE provides, but which we do not make use of, are `fs_auntsubattr` and `fs_subattr`. The properties based on `fs_auntsubattr` are discarded because we have no intuition as to which instances of this property template would be linguistically relevant. Moreover, properties based on `fs_auntsubattr` are likely to be correlated highly with corresponding properties based on `fs_subattr`, which in turn correlate highly with properties based on `fs_adj_attr`, since `fs_subattr` is the non-local variant of `fs_adj_attr`.

8.2 Estimation of property weights

The weights associated with the properties just presented are estimated on the basis of the 8,881 pairs of packed c-/f-structure representations that constitute our training data. The training is carried out by means of the *cometc* software developed by Stefan Riezler, which is available as part of XLE. Instead of ap-

plying a Gaussian prior to the property weights, as it is done in Riezler et al. (2002), we employ the method of combined regularization and property selection presented in Riezler & Vasserman (2004) and discussed in Section 10.4. The best-performing training parameters, which, in this training scheme, are a regularization constant for gradient-based property selection and the number of properties to be added in each grafting step, are determined on our heldout set of 371 TiGer DB structures. The optimal settings turn out to be 2 for the regularization constant and 20 for the number of properties to be added in each grafting step.

8.3 Evaluation

The results achieved with the resulting log-linear model for disambiguation on our test set of 1,497 TiGer DB structures are shown in Table 8.1. We give the figures resulting from the best possible selection as the upper bound, the figures resulting from an arbitrary selection as the lower bound (or baseline) and the figures resulting from the selection by the model presented in this chapter. Alongside the F-scores achieved by the model, we additionally indicate the corresponding error reductions, error reduction being calculated as follows:

$$F_{\kappa} = \frac{F_{actual} - F_{lower}}{F_{upper} - F_{lower}}$$

Concerning the lower bound, we would like to stress that it is based on an arbitrary selection among the parses produced by the symbolic part of the grammar. An alternative lower bound, probably less prone to coincidences that may occur in this arbitrary selection, would involve averaging over all parses produced by the symbolic grammar. However, the implementation of this mode of evaluation in the software by Crouch et al. (2002) is problematic because, for the calculation of the overall precision, recall and F-score, the program sums over dependencies in all readings of all sentences without weighting them according to the number of readings in a parse. This gives a higher weight to more ambiguous sentences, for which the evaluation figures tend to be less favorable, than to less ambiguous sentences, so that the computed lower bound actually tends to be lower than a random selection among the parses produced by the symbolic grammar is expected to be. Often, the difference between evaluation figures based on an arbitrary selection and the ‘average’ figures computed by the evaluation software is of several points and the error reduction computed on the basis of these ‘average’ figures as a lower bound would appear to be much higher than they actually are. In order to correct this problematic aspect in the calculation of ‘average’ figures, it would be necessary to *average* over the dependency counts of each sentence instead of summing up the dependency counts in all its readings.

relation/feature	upper bound	template-based pr. F-score	error red.	lower bound
all	85.50	82.17	34.5	80.42
PREDs only	79.36	74.69	31.0	72.59
app (close apposition)	63	61	75	55
app_cl (appositive clause)	53	52	86	46
cc (comparative complement)	28	19	-29	21
cj (conjunct of coord.)	70	67	25	66
da (dative object)	67	62	58	55
det (determiner)	92	91	50	90
gl (genitive in spec. pos.)	89	88	75	85
gr (genitive attribute)	88	84	56	79
mo (modifier)	70	62	27	59
mod (non-head in compound)	94	89	29	87
name_mod (non-head in compl. name)	82	81	67	79
number (number as determiner)	83	81	33	80
oa (accusative object)	78	69	31	65
obj (arg. of prep. or conj.)	90	87	25	86
oc_fin (finite cl. obj.)	67	64	0	64
oc_inf (infinite cl. obj.)	83	82	0	82
op (prepositional obj.)	57	54	40	52
op_dir (directional argument)	30	23	13	22
op_loc (local argument)	59	49	29	45
pd (predicative argument)	62	59	25	58
pred_restr	92	84	38	79
quant (quantifying determiner)	70	68	33	67
rc (relative clause)	74	59	0	59
sb (subject)	76	71	38	68
sbp (logical subj. in pass. constr.)	68	61	46	55
case	87	83	50	79
comp_form (complementizer form)	74	74	100	72
coord_form (coordinating conj.)	86	86	100	85
degree	89	87	0	87
det_type (determiner type)	95	95	-	95
fut (future)	86	86	-	86
gend (gender)	92	89	40	87
mood	90	90	-	90
num (number)	91	89	50	87
pass_asp (passive aspect)	80	79	0	79
perf (perfect)	86	86	100	85
pers (person)	85	82	50	79
pron_form (pronoun form)	73	73	-	73
pron_type (pronoun type)	71	71	100	70
tense	92	91	0	91

Table 8.1: F-scores (in %) in the 1,497 TiGer DB examples of our test set

As for the evaluation figures in Table 8.1 (p. 141), we observe that the overall F-score as well as the F-score of most grammatical relations is improved with respect to the lower bound. For many of them, however, this improvement is small, in particular for core grammatical functions like *sb* (subject) and *oa* (accusative object) and for grammatical relations concerning potentially extraposed constituents like *rc* (relative clause). Moreover, we notice that the overall error reduction is a little bit lower than the error reduction of 36% that Riezler et al. (2002) report for English.¹¹

We conclude from these observations that the model is informative in the sense that it performs a parse selection that is significantly better than an arbitrary selection. But often it seems to be the case that the properties included in the model so far do not capture the information necessary to distinguish between intended and unintended analyses. Our next step is thus to provide the system with more (and more informative) properties, which will hopefully allow for better discrimination between intended and unintended analyses.

8.4 Summary

In this chapter, we have presented the properties that are used in the first log-linear model for the disambiguation of German LFG parses that we trained as well as the training scheme applied in the development of the model. Both with respect to the properties and with respect to the training scheme, the work documented in this chapter follows Riezler et al. (2002) and Riezler & Vasserman (2004). The evaluation of the model shows that the stochastic disambiguation machinery implemented in XLE can successfully be applied to the task of disambiguating German LFG parses. However, the error reduction of 34.5% achieved on German LFG parses is a little bit lower than the error reduction reported for the disambuation module of the English *ParGram* LFG.

¹¹In previous experiments, where we used a little bit less training data and where the hyperparameters were not adjusted as carefully, we only achieved error reductions of around 25%.

Chapter 9

Property design for the disambiguation of German LFG parses

Since many ambiguities in German LFG parses can actually hardly be captured by means of the properties based on the XLE property templates, we have developed additional properties for the disambiguation of German LFG parses, which are presented in this chapter. The strategy in designing these properties was to consult the typological literature on word order constraints in general and the linguistic literature on soft constraints that govern word order in German and related languages in particular. Finally, we also looked for inspiration in articles on similar disambiguation techniques, such as Malouf & van Noord (2004), van Noord (2006).

Most of the new properties make reference to information that is actually present in the analyses, but cannot be extracted directly with the help of the XLE property templates, but we also carried out experiments with properties that are based on external resources. These refer to information on humanness, animacy and ‘groupness’ that we obtain from *GermaNet* and to auxiliary distributions based on data from the *Gramotron* activities at the Institute of Natural Language Processing of the University of Stuttgart (Schulte im Walde 2003a,b).

9.1 Additional properties based on grammar-internal information

The majority of the additional properties employed in the final log-linear model refer to information that is actually present in the packed c-/f-structure representations resulting from parsing texts with the German *ParGram* LFG, but that

cannot be extracted from the parses with the help of the XLE property templates. Many of these properties refer to the c-structure and f-structure parts of the representations simultaneously, whereas properties based on the XLE property templates refer either only to the c-structure or only to the f-structure. Other additional properties extract only f-structural information, but refer to configurations that are more complex than what can be captured with the help of the f-structural XLE property templates.

Technically, the new properties are introduced by means of the XLE term-rewriting system presented in the context of treebank conversion (Chapter 4). They thus take the form of transfer rules, although they actually do not alter the representations produced by the grammar, but check the representations for certain c-/f-structure configurations and record their findings in a form that allows for the extraction of the corresponding information by means of the hardwired property template `fs_attr_val`.

Just like the properties presented so far, most of the additional properties are built on property templates. Based on the template of which they are an instantiation, properties can be classified into property ‘families’. In the following, we present the additional property templates that we introduced and exemplify the purpose of each property family through one of its members.

9.1.1 Properties for resolving ambiguities concerning the dependency mod

In the German *ParGram* LFG, compounds are first treated in the finite-state morphology by being split into their components. Then their head is projected as the main PRED of the word and all non-heads are projected as PREDS into a set-valued feature MOD (for modifier). In cases where several decompositions are possible for a compound, both the head PRED and the PREDS in the MOD set can vary from one reading to another. In order to disambiguate this type of lexical ambiguity, we introduced two property families: `MOD_<lemma>` and `<lemma>_MOD_<lemma>`.

The list of possible instantiations of these two property templates (as well as of all other lexicalized property templates presented in this chapter) was established by *not* specifying the transfer rules that check for the corresponding configurations with respect to the PREDS involved. Instead, these PREDS are determined during transfer, and the corresponding property names are built on the fly according to a predefined pattern and encoded in the c-/f-structure representation that is being transferred. After transfer, the list of properties built with the help of these templates can then be obtained via pattern-matching on the representations augmented with these additional properties.

- `fs_attr_val ADD-PROP MOD_Stan` is an instantiation of the property type `MOD_<lemma>`. It counts the number of PRED features with the value *Stan*

9.1 Additional properties based on grammar-internal information

- (9.3) über den Handel mit Dokortiteln
 about the trade with Ph.D. titles
 ‘about the trade with Ph.D. titles’⁷

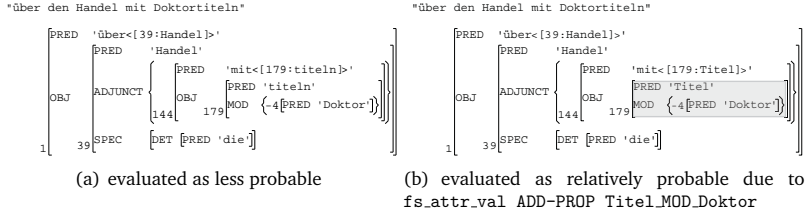


Figure 9.3: Competing f-structures for (9.3)

Although we introduced new properties of these two types, the evaluation figures for the dependency `mod` hardly improved. This is probably due to problems of data sparseness, but it is even less surprising if one considers that there is hardly any information about the correct decomposition of compounds in the TIGER Treebank and, hence, in our training data. Only from sentences where different tokenizations and/or morphological analyses lead to analyses that differ structurally can we expect to learn something about the decomposition of compounds.

9.1.2 Properties for resolving ambiguities due to case-ambiguous DPs on the basis of the nature of these DPs

The order of nominal and prepositional constituents is known to be relatively free in German. This freedom in word order, combined with frequent case syncretism, causes SUBJ-OBJ ambiguities, SUBJ-OBJ-TH ambiguities, SUBJ-XCOMP-PRED ambiguities, OBJ-OBJ-TH ambiguities etc. to be pervasive in the output of a symbolic LFG for German. Moreover, the assignment of the core grammatical functions obviously interacts with the assignment of other functions, such as ADJ-GEN (genitive attribute) and APP (close apposition). For high quality disambiguation of German LFG parses, it is thus crucial to provide properties that capture factors involved in the assignment of the core grammatical functions. Here, we present those properties that provide information about the nature of DPs that have a given grammatical function. They are, at least in part, inspired by the work on differential object and subject marking by Aissen (2003) and others, who observe that prototypical subjects are high

⁷s26404

on the definiteness scale, whereas prototypical objects are low on the definiteness scale, the scale being: Personal pronoun > Proper name > Definite DP > Indefinite specific DP > Non-specific DP.

German is a language that does not exhibit differential object marking. Nevertheless, we believe that the degree of definiteness of a DP may play a role in the identification of this DP as the SUBJ or OBJ of a clause. The general tendency that we expect to hold is that, among two case-ambiguous DPs, the one that is ranked higher on the the definiteness scale is more likely to be the SUBJ of the clause under consideration, and inversely, the DP that is ranked lower on the definiteness scale is likely to be the OBJ of the clause. It is, of course, impossible to read off all of these distinctions from the analyses produced by the grammar; this holds in particular for the distinction of indefinite specific DPs and non-specific DPs. Nevertheless, it is straightforward to identify personal pronouns, proper names, definite DPs and indefinite DPs. We thus introduce properties of the following types:

- `isCommon_<function>`
- `isDef_<function>`
- `isDemon_<function>`
- `isDies_<function>`
- `isIndef_<function>`
- `isPronPers_<function>`
- `isPronPersRecipRefl_<function>`
- `isPronRecip_<function>`
- `isPronRefl_<function>`
- `isProper_<function>`
- `isProperLocation_<function>`
- `isProperName_<function>`
- `isProperOrganization_<function>`
- `isQuant_<function>`

They are instantiated not only for SUBJ and OBJ, which are the grammatical functions generally investigated in the theoretical linguistic literature, but also for OBJ-TH and XCOMP-PRED. We thus leave it up to the learning procedure to

9.1 Additional properties based on grammar-internal information

determine on the basis of the training data whether similar tendencies exist for these two latter grammatical functions.

Let us consider some instantiations of these property (sub-)types.

- `fs_attr_val ADD-PROP isCommon_OBJ` counts the number of OBJs in an f-structure that are headed by common nouns. Whether a noun is common or not, can be read off its NTYPE NSYN feature.
- `fs_attr_val ADD-PROP isDef_OBJ` counts the number of OBJs that are definite DPs. A DP is considered definite if its f-structure embeds a SPEC DET DET-TYPE feature with the value *def*.⁸
- `fs_attr_val ADD-PROP isDemon_XCOMP-PRED` counts the number of XCOMP-PREDS that are demonstrative DPs. A DP is considered demonstrative if its f-structure embeds a SPEC DET DET-TYPE feature or a PRON-TYPE feature with the value *demon*. Its weight being highly negative, this property contributes to the correct determination of the f-structure in Figure 9.4(b) (p. 150) as the intended reading of (9.4).

(9.4) Ein perforierter Diskurs ist dieses Protokoll.
A perforated discourse/speech is this protocol.
'This protocol is a perforated discourse/speech.'⁹

- `fs_attr_val ADD-PROP isDies_OBJ` counts the number of OBJs that are just realized by the demonstrative pronoun *dies*. Being associated with a highly positive weight, this property contributes to the correct identification of the f-structure shown in Figure 9.5(b) (p. 150), with *dies* as the OBJ, as the intended reading of (9.5). *Dies*, which is neuter and often refers to events or statement, thus seems to behave differently from other demonstrative DPs in that it is rather an OBJ than a SUBJ although it is high in the definiteness scale.

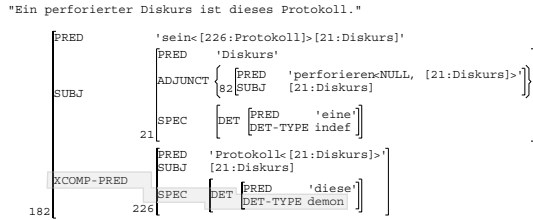
(9.5) Dies garantiert eine Vereinbarung [...].
This guarantees an agreement [...].
'An agreement [...] guarantees this./This guarantees an agreement [...].'¹⁰

⁸Since DPs that embed a possessor (either in the form of a possessive determiner or in the form of a DP-initial DP in the genitive) are definite as well, although they do not contain a SPEC DET DET-TYPE feature with the value *def*, we will extend the definition of 'definite' for future experiments.

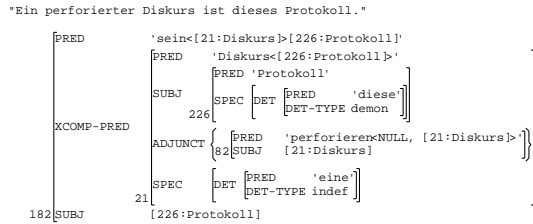
⁹s34738

¹⁰s38095

Property design for the disambiguation of German LFG parses

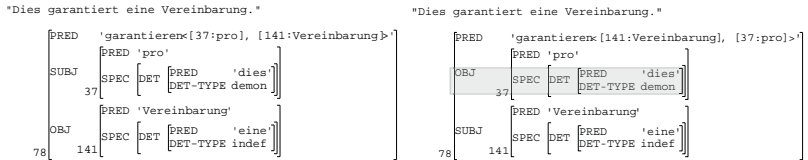


(a) evaluated as relatively improbable due to `fs_attr_val ADD-PROP isDemon_XCOMP-PRED`



(b) evaluated as more probable

Figure 9.4: Competing f-structures for (9.4)



(a) evaluated as less probable

(b) evaluated as relatively probable due to `fs_attr_val ADD-PROP isDies_OBJ`

Figure 9.5: Competing f-structures for (9.5)

- `fs_attr_val ADD-PROP isIndef_OBJ` counts the number of OBJs that are indefinite DPs. A DP is considered indefinite if its f-structure embeds a SPEC DET DET-TYPE feature with the value *indef*.
- `fs_attr_val ADD-PROP isPronPers_OBJ` counts the number of OBJs that are personal pronouns. A DP is a personal pronoun if its f-structure embeds a PRON-TYPE feature with the value *pers*.
- `fs_attr_val ADD-PROP isPronPersRecipRefl_OBJ` counts the number of OBJs that are personal pronouns, reciprocal pronouns or reflexive pro-

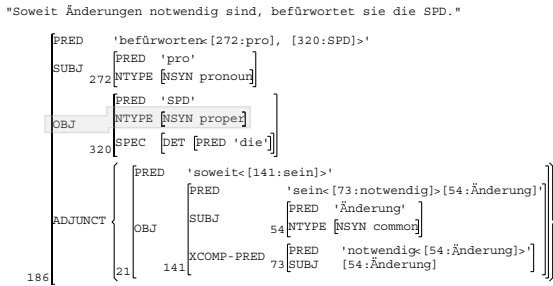
9.1 Additional properties based on grammar-internal information

nouns. A DP is considered as such if its f-structure embeds a PRON-TYPE feature with the value *pers*, *recip* or *refl*.

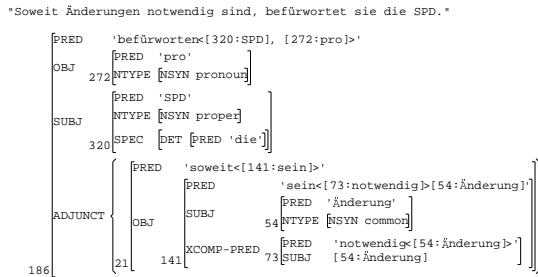
- `fs_attr_val ADD-PROP isPronRecip_OBJ` counts the number of OBJs that are reciprocal pronouns. A DP is a reciprocal pronoun if its f-structure embeds a PRON-TYPE feature with the value *recip*.
 - `fs_attr_val ADD-PROP isPronRefl_OBJ` counts the number of OBJs that are reflexive pronouns. A DP is a reflexive pronoun if its f-structure embeds a PRON-TYPE feature with the value *refl*.
 - `fs_attr_val ADD-PROP isProper_OBJ` counts the number of OBJs that are proper nouns. Whether a noun is proper or not, can be read off its NTYPE NSYN feature. Its weight being highly negative, this property contributes to the correct identification of the f-structure shown in Figure 9.6(b) (p. 152), with *die SPD* as the SUBJ rather than the OBJ, as the intended analysis of (9.6).
- (9.6) Soweit [...] Änderungen [...] notwendig seien, befürwortet sie
 As far as [...] changes [...] necessary are, approves them
 die SPD.
 the SPD.
 ‘As far as changes [...] are necessary [...], the SPD approves
 them/she approves the SPD.’¹¹
- `fs_attr_val ADD-PROP isProperLocation_OBJ` counts the number of OBJs that are proper nouns designating a location. Whether a proper noun designates a location is read off its NTYPE NSEM PROPER PROPER-TYPE feature.
 - `fs_attr_val ADD-PROP isProperName_OBJ` counts the number of OBJs that are proper nouns designating persons. Whether a proper noun designates a person is read off its NTYPE NSEM PROPER PROPER-TYPE feature.
 - `fs_attr_val ADD-PROP isProperOrganization_OBJ` counts the number of OBJs that are proper nouns designating organizations. Whether a proper noun designates an organization is read off its NTYPE NSEM PROPER PROPER-TYPE feature, just like for `isProperLocation_OBJ` and `isProperName_OBJ`.
 - `fs_attr_val ADD-PROP isQuant_OBJ` counts the number of OBJs that are indefinite pronouns or contain a quantifying determiner. Whether this is the case is read off the feature SPEC QUANT or SPEC AQUANT or the value *quant* of the feature PRON-TYPE in the sub-f-structure of the OBJ.

¹¹s33342

Property design for the disambiguation of German LFG parses



(a) evaluated as relatively improbable due to `fs_attr_val ADD-PROP isProper_OBJ`



(b) evaluated as more probable

Figure 9.6: Competing f-structures for (9.6)

9.1.3 Properties for resolving ambiguities due to case-ambiguous DPs on the basis of their relative linear order

Although the relative order of nominal and prepositional constituents is free in German, there are clear statistical tendencies towards a default order of nominal constituents. According to Uszkoreit (1987) (p. 24), following earlier work by Lenerz (1977) and others, “the unmarked order is SUBJ, IOBJ, DOBJ” or, translated to our terminology, SUBJ, OBJ-TH, OBJ.

We therefore introduced additional properties of the type `<function1>_precedes_<function2>`, instantiating this type for all combinations of the grammatical functions SUBJ, OBJ, OBJ-TH and XCOMP-PRED. An example instantiation of this property type is the following:

- `fs_attr_val ADD-PROP SUBJ_precedes_OBJ` counts the number of (sub-)

their weight. Just like for the different combinations of grammatical functions, we formulated properties that are intended to capture this information.

- `fs_attr_val ADD-PROP PRONOUN_precedes_FULLLDP` counts the number of combinations of the grammatical functions `SUBJ`, `OBJ`, `OBJ-TH` and `XCOMP-PRED` which are embedded in the same (sub-)f-structure and of which the first one is realized by a personal, reciprocal or reflexive pronoun and the second one is a full DP. Likewise, `FULLDP_precedes_PRONOUN` counts the number of combinations of these grammatical functions of which the first one is realized by a full DP and the second one is a personal, reciprocal or reflexive pronoun.

Both of these properties survive property selection (see Section 10.4) only rarely and, when they do, they are associated with very small weights. The reason for this is probably that the relative order of pronominal DPs and full DPs is given in the input string and only rarely do ambiguities arise that allow the two DPs considered to be interpreted at different levels in the f-structure. The two properties are thus likely to be pseudo-constant and can hardly contribute to disambiguation. We expect, however, that these properties are of high relevance in generation, since in that scenario, any order of pronominal and non-pronominal DPs is produced and, for realization ranking, it is important to capture information about their relative order.

- `fs_attr_val ADD-PROP DEF_precedes_NONDEF` counts the number of combinations of the grammatical functions `SUBJ`, `OBJ`, `OBJ-TH` and `XCOMP-PRED` which are embedded in the same (sub-)f-structure and of which the first one is realized by a definite DP and the second one is a non-definite DP. Likewise, `NONDEF_precedes_DEF` counts the number of combinations of these grammatical functions of which the first one is realized by a non-definite DP and the second one is a definite DP.

Like `PRONOUN_precedes_FULLLDP` and `FULLDP_precedes_PRONOUN`, these properties are nearly pseudo-constant in disambiguation, since the order of definite and non-definite DPs is obviously given in the input string and ambiguities that allow the two DPs considered to be interpreted at different levels in the f-structure are rare. As a consequence, these properties are only rarely associated with a significant weight for disambiguation. However, we expect them, again like `PRONOUN_precedes_FULLLDP` and `FULLDP_precedes_PRONOUN`, to be very relevant for realization ranking.

- `fs_attr_val DIFF-IN-TOKEN-LENGTH-BTW-SUBSEQUENT-DPsPPs %X` calculates the differences in token length between pairs of subsequent DPs and PPs that are daughters of VPx nodes that are projected to the same f-structure. These differences are then summed up.

9.1 Additional properties based on grammar-internal information

DIFF-IN-TOKEN-LENGTH-BTW-SUBSEQUENT-DPsPPs %X is intended to capture information about the relative weight of subsequent constituents dominated by a VPx chain, and thus to allow the system to learn that “light constituents precede heavy constituents” (Uszkoreit (1987), p. 24).

Although we expect DIFF-IN-TOKEN-LENGTH-BTW-SUBSEQUENT-DPsPPs %X to play a greater role in realization ranking, it does survive property selection with most parameter settings and is always associated with a positive property weight. It thus can contribute to the correct resolution of the PP attachment ambiguity in (9.8), since the value of the property is $(5 - 4) + (2 - 4) + (2 - 4) + (2 - 5) + (2 - 5) + (2 - 2) = -9$ for the c-structure illustrated in Figure 9.8(a) and $(5 - 4) + (4 - 4) + (4 - 5) = 0$ for the c-structure illustrated in Figure 9.8(b) (both p. 156).

- (9.8) dass die Staats- und Regierungschefs “aller Voraussicht
that the state and government heads “all foresight
nach” eine Sondererklärung zu Jugoslawien verabschieden
according-to” a special declaration to Yugoslavia adopt
würden
would
‘that the heads of states and governments would very probably adopt
a special declaration concerning Yugoslavia/to Yugoslavia’¹³

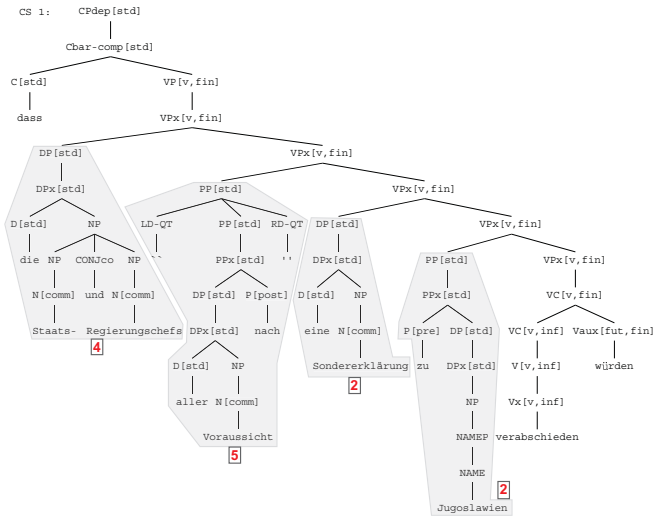
In addition to the unlexicalized properties just presented, we also introduced lexicalized properties:

- `fs_attr_val ADD-PROP AdjsWithDatObjAndGenObjTh_OBJ_precedes_`
OBJ-TH counts the number of PRED features with the value *bewußt*, *gewiß* or *sicher* that subcategorize for an OBJ and an OBJ-TH, with the OBJ preceding the OBJ-TH at the c-structure level. We introduced this property because, for the adjectives mentioned, the default order of OBJ and OBJ-TH is OBJ, OBJ-TH rather than the general default order OBJ-TH, OBJ. Likewise, `AdjsWithDatObjAndGenObjTh_OBJ-TH_precedes_OBJ` records the inverse order of the OBJ and OBJ-TH of these adjectives.

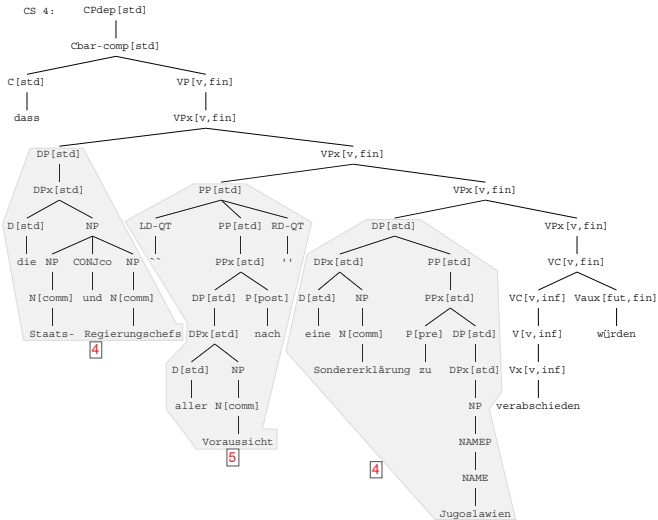
This type of property, which applies to a number of predicates that, according to the relevant literature, share their preference for a given order of their arguments, is intended to avoid problems of data sparseness that are likely to arise with properties that apply to individual predicates, while still making it possible to capture characteristic features of particular lexical elements.

¹³s2550

Property design for the disambiguation of German LFG parses



(a) evaluated as less probable



(b) evaluated as relatively probable due to fs.attr.val DIFF-IN-TOKEN-LENGTH-BTW-SUBSEQUENT-DPsPPs %X

Figure 9.8: Competing f-structures for (9.8)

9.1 Additional properties based on grammar-internal information

- `fs_attr_val ADD-PROP VerbsWithAccObjAndGenObjTh_OBJ_precedes_`
OBJ-TH counts the number of PRED features with the value *belehren, berauben, beschuldigen, entbinden, entblößen, entheben, verdächtigen, würdigen, zeihen* or *überführen* that subcategorize for an OBJ and an OBJ-TH, with the OBJ preceding the OBJ-TH at the c-structure level. We introduced this property because, for the verbs mentioned, the default order of the OBJ in the accusative and the OBJ-TH in the genitive is OBJ, OBJ-TH rather than the general default order OBJ-TH, OBJ. Likewise, `VerbsWithAccObjAndGenObjTh_OBJ-TH_precedes_OBJ` records the inverse order of the OBJ and OBJ-TH of these verbs.
- `fs_attr_val ADD-PROP VerbsWithAccObjAndAccObjTh_OBJ_precedes_`
OBJ-TH counts the number of PRED features with the value *ab#fragen, ab#hören, apostrophieren, bedeuten, fragen, kosten, lehren, lehren* or *überhören* that subcategorize for an OBJ and an OBJ-TH, with the OBJ preceding the OBJ-TH at the c-structure level. We introduced this property because, for the verbs mentioned, the default order of the OBJ in the accusative and the OBJ-TH in the accusative is OBJ, OBJ-TH rather than the general default order OBJ-TH, OBJ. Likewise, `VerbsWithAccObjAndAccObjTh_OBJ-TH_precedes_OBJ` records the inverse order of the OBJ and OBJ-TH of these verbs.
- `fs_attr_val ADD-PROP VerbsWithDatObjThAndNomSubj_OBJ-TH_precedes_`
SUBJ counts the number of PRED features with the value *begegnen, behagen, beklemmen, belieben, gefallen, gelingen, mißlingen, munden* or *schmecken* that subcategorize for a SUBJ and an OBJ-TH, with the SUBJ preceding the OBJ-TH at the c-structure level. We introduced this property because, for the verbs mentioned, the default order of SUBJ and OBJ-TH is OBJ-TH, SUBJ rather than the general default order SUBJ, OBJ-TH. Likewise, `VerbsWithDatObjThAndNomSubj_SUBJ_precedes_OBJ-TH` records the inverse order of the SUBJ and the OBJ-TH of these verbs. This property as well as the following property are built on insights from Frey (1993), where the author establishes regularities between the relative order of arguments in the thematic hierarchy and their relative linear order that are in fact stronger than the regularities between the relative order of arguments on the syntactic obliqueness scale and their relative linear order.
- `fs_attr_val ADD-PROP VerbsWithAccObjAndDatObjTh_OBJ_precedes_`
OBJ-TH counts the number of PRED features with the value *aus#liefern, aus#setzen, unterziehen, unterwerfen* or *zu#führen* that subcategorize for an OBJ and an OBJ-TH, with the OBJ preceding the OBJ-TH at the c-structure level. We introduced this property because, for the verbs mentioned, the default order of the OBJ in the accusative and the OBJ-TH

in the dative is OBJ, OBJ-TH rather than the general default order OBJ-TH, OBJ. Likewise, `VerbsWithAccObjAndDatObjTh_OBJ-TH_precedes_OBJ` records the inverse order of the OBJ and the OBJ-TH of these verbs.

- `fs_attr_val ADD-PROP begegnen_OBJ-TH_precedes_SUBJ` is an instantiation of the property type `<lemma>.<function1>_precedes_<function2>`, which is instantiated for all adjectives and verbs and all combinations of the functions SUBJ, OBJ, OBJ-TH and XCOMP-PRED. This property counts the number of PRED features with the value *begegnen* that subcategorize for a SUBJ and an OBJ-TH, with the OBJ-TH preceding the SUBJ. We introduced this type of property in order to allow the system to learn default orders of grammatical functions for specific subcategorizing elements that diverge from the general default order. However, we expect these properties to be of limited reliability and usefulness due to problems of data sparseness which result from them being fully specific to a given predicate.

9.1.4 Properties for the resolution of PP and ADVP attachment ambiguities

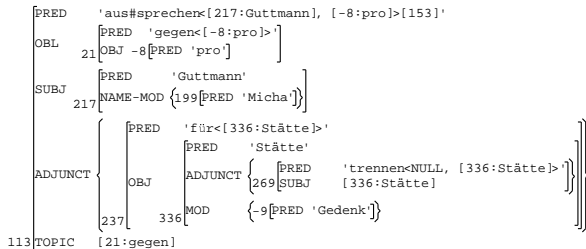
Other types of frequent ambiguities that are resolved only unsatisfactorily with the help of the XLE template-based properties are PP and ADVP attachment ambiguities. These attachment ambiguities are notoriously difficult to resolve; nevertheless, we managed to improve the learning rate for the dependency `mo`, which is the dependency that links most PPs and ADVPs to their heads, by introducing the following additional properties.

- `fs_attr_val ADD-PROP ADJPP_precedes_ARGPP` counts the number of argument, i.e. OBL, OBL-DIR, OBL-LOC and OBL-MANNER, PPs that are preceded by an ADJUNCT PP. Inversely, `fs_attr_val ADD-PROP ARGPP_precedes_ADJPP` counts the number of argument PPs that precede an ADJUNCT PP. The former property being associated with a positive and the latter with a negative weight, these properties contribute to the correct resolution of the ADJUNCT-OBL ambiguity in (9.9) by making the reading illustrated in Figure 9.9(b) more probable than the one in Figure 9.9(a).

9.1 Additional properties based on grammar-internal information

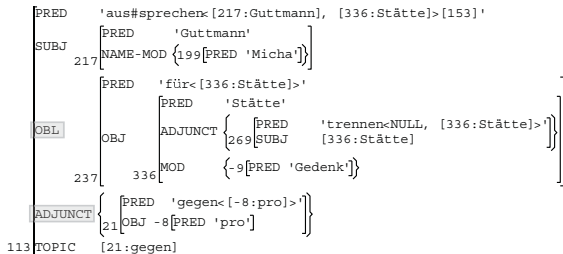
- (9.9) Dagegen sprach sich [...] Micha Guttman für
 Against this/In contrast spoke himself [...] Micha Guttman for
 getrennte Gedenkstätten aus.
 separate memorials out.
 ‘In contrast, [...] Michael Guttman argued for separate memori-
 als./For separate memorials, [...] Michael Guttman argued against
 this.’¹⁴

"Dagegen sprach sich Micha Guttman für getrennte Gedenkstätten aus."



(a) evaluated as less probable

"Dagegen sprach sich Micha Guttman für getrennte Gedenkstätten aus."



(b) evaluated as relatively probable due to `fs_attr_val ADD-PROP ADJPP_precedes_ARGPP`

Figure 9.9: Competing f-structures for (9.9)

- `fs_attr_val ADD-PROP DEFARG_precedes_ADJUNCT` counts the number of combinations of a SUBJ, an OBJ or an OBJ-TH that is a definite DP and an ADJUNCT PP or ADVP that is preceded by this DP. Inversely, `ADJUNCT_precedes_DEFARG` counts the number of combinations of a SUBJ, an OBJ or an OBJ-TH that is definite and an ADJUNCT PP or ADVP that precedes this DP. These properties are intended to capture the tendency

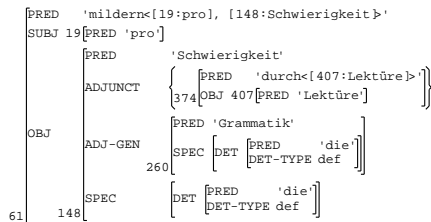
¹⁴s2092

of definite arguments to be placed to the left of ADJUNCTS, stated in many manuals and reference works for learners of German as a foreign language, e.g. Helbig & Buscha (2001).

Although it is a property that we expect to play a greater role in realization ranking, this property is associated with a clearly positive weight. Therefore, it contributes to the correct disambiguation of (9.10). The analysis displayed in Figure 9.10(a) has no value for the property, while the analysis in Figure 9.10(b) has a positive value, so that the latter is correctly determined as being more probable.

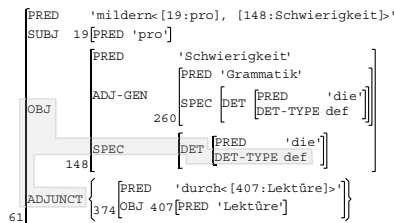
- (9.10) Ich werde [...] die Schwierigkeit der Grammatik durch
 I will [...] the difficulty the-GEN grammar through
 Lektüre [...] mildern [...]
 lecture [...] alleviate [...]
 'I will alleviate the difficulty of grammar by means of lectures [...]'¹⁵

"Ich werde die Schwierigkeit der Grammatik durch Lektüre mildern



(a) evaluated as less probable

"Ich werde die Schwierigkeit der Grammatik durch Lektüre mildern



(b) evaluated as relatively probable due to fs_attr_val
 ADD-PROP DEFARG_precedes_ADJUNCT

Figure 9.10: Competing f-structures for (9.10)

¹⁵s48637

9.1 Additional properties based on grammar-internal information

- `fs_attr_val ADD-PROP NONDEFARG_precedes_ADJUNCT` counts the number of combinations of a SUBJ, an OBJ or an OBJ-TH that is a non-definite DP and a PP or ADVP ADJUNCT that is preceded by this DP. Inversely, `ADJUNCT_precedes_NONDEFARG` counts the number combinations of a SUBJ, an OBJ or an OBJ-TH that is definite and a PP or ADVP ADJUNCT that precedes this DP. Just like `DEFARG_precedes_ADJUNCT` and `ADJUNCT_precedes_DEFARG`, these properties are intended to capture the tendency of nominal arguments to be placed to the left or to the right of ADJUNCT PP or ADVPs, depending on whether they are definite or non-definite.

Another way in which we hope to improve the resolution of ADJUNCT PP attachment ambiguities is by capturing information about their relative linear order in relation to some other feature. A good deal of linguistic work has gone into understanding what determines this relative order, most of the authors concluding that it is the meaning of a PP that determines its placement. For German, Helbig & Buscha (2001), for instance, state that temporal and causal ADJUNCTS precede local and modal ADJUNCTS. Given that many prepositions can have several of these meanings, depending on their argument, and that it would be very costly (both in resources and in computation) to determine the exact meaning of each ADJUNCT PP, we decided to approximate the ideal solution by defining properties that check whether ADJUNCT PPs that are part of the same (sub-)f-structure and that are headed by certain prepositions precede each other. We thus introduce properties of the three following types: `AADJUNCT_<preposition1>_precedes_<preposition2>`, `NADJUNCT_<preposition1>_precedes_<preposition2>` and `VADJUNCT_<preposition1>_precedes_<preposition2>`. We explain these now with the help of an instantiation of each type.

- `fs_attr_val ADD-PROP AADJUNCT_von_precedes_zu` counts the number of combinations of ADJUNCT PPs which are part of the same set, which are embedded into an f-structure that has a feature `ATYPE` and of which the first PP (considering linear order) is headed by the preposition *von* and the second PP, by the preposition *zu*.
- `fs_attr_val ADD-PROP NADJUNCT_aus_precedes_auf` counts the number of combinations of ADJUNCT PPs which are part of the same set, which are embedded into an f-structure that has a feature `NTYPE` and of which the first PP (considering linear order) is headed by the preposition *aus* and the second PP, by the preposition *auf*.
- `fs_attr_val ADD-PROP VADJUNCT_von_precedes_bis` counts the number of combinations of ADJUNCT PPs which are part of the same set, which are embedded into an f-structure that has a feature `VTYPE` and of which

the first PP (considering linear order) is headed by the preposition *von* and the second PP, by the preposition *bis*.

This property is associated with a highly positive weight, so that it contributes to the correct resolution of the PP attachment ambiguity in (9.11). The alternative analyses of this sentence are given in Figure 9.11.

- (9.11) Dieser beteiligt sich von der Vorbereitung bis zur
This participates himself from the preparation until to the
Ausführung an den Überwachungsflügen.
execution at the surveillance flights.
‘This person participates in the surveillance flights from the preparation until the execution.’¹⁶

9.1.5 Properties for the resolution of GEND and NUM ambiguities in XCOMP-PREDS

- `fs.attr_val GEND.SUBJ_XCOMP-PRED.UNIFIABLE %X` captures whether, in an f-structure that contains both a SUBJ and an XCOMP-PRED, the values of the GEND features in the SUBJ and the XCOMP-PRED are unifiable. If they are, the value of the property is incremented by one. If they are not, the value is decremented by one. The motivation for this property is to favor analyses where the SUBJ and the XCOMP-PRED agree in gender. However, the property ultimately often has a different effect.

In (9.12), for instance, it favors an analysis that involves an XCOMP-PRED (see Figure 9.12(b) on p. 164) over the competing analysis without an XCOMP-PRED (see Figure 9.12(a), also on p. 164). In this case, this effect is desired, but we doubt whether this is always the case.

- (9.12) Noch scheint die Ausreise einfach.
Still seems/shines the departure simple.
‘Leaving the country still seems to be simple./still shines simply.’¹⁷

- `fs.attr_val NUM.SUBJ_XCOMP-PRED.UNIFIABLE %X` is motivated and defined analogously, with the difference that it checks whether the values of the NUM features in the SUBJ and the XCOMP-PRED of a given (sub-)f-structure are unifiable.

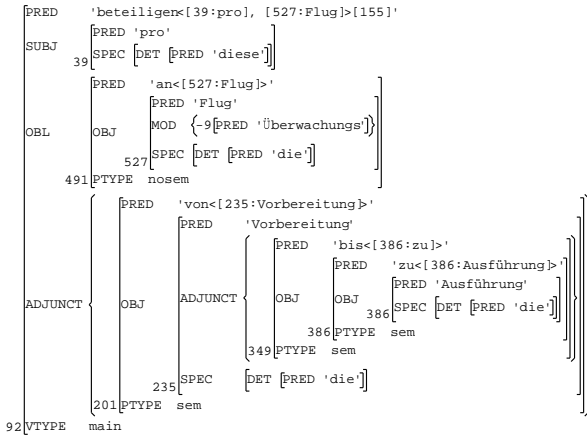
(9.13) exemplifies nicely the kind of ambiguity that is intended to be captured by this and the previous property. As we can see in the corresponding f-structures in Figure 9.13 (p. 165), *Kommunikationsmittel* can

¹⁶s35891

¹⁷s48239

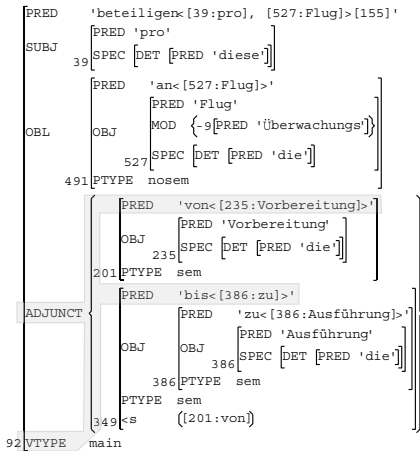
9.1 Additional properties based on grammar-internal information

"Dieser beteiligt sich von der Vorbereitung bis zur Ausführung an den Überwachungsflüger



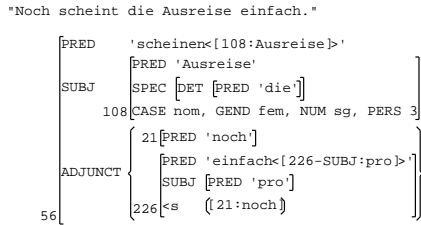
(a) evaluated as less probable

"Dieser beteiligt sich von der Vorbereitung bis zur Ausführung an den Überwachungsflüger

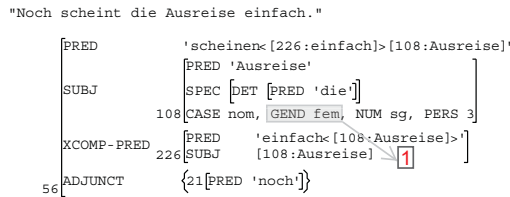


(b) evaluated as relatively probable due to fs_attr_val ADD-PROP VADJUNCT.von_precedes_bis

Figure 9.11: Competing f-structures for (9.11)



(a) evaluated as less probable



(b) evaluated as relatively probable due to fs_attr_val GEND_SUBJ_XCOMP-PRED_UNIFIABLE %X

Figure 9.12: Competing f-structures for (9.12)

be both singular and plural. A model where the property has a positive weight would then correctly prefer the analysis where *Kommunikationsmittel* agrees in number with the SUBJ in the plural over the competing analysis where it is analyzed as being singular.

- (9.13) Aktionen sind Kommunikationsmittel.
actions are communication means.
'Actions are means of communication.'¹⁸

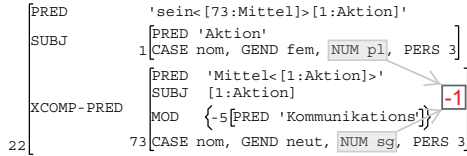
9.1.6 Properties for the resolution of attachment ambiguities concerning extraposed ADJ-RELS, APP-CLAUSES, COMPS and VCOMPS

Still another relatively frequent type of ambiguity is due to the extraposition of relative clauses (ADJ-RELS), argument clauses (COMPS, APP-CLAUSES) and infinitives (VCOMPS, APP-CLAUSES). At the c-structure level, these constituents are not attached to the constituent they depend on from a semantic point of

¹⁸s6854

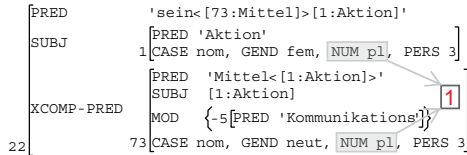
9.1 Additional properties based on grammar-internal information

"Aktionen sind Kommunikationsmittel."



(a) evaluated as less probable

"Aktionen sind Kommunikationsmittel."



(b) evaluated as relatively probable due to `fs_attr_val NUM_SUBJ_XCOMP-PRED_UNIFIABLE %X`

Figure 9.13: Competing f-structures for (9.13)

view, this latter dependency being established via a functional uncertainty equation. As is to be expected, these functional uncertainty equations can often be instantiated in several ways, so that the resulting ambiguities need to be resolved.

One factor that has been shown to play a role for the determination of the correct attachment of such an extraposed constituent is the distance between this constituent and the element that it modifies or is subcategorized by. We therefore introduced the properties `fs_attr_val DISTANCE-TO-ANTECEDENT %X`, `fs_attr_val DISTANCE-TO-APP-CLAUSE-GOVERNOR %X`, `fs_attr_val DISTANCE-TO-COMP-GOVERNOR %X` and `fs_attr_val DISTANCE-TO-VCOMP-GOVERNOR %X`.

Let us look at `fs_attr_val DISTANCE-TO-ANTECEDENT %X` in more detail. It computes the distance in characters between each relative pronoun and its antecedent in a parse and then sums up the distances.¹⁹ Its weight being negative, it prefers shorter distances over longer distances and thus contributes to the correct attachment of the relative clause in (9.14) to *Autoversicherung*, which is the closest noun that agrees in number and gender with the relative pronoun *die*. This reading as well as the competing analysis are illustrated in Figure 9.14 (p. 166).

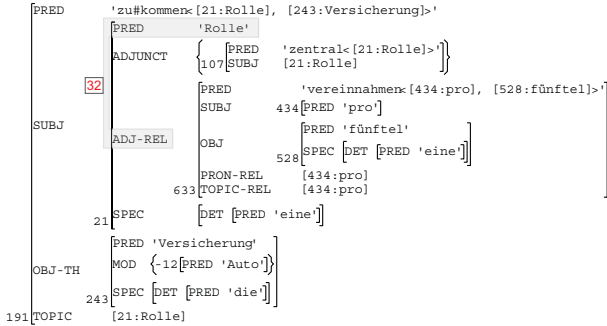
¹⁹It might be better to define the distance in terms of words rather than characters; this may eventually be changed.

Property design for the disambiguation of German LFG parses

- (9.14) Eine zentrale Rolle [...] kommt der Autoversicherung zu, die ein Fünftel [...] vereinnahmt.
 A central role [...] comes the car insurance to, which a fifth [...] receives.

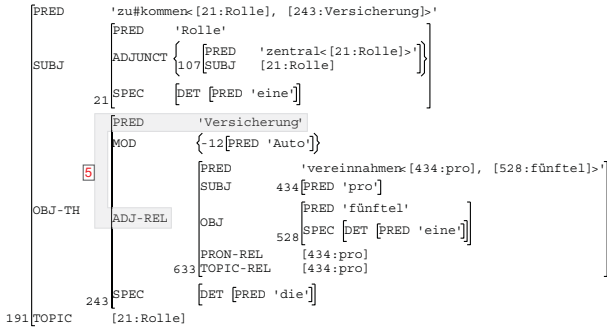
'There is a central role for the car insurance, which receives a fifth [...].'²⁰

"Eine zentrale Rolle kommt der Autoversicherung zu, die ein Fünftel vereinnahmt."



(a) evaluated as relatively improbable due to fs_attr_val DISTANCE-TO-ANTECEDENT %X

"Eine zentrale Rolle kommt der Autoversicherung zu, die ein Fünftel vereinnahmt."



(b) evaluated as more probable

Figure 9.14: Competing f-structures for (9.14)

The properties fs_attr_val DISTANCE-TO-APP-CLAUSE-GOVERNOR %X, fs_attr_val DISTANCE-TO-COMP-GOVERNOR %X and fs_attr_val DISTANCE-TO-VCOMP-GOVERNOR %X are defined similarly.

²⁰s27542

9.1 Additional properties based on grammar-internal information

However, surface distance does not seem to be the only factor involved in determining the correct antecedent or governor of an extraposed constituent. An example where the intended antecedent is not the closest noun that agrees in gender and number with the relative pronoun is given in (9.15).

- (9.15) [...] will er eine Fabrik zur Herstellung von Kühlanlagen
[...] wants he a factory to the production of cooling devices
besichtigen, die auch FCKW-freie Kühlschränke baut.
visit, which also CFC-free refrigerators builds.
'He [...] wants to visit a factory for the production of cooling devices that
also builds CFC-free refrigerators.'²¹

Our intuition is that the level of functional embedding also plays a role in this kind of long-distance dependencies, and that in certain sentences, it may counteract and even override the effect of distance. We therefore introduced the following three property types, which are all instantiated for ADJ-REL, APP-CLAUSE, COMP and VCOMP:

- `fs_attr_val DEPTH-OF-DP-PATH_<dependency> %X` encodes the depth at which the grammar template DP-PATH that is part of the functional uncertainty annotation of this kind of extraposed constituents is instantiated in the analysis under consideration.
- `fs_attr_val DEPTH-OF-VP-PATH_<dependency> %X` encodes the depth at which the grammar template VP-PATH that is part of the functional uncertainty annotation of this kind of extraposed constituents is instantiated in the analysis under consideration.
- `fs_attr_val DEPTH-OF-PATH_<dependency> %X`, finally, records the sum of the two previous properties, i.e. the depth of the entire functional uncertainty path, as it is instantiated in the analysis under consideration.

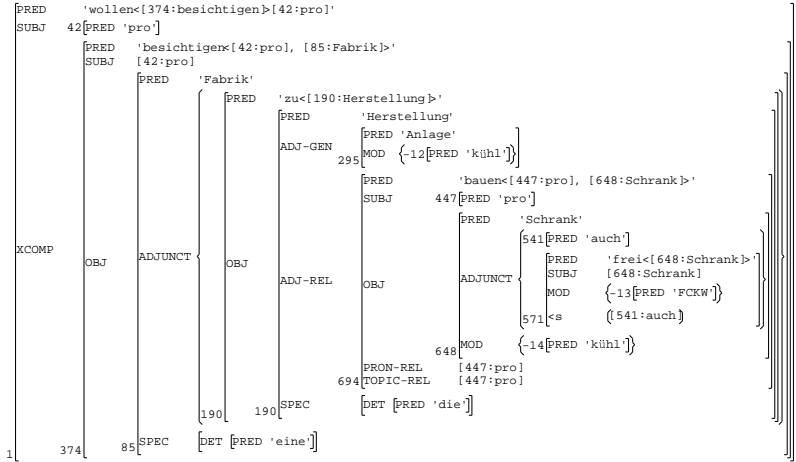
Furthermore, we introduced properties for each of the four dependencies potentially involving extraposition that record the last three levels of embedding of the corresponding functional uncertainty path as it is instantiated in the analysis under consideration. `fs_attr_val ADD-PROP PATH_ADJ-REL_XCOMP_OBJ_NULL`, which, by being associated with a positive weight, contributes to the correct disambiguation of the relative clause attachment ambiguity in (9.15), is such a property.²² It makes the analysis illustrated in Figure 9.15(b) more probable than the one in Figure 9.15(a) (both p. 168).

²¹s25979

²²For technical reasons, all properties of this kind provide room for three levels of embedding. For paths shorter than three levels, the remaining levels are filled with the string NULL.

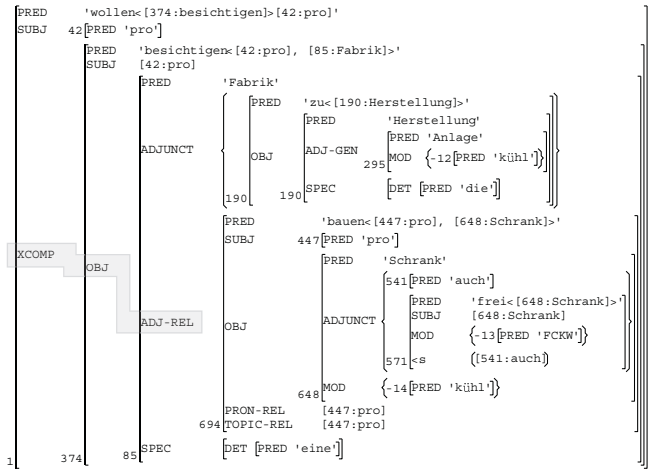
Property design for the disambiguation of German LFG parses

"will er eine Fabrik zur Herstellung von K hlanlagen besichtigen, die auch FCKW-freie K hlchr nke baut,"



(a) evaluated as less probable

"will er eine Fabrik zur Herstellung von K hlanlagen besichtigen, die auch FCKW-freie K hlchr nke baut"



(b) evaluated as relatively probable due to fs_attr_val ADD-PROP PATH_ADJ-REL_XCOMP_OBJ_NULL

Figure 9.15: Competing f-structures for (9.15)

9.1.7 Properties for the resolution of part-of-speech ambiguities

Inspired by Malouf & van Noord (2004) and van Noord (2006), we introduced lexicalized properties for the resolution of part-of-speech ambiguities. They are of the type $F2.<PoS>.<lemma>$, the possible parts-of-speech being *ATYPE*, *ADV-TYPE*, *DET-TYPE*, *NUMBER-TYPE*, *P-TYPE*, *V-TYPE*, *common* and *proper*. The first six of these parts-of-speech correspond to f-structure features, the last two to possible values of the f-structure feature *N-TYPE*, so that all of these can easily be read off an f-structure. We present now two instantiations of this type of property.

- `fs_attr_val ADD-PROP F2.common.Thüringer` counts the number of (sub-)f-structures whose *PRED* feature has the value *Thüringer* and whose *N-TYPE* *NSYN* feature has the value *common*. It helps to disambiguate between the adjectival reading of *Thüringer*, illustrated in Figure 9.16(b), and its nominal reading, illustrated in Figure 9.16(a), which are both possible for sentences like (9.16).

(9.16) Der Konkursverwalter der Thüringer Docter-Optic-Gruppe
 The liquidator the-GEN Thuringian(s) Docter Optic Group
 ‘The liquidator of the Thuringian Docter Optic Group’²³

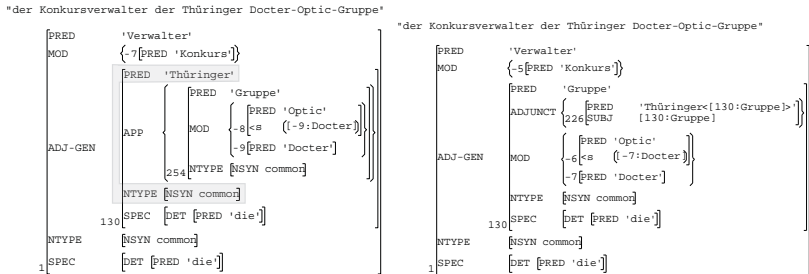


Figure 9.16: Competing f-structures for (9.16)

Thüringer is part of a series of forms that can be both an adjective denoting origin or a noun denoting origin. Since this type of adjective is

²³s23765

always upper-cased and invariable and the noun changes its form only in the genitive singular and the dative plural, the chance of the forms of this series appearing in contexts where their part-of-speech is not categorically disambiguated by the context is relatively high. It is thus not surprising that other properties of this type, such as `F2_common_Köln` or `F2_common_München`, turn out to be relevant for disambiguation as well. However, we have made this observation only after training the weights of our set of properties, so that it is only in a future series of experiments that we can take advantage of this observation by adding a more mildly lexicalized property related to `fs_attr_val ADD-PROP F2_common_Thüringer` that would be based on the entire list of adjectives and nouns denoting origin that we know.

- `fs_attr_val ADD-PROP F2_proper_Powell` counts the number of (sub-) f-structures whose PRED feature has the value *Powell* and whose NTYPE NSYN feature has the value *proper*. Although it was designed to disambiguate part-of-speech ambiguities, it does not serve this purpose. Instead, it helps to disambiguate between competing solutions proposed for the form *Powells*, like in (9.17). Since *Powells* does not receive an analysis from the morphology, it is analyzed by the finite-state guesser, which proposes the following solutions:

```
Powells
Powells +NPROP .NoGend .NGDA .SP .Guessed
Powell +NPROP .NoGend .Gen .Sg .Guessed
```

- (9.17) die schwarze Hautfarbe Powells
the black skin color Powell-GEN/Powells
'Powell's black skin color'²⁴

Interestingly, `F2_proper_Powell` does thus not end up resolving part-of-speech ambiguities, but helps to determine that *Powells* is a form of *Powell*, as in the analysis illustrated in Figure 9.17(b), rather than of *Powells*, as in the f-structure shown in Figure 9.17(a).

²⁴s15435

9.1 Additional properties based on grammar-internal information

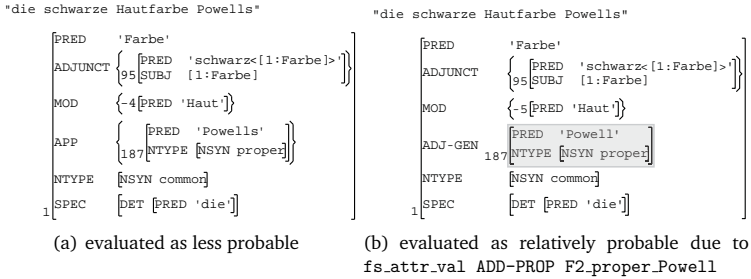
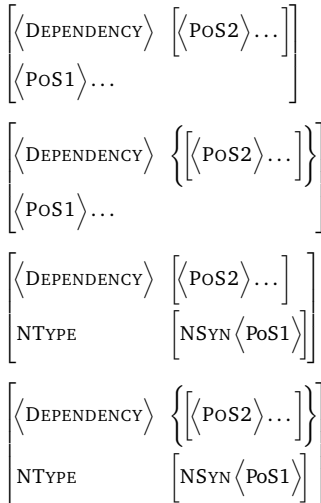


Figure 9.17: Competing f-structures for (9.17)

9.1.8 Properties capturing dependencies

Finally, and again inspired by Malouf & van Noord (2004) and van Noord (2006), we have introduced properties that capture dependencies. The first type, `DEP11-<PoS1>-<dependency>-<PoS2>` is not lexicalized; it just counts the number of f-structure snippets that match one of the following configurations:



Property design for the disambiguation of German LFG parses

$$\begin{array}{c}
 \left[\begin{array}{l} \langle \text{DEPENDENCY} \rangle \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \\ \langle \text{PoS1} \rangle \dots \end{array} \right] \\
 \\
 \left[\begin{array}{l} \langle \text{DEPENDENCY} \rangle \left\{ \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \right\} \\ \langle \text{PoS1} \rangle \dots \end{array} \right] \\
 \\
 \left[\begin{array}{l} \langle \text{DEPENDENCY} \rangle \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \\ \text{NTYPE} \left[\text{NSYN} \langle \text{PoS1} \rangle \right] \end{array} \right] \\
 \\
 \left[\begin{array}{l} \langle \text{DEPENDENCY} \rangle \left\{ \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \right\} \\ \text{NTYPE} \left[\text{NSYN} \langle \text{PoS1} \rangle \right] \end{array} \right] \\
 \\
 \left[\begin{array}{l} \text{SPEC} \left[\langle \text{DEPENDENCY} \rangle \left[\langle \text{PoS2} \rangle \dots \right] \right] \\ \langle \text{PoS1} \rangle \dots \end{array} \right] \\
 \\
 \left[\begin{array}{l} \text{SPEC} \left[\langle \text{DEPENDENCY} \rangle \left\{ \left[\langle \text{PoS2} \rangle \dots \right] \right\} \right] \\ \langle \text{PoS1} \rangle \dots \end{array} \right] \\
 \\
 \left[\begin{array}{l} \text{SPEC} \left[\langle \text{DEPENDENCY} \rangle \left[\langle \text{PoS2} \rangle \dots \right] \right] \\ \text{NTYPE} \left[\text{NSYN} \langle \text{PoS1} \rangle \right] \end{array} \right] \\
 \\
 \left[\begin{array}{l} \text{SPEC} \left[\langle \text{DEPENDENCY} \rangle \left\{ \left[\langle \text{PoS2} \rangle \dots \right] \right\} \right] \\ \text{NTYPE} \left[\text{NSYN} \langle \text{PoS1} \rangle \right] \end{array} \right]
 \end{array}$$

9.1 Additional properties based on grammar-internal information

$$\left[\begin{array}{l} \text{SPEC} \quad \left[\langle \text{DEPENDENCY} \rangle \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \right] \\ \langle \text{PoS1} \rangle \dots \end{array} \right]$$

$$\left[\begin{array}{l} \text{SPEC} \quad \left[\langle \text{DEPENDENCY} \rangle \left\{ \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \right\} \right] \\ \langle \text{PoS1} \rangle \dots \end{array} \right]$$

$$\left[\begin{array}{l} \text{SPEC} \quad \left[\langle \text{DEPENDENCY} \rangle \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \right] \\ \text{NTYPE} \left[\text{NSYN} \langle \text{PoS1} \rangle \right] \end{array} \right]$$

$$\left[\begin{array}{l} \text{SPEC} \quad \left[\langle \text{DEPENDENCY} \rangle \left\{ \left[\text{NTYPE} \left[\text{NSYN} \langle \text{PoS2} \rangle \right] \right] \right\} \right] \\ \text{NTYPE} \left[\text{NSYN} \langle \text{PoS1} \rangle \right] \end{array} \right]$$

The second type, DEP12_<PoS1>_<dependency>_<PoS2>_<lemma2>, is lexicalized with respect to the dependent of the dependency relation. It makes use of the same f-structure configurations as DEP11_<PoS1>_<dependency>_<PoS2>, but counts only f-structure snippets where the dependent's PRED feature has a given value.

$$\left[\begin{array}{l} \langle \text{DEPENDENCY} \rangle \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \langle \text{PoS2} \rangle \dots \end{array} \right] \\ \langle \text{PoS1} \rangle \dots \\ \dots \end{array} \right]$$

The third type, DEP21_<PoS1>_<lemma1>_<dependency>_<PoS2>, is lexicalized with respect to the head of the dependency relation. It also makes use of the f-structure configurations above, but counts only f-structure snippets where the head's PRED feature has a given value.

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \langle \text{DEPENDENCY} \rangle \left[\langle \text{POS2} \rangle \dots \right] \\ \langle \text{POS1} \rangle \dots \end{array} \right]$$

...

The fourth type, DEP22_<PoS1>_<lemma1>_<dependency>_<PoS2>_<lemma2>, finally, is lexicalized with respect to both the head and the dependent of the dependency relation. It thus counts only f-structure snippets where both the head's PRED feature and the dependent's PRED feature have given values.

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \langle \text{DEPENDENCY} \rangle \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \langle \text{POS2} \rangle \dots \end{array} \right] \\ \langle \text{POS1} \rangle \dots \end{array} \right]$$

...

In the following, we present instantiations of each property type and explain how they contribute to the disambiguation of example sentences.

- `fs_attr_val ADD-PROP DEP12_ATYPE_ADJUNCT_ATYPE_extrem` counts the number of sub-f-structures that have a PRED feature with the value *extrem* ('extreme(ly)') and are members of ADJUNCT sets in (sub-)f-structures that have an ATYPE feature. Being associated with a highly positive weight, it facilitates the capturing of the tendency of the adjective *extrem* to modify other adjectives rather than verbs when used adverbially, as in the example in (9.18), whose competing analyses are illustrated in Figure 9.18.

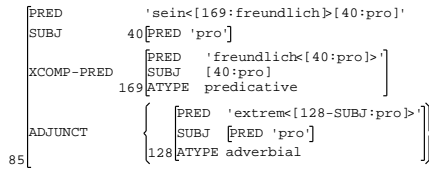
(9.18) Es war extrem freundlich.
 It was extremely friendly.
 'It was extremely friendly.'²⁵

- `fs_attr_val ADD-PROP DEP12_ATYPE_ADJUNCT_ATYPE_künftig` counts the number of sub-f-structures that have a PRED feature with the value *künftig* ('future(ly)') and are members of ADJUNCT sets in (sub-)f-structures that have an ATYPE feature. Being associated with a highly negative weight, it

²⁵s16117

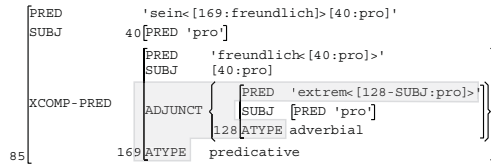
9.1 Additional properties based on grammar-internal information

"Es war extrem freundlich."



(a) evaluated as less probable

"Es war extrem freundlich."



(b) evaluated as relatively probable due to `fs_attr_val`
 ADD-PROP DEP12_ATYPE_ADJUNCT_ATYPE_extrem

Figure 9.18: Competing f-structures for (9.18)

makes it possible to capture the tendency of the adjective *künftig* to modify verbs rather than other adjectives when used adverbially. It thus contributes to the correct resolution of the attachment ambiguity in (9.19) by making the analysis shown in Figure 9.19(b) (p. 176) than the competing analysis.

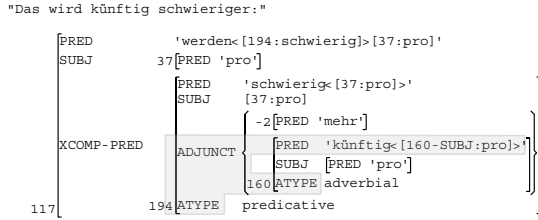
- (9.19) Das wird künftig schwieriger:
 That becomes futurely more difficult:
 'That will become more difficult:/That becomes futurely more difficult.'²⁶

- `fs_attr_val` ADD-PROP DEP21_common_Anwalt_APP_proper captures the tendency of the common noun *Anwalt* to take close appositions that are proper names. Being associated with a highly positive weight, it contributes to the correct identification of *Klaus Bollig* in (9.20) as an apposition to *Anwalt* and thus prefers this analysis, illustrated in Figure 9.20(b) (p. 177), over the alternative analysis of the complex name as a separate DP.²⁷

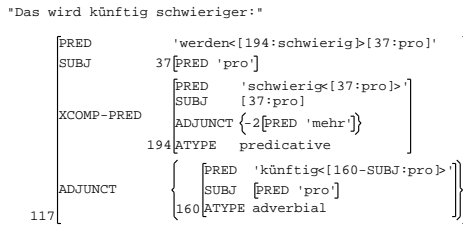
²⁶s44634

²⁷Based on the assumption that common nouns designating professions behave alike in this respect, we will define a more general property that, instead of being completely lexicalized,

Property design for the disambiguation of German LFG parses



(a) evaluated as relatively improbable due to `fs_attr_val`
`ADD-PROP DEP12_ATYPE_ADJUNCT_ATYPE_künftig`



(b) evaluated as more probable

Figure 9.19: Competing f-structures for (9.19)

(9.20) [...], das den Anwalt Klaus Bollig zum vorläufigen
 [...] which the lawyer Klaus Bollig to the interim
 Verwalter bestellte.
 administrator appointed.
 '[...] which appointed lawyer Klaus Bollig as interim administra-
 tor./[...] which appointed lawyer Klaus as interim administrator for
 Bollig.'²⁸

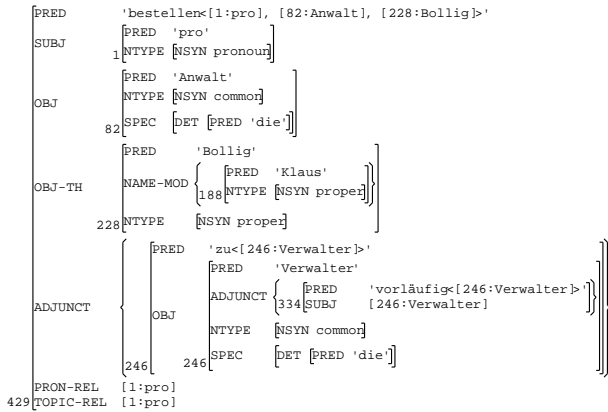
- `fs_attr_val` `ADD-PROP` `DEP21_VTYPE_aus#schließen_OBJ_proper` counts the number of f-structures that are headed by the verb *aus#schließen* and contain an OBJ that is a proper name. Being associated with a highly negative weight, the property reflects the fact that in the training data, proper

would be only mildly lexicalized in the sense that the head lemma of the dependency could be any common noun designating a profession. Given that we have a (potentially incomplete, but reasonably reliable) list of these nouns available, this is in fact straightforward to do. Nevertheless, we did not anticipate this generalization during property design, but discovered it after training by inspecting the properties that have proven to be relevant for disambiguation. Therefore, the property can only be included in a new cycle of experiments that we will run in the future.

²⁸s37602

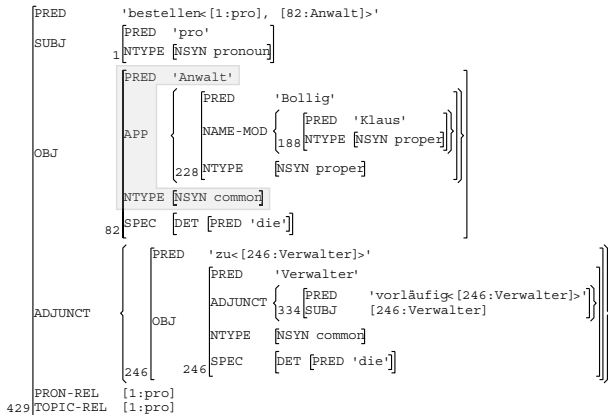
9.1 Additional properties based on grammar-internal information

"das den Anwalt Klaus Bollig zum vorläufigen Verwalter bestellte"



(a) evaluated as less probable

"das den Anwalt Klaus Bollig zum vorläufigen Verwalter bestellte"



(b) evaluated as relatively probable due to fs_attr_val ADD-PROP
DEP21_common_Anwalt_APP_proper

Figure 9.20: Competing f-structures for (9.20)

names occur rather as SUBJS than as OBJs of the verb *aus#schließen*, as it is the case in the f-structure shown in Figure 9.21(b) (p. 178), which is the intended analysis for (9.21).

- (9.21) Eine Fusion schließt Cromme nicht aus.
 A merger excludes Cromme not.
 ‘Cromme does not exclude a merger./
 A merger does not exclude Cromme.’²⁹

"Eine Fusion schließt Cromme nicht aus."

PRED	'aus#schließen [21:Fusion], [177:Cromme]>'
SUBJ	[PRED 'Fusion' NTYPE [NSYN commor]]
21	SPEC [DET [PRED 'eine']]
OBJ	[PRED 'Cromme' NTYPE [NSYN proper]]
177	
	ADJUNCT {197[PRED 'nicht']}
	TOPIC [21:Fusion]
125	VTYPE main

(a) evaluated as relatively improbable due to `fs_attr_val`
 ADD-PROP DEP21_VTYPE_aus#schließen_OBJ_proper

"Eine Fusion schließt Cromme nicht aus."

PRED	'aus#schließen [177:Cromme], [21:Fusion]>'
OBJ	[PRED 'Fusion' NTYPE [NSYN commor]]
21	SPEC [DET [PRED 'eine']]
SUBJ	[PRED 'Cromme' NTYPE [NSYN proper]]
177	
	ADJUNCT {197[PRED 'nicht']}
	TOPIC [21:Fusion]
125	VTYPE main

(b) evaluated as more probable

Figure 9.21: Competing f-structures for (9.21)

- `fs_attr_val` ADD-PROP DEP22_VTYPE_zeigen_OBJ_common_Folge counts the f-structure snippets where the common noun *Folge* is the OBJ of the verb *zeigen*. Being associated with a highly positive weight, it contributes to correctly identifying *die negativen Folgen* in (9.22) as the OBJ of the sentence rather than the SUBJ. Figure 9.22 illustrates the competing analyses.

²⁹s41657

Property design for the disambiguation of German LFG parses

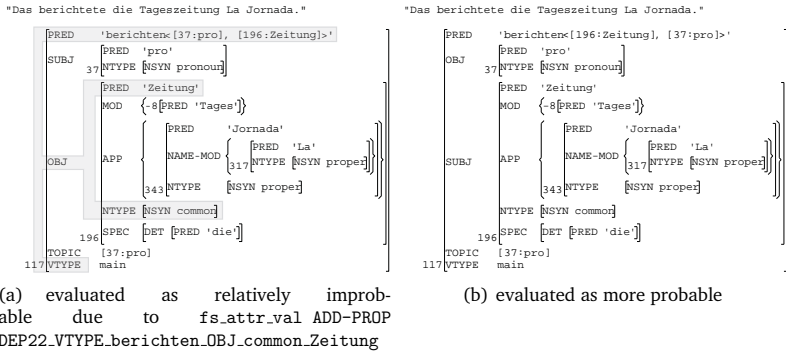


Figure 9.23: Competing f-structures for (9.23)

immediately follows the adjective *rückwirkend* generally modifies it and thus contributes to the correct resolution of the attachment ambiguity in (9.24), illustrated in Figure 9.24.

- (9.24) [...] Kodak übernimmt rückwirkend zum 1. Juli sein früheres
 [...] Kodak takes over retroactively as of 1st July its former
 Werk [...].
 factory [...].
 ‘[...] Kodak takes over his former factory [...] retroactively as of
 July 1.’³²

9.2 Additional properties based on external resources

In addition to the new properties based on grammar-internal information, we introduced a small number of properties based on external resources. The motivation for this is basically twofold: (i) Our intuition indicates that selectional preferences should play an important role in disambiguation. Information on the semantic category of dependents as it can be obtained on the basis of resources such as *GermaNet* is thus expected to help in correctly disambiguating syntactic analyses. Related to this, but in a slightly different perspective, many linguists working on differential object and subject marking have observed that the prominence of arguments on the (essentially semantic) animacy scale plays

³²s2264

being sparse in corpora of the size of our training corpus, which contains 8,881 annotated sentences. Any way of soundly integrating statistical information about bilocal dependencies acquired on much larger corpora may thus overcome or at least alleviate this limitation and is therefore of central interest for our approach.

Technically, the additional properties based on external resources are introduced into the analyses via the XLE transfer system just like the additional properties that are based on grammar-internal information. The external information is provided in the form of Prolog databases that have previously been created from the original external resources.

9.2.1 Properties based on *GermaNet* information on humanness, ‘groupness’ and animacy

We introduced three types of properties based on *GermaNet* information into the model, namely `isGNAnimate_<function>`, `isGNGroup_<function>`, `isGNHuman_<function>`. They are very similar in nature to the properties for resolving ambiguities due to case-ambiguous DPs on the basis of the nature of these DPs, such as `isDef_<function>` and are also inspired by Aissen (2003) and others’ work on differential object marking. Actually, the information in these properties refers to the animacy scale, which, according to Aissen (2003), is the other scale at work in differential object and subject marking. Prototypical (and hence potentially unmarked) subjects are high on the animacy scale, whereas prototypical (and hence potentially unmarked) objects are low on the animacy scale.

The only difference between the three new types of properties and the other properties referring to the nature of argument DPs is that instead of making reference to grammar-internal information, they refer to *GermaNet* information on humanness, ‘groupness’ and animacy that is not directly available in the grammar’s analyses, since there are no hard grammatical constraints in German that are driven by these features. Nevertheless, we assume that humanness, ‘groupness’ and animacy play a role in the identification of grammatical functions in the syntactic analysis of German text.

In the following, we present three representative instantiations of these property types in detail:

- `fs_attr_val ADD-PROP isGNAnimate_OBJ` counts the number of OBJs whose head noun belongs to the *GermaNet* synset *Tier* (animal). The *GermaNet* information is introduced into the competing analyses by means of the XLE term-rewriting system and a Prolog database previously created on the basis of the most recent *GermaNet* release that can be accessed by this term-rewriting system. The Prolog facts in the database that encode information on animacy have the following form.

9.2 Additional properties based on external resources

```
animate('Bulle').    'bull'  
animate('Färse').   'heifer'  
animate('Gockel').  'cock'  
animate('Hahn').    'cock'  
animate('Henne').   'hen'  
animate('Kater').   'male cat'  
animate('Katze').   '(female) cat'  
animate('Kuh').     'cow'  
animate('Ochse').   'ox'  
animate('Stier').   'bull'
```

- `fs_attr_val ADD-PROP isGNGroup.SUBJ` counts the number of SUBJS whose head noun is part of the *GermaNet* synset *Gruppe* (group). Whether a noun is part of the synset *Gruppe* is determined on the basis of facts like the following in the Prolog database previously created from the latest *GermaNet* release.

```
group('Belegschaft'). 'staff, personnel'  
group('Collegium').   'college'  
group('Haus').        'house'  
group('Hofstaat').   'royal household'  
group('Gefolge').    'entourage'  
group('Gefolgschaft'). 'following'  
group('Kollegium').  'college'  
group('Lehrerschaft'). 'faculty'  
group('Personal').   'personnel'  
group('Redaktion').  'editorial staff'
```

`fs_attr_val ADD-PROP isGNGroup.SUBJ` is associated with a positive weight; it thus contributes to the correct resolution of the SUBJ–OBJ ambiguity in the relative clause in (9.25), illustrated in Figure 9.25 (p. 184).

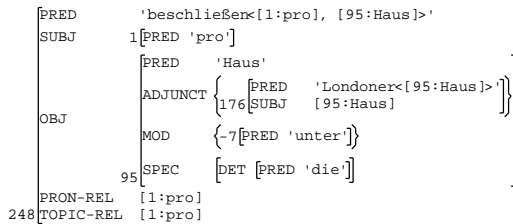
- (9.25) [...] die das Londoner Unterhaus beschlossen hat:
[...] which the London House of Commons decided has:
'[...] which the House of Commons in London has decided:/
which has decided the House of Commons in London:³³

- `fs_attr_val ADD-PROP isGNHuman.OBJ` counts the number of OBJs whose head noun is part of the *GermaNet* synset *Mensch* (human). Whether a noun is part of the synset *Mensch* is, again, determined on the basis of facts like the following in the Prolog database previously created from the latest *GermaNet* release.

³³s10159

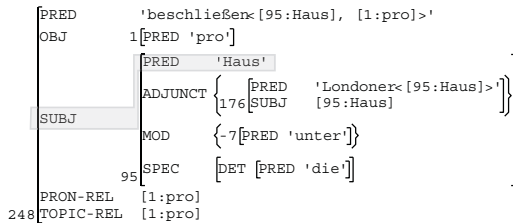
Property design for the disambiguation of German LFG parses

"die das Londoner Unterhaus beschlossen hat"



(a) evaluated as less probable

"die das Londoner Unterhaus beschlossen hat"



(b) evaluated as relatively probable due to fs_attr_val ADD-PROP isGNGroup_SUBJ

Figure 9.25: Competing f-structures for (9.25)

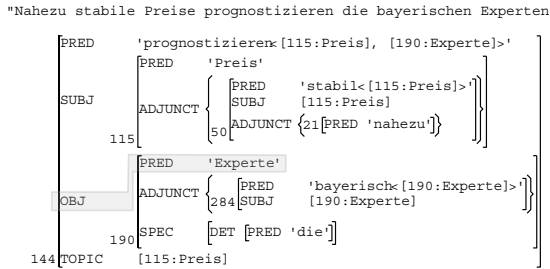
- human('Aktionär'). 'shareholder'
- human('Besitzer'). 'owner'
- human('Eigentümer'). 'owner'
- human('Experte'). 'expert'
- human('Freund'). 'friend'
- human('Inhaber'). 'holder, occupant'
- human('Insasse'). 'inmate'
- human('Junker'). 'donzel'
- human('Knacki'). 'lag, inmate'
- human('Redner'). 'speaker'

The weight associated with fs_attr_val ADD-PROP isGNHuman_OBJ is negative; this property thus contributes to the correct identification of *die bayerischen Experten* as the SUBJ rather than the OBJ of (9.26). In other words, it makes the analysis shown in Figure 9.26(b) more probable than the one in Figure 9.26(a).

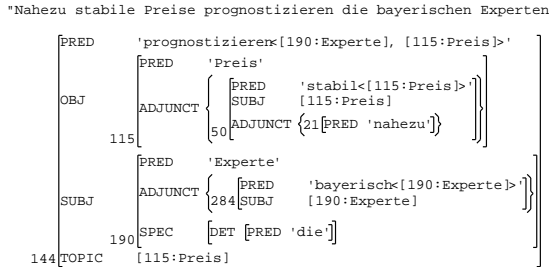
9.2 Additional properties based on external resources

- (9.26) Nahezu stabile Preise prognostizieren die bayerischen Experten
 Nearly stable prices forecast the Bavarian experts
 [...]

 [...].
 'The Bavarian experts forecast nearly stable prices [...]./Nearly stable prices forecast the Bavarian experts [...].'³⁴



(a) evaluated as relatively improbable due to `fs_attr_val`
`ADD-PROP isGNHuman_OBJ`



(b) evaluated as more probable

Figure 9.26: Competing f-structures for (9.26)

In order to evaluate the relevance of the properties base on *GermaNet* information on animacy, 'groupness' and humanness, we can examine whether they are selected in the process of automatic property selection, and we find out that more than half of them are under most parameter settings. Then, as a more strict test of their relevance for disambiguation, we also trained a model without them, everything else like the frequency-based cutoff and the regularization parameters being kept equal, and then compared this model to the model based on all properties. Table 9.1 (p. 186) shows the result.

³⁴s7360

relation/feature	all properties (6,378 provided, 4,340 selected)	all but <i>GermaNet</i> pr. (6,367 provided, 4,380 selected)
all	83.01	82.97
PREDS only	75.74	75.69
cj (conjunct of coord.)	68.0	67.6
gr (genitive attribute)	84.3	84.5
obj (arg. of prep. or conj.)	88.0	87.9
pred_restr	87	88
sb (subject)	73.4	73.2
sbp (logical subj. in pass. constr.)	63	61
fut (future)	86	84
pass_asp (passive aspect)	80	79

Table 9.1: F-scores (in %) in the 1,497 TiGer DB examples of our test set

We observe that the performance difference between the two models is extremely small and not statistically significant, which we find surprising and, as we have to admit, disappointing. We expected this type of information to make a larger contribution to the correct identification of grammatical functions. Moreover, we are surprised to see that only one of the grammatical functions for which this information was introduced, namely *sb*, is at all positively affected in terms of F-score by these additional properties. The other grammatical relations for which the F-scores achieved by the two models differ are actually not (or only very indirectly) related to the *GermaNet* information on animacy, ‘groupness’ and humanness that we introduced.

9.2.2 Auxiliary distributions from *Gramotron* data

A circumstance that the developers of log-linear models for syntactic disambiguation (as well as of many other probabilistic models used in NLP) regularly perceive as an important problem is the lack of training data. The developers have the intuition that properties capturing bilexical (or even trilexical) dependencies would have an important role to play in disambiguation, but weights for heavily lexicalized properties are often difficult to estimate reliably in a (semi-)supervised training scheme, as these heavily lexicalized properties occur too rarely in annotated corpora of a few hundreds, thousands or maximally tens of thousands of sentences. The more specialized the properties are, the more this problem is exacerbated.

One possible way of overcoming this problem is to use so-called auxiliary distributions, which are numeric scores calculated for competing analyses on the basis of statistical information that has previously been acquired on large

corpora. Contrarily to what the term ‘auxiliary distribution’ suggests, these numeric scores need not be sound probability distributions.

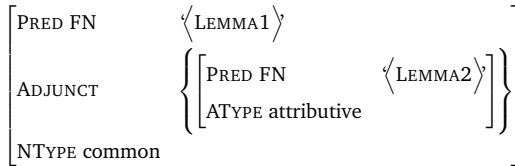
In the context of log-linear models for the disambiguation of deep syntactic analyses, this idea is first presented in Johnson & Riezler (2000), where the authors report on experiments with the English *ParGram* LFG on the HomeCentre Corpus. They base their auxiliary properties on statistical information on SUBJ and OBJ arguments of verbs and prepositions that they obtained with the help of a shallow parser from the British National Corpus. They then build two models: The first one contains only one auxiliary property, namely basically the sum of the logarithms of the conditional properties of SUBJ and OBJ lemmas given their function (SUBJ or OBJ) and the lemma of their head. The second model contains, in addition to the first auxiliary property, the number of SUBJ and OBJ dependencies found in the analysis considered and a normalized version of the first auxiliary property. The results of the experiments carried out with these two models are, as the authors themselves state it, disappointing because the improvements achieved by means of the auxiliary properties are very slight; the authors report an improvement of 1.67% on the Verbmobil Corpus in terms of the correct parses measure and of 0.6% on the HomeCentre Corpus, which may not be statistically significant. Regarding reasons for the contradiction between these results and the intuition that lexical dependencies are important for disambiguation, the authors conjecture that the disappointing results may be due to the important differences between the representations produced by the English *ParGram* LFG on the one hand and the shallow parser on the other hand. We think that other reasons may be more important: First of all, the dependencies captured by the auxiliary properties may actually be hardly affected by the ambiguities in the English parses, since, unlike in German, SUBJ–OBJ ambiguities are extremely rare in English. Providing information concerning mostly unambiguous dependencies cannot, of course, lead to much improvement in overall disambiguation. Second, the combinations of auxiliary properties introduced into the two experimental models may play a role. If it is important to counterbalance the effect of the number of dependencies in a given analysis, then one may actually achieve better results by either using the initial auxiliary property and the property encoding the number of dependencies or the normalized auxiliary property. Instead, the authors use one model with the unnormalized auxiliary property only and one model with all three auxiliary properties combined, where the initial, unnormalized property may actually counteract the positive effects of the normalized auxiliary property.

In our experiments, we therefore introduced only normalized auxiliary distributions into our model. All of them are based on conditional probabilities of given lexical items in certain syntactic contexts which were obtained in the *Gramotron* activities at the Institute for Natural Language Processing of the University of Stuttgart. There are four different kinds of dependencies captured by these auxiliary properties. All of them are calculated as follows: The n condi-

tional probabilities of the same kind in an analysis are multiplied and the result is taken to the power of $\frac{1}{n}$ for normalization. An alternative approach would be to sum up the n conditional probabilities of the same kind in an analysis and to divide the result by n for normalization. A potential advantage of this alternative is that zero probabilities do not necessarily cause the value of the property to be zero. For reasons of time, we could not compare the two approaches, but we plan to do so in future work.

In the following, we present the four properties based on auxiliary distributions in detail:

- `fs_attr_val ADJ-NOUN-PROB %X` refers to the following kind of f-structure configuration.



For each f-structure snippet of this kind encountered in an analysis, the conditional probability of the adjective lemma (Lemma2) given the noun lemma (Lemma1) is retrieved from a database created on the basis of *Gramotron* data. Since this information is introduced into the analyses by means of the XLE term-rewriting system, the database is a collection of Prolog facts like the following, where the first argument of the Prolog predicate `adj_noun` is the noun lemma (Lemma1), the second argument is the adjective lemma (Lemma2) and the third argument, the conditional probability of the adjective lemma given the noun lemma in this type of configuration.

```
adj_noun('Finland', 'fern', 0.33333).    'Finland', 'distant'
adj_noun('Finland', 'neutral', 0.33333). 'Finland', 'neutral'
adj_noun('Finsternling', 'ungenannt', 1.0). 'villain', 'undisclosed'
adj_noun('Finsternis', 'total', 0.28620). 'darkness', 'total'
adj_noun('Finsternis', 'tief', 0.11448). 'darkness', 'deep'
adj_noun('Finte', 'üblich', 0.15337). 'feint', 'usual'
adj_noun('Finte', 'politisch', 0.07975). 'feint', 'political'
adj_noun('Firlefanfanz', 'ganz', 0.12500). 'faldederal', 'whole'
adj_noun('Firma', 'eigen', 0.05162). 'company', 'own'
adj_noun('Firma', 'privat', 0.02709). 'company', 'private'
```

The n conditional probabilities of this kind encountered in a given analysis are then multiplied and the result is taken to the power of $\frac{1}{n}$. This nor-

9.2 Additional properties based on external resources

malized score is the value of the property `fs_attr_val ADJ-NOUN-PROB %X`.

- `fs_attr_val NOUN-MOD-PROB %X` refers to the following kind of f-structure configuration.³⁵

$$\left[\begin{array}{l} \text{PRED FN} \\ \text{ADJ-GEN} \\ \text{NTYPE common} \end{array} \left[\begin{array}{l} \langle \text{LEMMA1} \rangle \\ \left[\begin{array}{l} \text{PRED FN} \\ \text{NTYPE common} \end{array} \right] \\ \langle \text{LEMMA2} \rangle \end{array} \right] \right]$$

For each f-structure snippet of this kind encountered in an analysis, the conditional probability of the dependent noun lemma (Lemma2) given the head noun lemma (Lemma1) is also retrieved from the database used by the XLE term-rewriting system. Below are sample Prolog facts where the first argument of the Prolog predicate `noun_mod` is the head noun lemma (Lemma1), the second argument is the dependent noun lemma (Lemma2) and the third argument, the conditional probability of the dependent noun lemma given the head noun lemma in this type of configuration.

```
noun_mod('Abfolge', 'Bewegung', 0.06406).    'sequence', 'movement'
noun_mod('Abfrage', 'Information', 0.33333).  'query', 'information'
noun_mod('Abführung', 'Verdienst', 0.35211). 'payment', 'earnings'
noun_mod('Abführung', 'Abwasser', 0.29577).  'eduction', 'wastewater'
noun_mod('Abfüllung', 'Medikament', 0.50000). 'filling', 'drug'
noun_mod('Abfüllung', 'Stoff', 0.50000).    'filling', 'substance'
noun_mod('Abgabe', 'Stimme', 0.04994).     'casting', 'vote'
noun_mod('Abgabe', 'Droge', 0.03329).      'dispensing', 'drug'
noun_mod('Abgabe', 'Erklärung', 0.03313).  'execution', 'declaration'
noun_mod('Abgabe', 'Methadon', 0.01665).   'dispensing', 'methadone'
```

The final value of the property `fs_attr_val NOUN-MOD-PROB %X` is computed on the basis of these conditional probabilities in the same way as the value of the property `fs_attr_val ADJ-NOUN-PROB %X`.

- `fs_attr_val NOUN-PP-PROB %X` refers to one of the following kinds of f-structure configurations.

³⁵The name was taken straight from the resources compiled in the *Gramotron* activities. It is not related to the MOD dependency used in *ParGram* in any way.

Property design for the disambiguation of German LFG parses

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \text{ADJUNCT} \quad \left\{ \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \text{PTYPE} \dots \end{array} \right] \right\} \\ \text{NTYPE common} \end{array} \right]$$

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \text{OBL} \quad \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \text{PTYPE} \dots \end{array} \right] \\ \text{NTYPE common} \end{array} \right]$$

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \text{OBL-DIR} \quad \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \text{PTYPE} \dots \end{array} \right] \\ \text{NTYPE common} \end{array} \right]$$

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \text{OBL-LOC} \quad \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \text{PTYPE} \dots \end{array} \right] \\ \text{NTYPE common} \end{array} \right]$$

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \text{OBL-MANNER} \quad \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \text{PTYPE} \dots \end{array} \right] \\ \text{NTYPE common} \end{array} \right]$$

For each f-structure snippet of one of these kinds, the conditional probability of the preposition lemma (Lemma2) given the noun lemma (Lemma1) is again retrieved from the database created on the basis of *Gramotron* data. Below are sample Prolog facts where the first argument of the Prolog predicate `noun_pp` is the noun lemma (Lemma1), the second argument is the preposition lemma (Lemma2) and the third argument, the conditional probability of the preposition lemma given the noun lemma in these types of configurations.

9.2 Additional properties based on external resources

noun_pp('Abtransport', 'nach', 0.45792).	'evacuation', 'to'
noun_pp('Abtretung', 'an', 1.00000).	'cession', 'to'
noun_pp('Abwanderung', 'nach', 0.11015).	'migration', 'to'
noun_pp('Abwurf', 'aus', 0.59630).	'dropping', 'from'
noun_pp('Abzahlung', 'für', 1.00000).	'repayment', 'for'
noun_pp('Abzeichen', 'auf', 0.12778).	'insignia', 'on'
noun_pp('Abzug', 'aus', 0.70696).	'pull-out', 'from'
noun_pp('Abzweigung', 'nach', 1.00000).	'junction', 'to'
noun_pp('Achtung', 'vor', 0.86352).	'respect', 'for'
noun_pp('Adapter', 'für', 0.79874).	'adapter', 'for'

- `fs_attr_val` PREP-NOUN-PROB %X, finally, refers to the following kind of f-structure configuration.

$$\left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA1} \rangle \\ \text{OBJ} \quad \left[\begin{array}{l} \text{PRED FN} \quad \langle \text{LEMMA2} \rangle \\ \text{NTYPE common} \end{array} \right] \\ \text{PTYPE ...} \end{array} \right]$$

For each f-structure snippet of this kind encountered in an analysis, the conditional probability of the noun lemma (Lemma2) given the preposition lemma (Lemma1) is yet again retrieved from the database created on the basis of *Gramotron* data. Below are sample Prolog facts where the first argument of the Prolog predicate `prep_noun` is the preposition lemma (Lemma1), the second argument is the case governed by the preposition, the third argument is the noun lemma (Lemma2) and the fourth argument, the conditional probability of the noun lemma given the preposition lemma in this type of configuration.

<code>prep_noun('unter', 'dat', 'Druck', 0.05391).</code>	'under', 'pressure'
<code>prep_noun('unter', 'dat', 'Nummer', 0.04930).</code>	'under', 'number'
<code>prep_noun('unter', 'dat', 'Umstand', 0.03084).</code>	'under', 'circumstance'
<code>prep_noun('unter', 'dat', 'Bedingung', 0.02711).</code>	'under', 'condition'
<code>prep_noun('unter', 'dat', 'Motto', 0.02228).</code>	'under', 'motto'
<code>prep_noun('unter', 'dat', 'Kontrolle', 0.02183).</code>	'under', 'control'
<code>prep_noun('unter', 'dat', 'Schutz', 0.01771).</code>	'under', 'protection'
<code>prep_noun('unter', 'dat', 'Leitung', 0.01579).</code>	'under', 'direction'
<code>prep_noun('unter', 'dat', 'Name', 0.01322).</code>	'under', 'name'
<code>prep_noun('unter', 'dat', 'Titel', 0.01090).</code>	'under', 'title'

As with the properties based on *GermaNet* information, we can examine whether the four auxiliary properties that we introduced are selected in the process of automatic property selection in order to evaluate their relevance. For

`fs_attr_val NOUN-PP-PROB %X`, this is the case in all models that we trained according to a scheme involving property selection. The other three auxiliary properties are included in the models trained on the data resulting from the application of a frequency-based cutoff of 4; nevertheless, `fs_attr_val NOUN-PP-PROB %X` is the auxiliary property associated with the highest weight.

In order to verify whether the four auxiliary properties actually improve the performance of our model, we then trained a model without them, everything else (frequency-based cutoff and regularization parameters) being kept equal. Table 9.2 (p. 192) shows the – admittedly disappointing – result.

relation/feature	all properties (6,378 provided, 4,340 selected)	all pr. but aux. distr. (6,374 provided, 4,280 selected)
all	83.01	82.99
PREDS only	75.74	75.76
<code>oa</code> (accusative object)	75	74
<code>oc_fin</code> (finite cl. obj.)	64	63
<code>op_dir</code> (directional argument)	23	24
<code>quant</code> (quantifying determiner)	68	69
<code>rc</code> (relative clause)	62	61
<code>case</code>	85	84
<code>fut</code> (future)	86	84
<code>perf</code> (perfect)	85	86

Table 9.2: F-scores in (%) in the 1,497 TiGer DB examples of our test set

We observe that the performance difference between the two models is minimal and definitely not statistically significant. What is more, the few grammatical relations and morphosyntactic features for which the F-score varies between the two models are mostly unrelated to the four auxiliary distributions. While the small improvements observed for `rc` and `case` may be due to them, the F-score of `op_dir`, which is supposed to take advantage of `fs_attr_val NOUN-PP-PROB %X`, actually decreases slightly, and `oa`, `oc_fin`, `quant`, `fut` and `perf` seem completely unrelated to the auxiliary properties.

More work is thus needed in order to find a good way of taking advantage of auxiliary distributions as properties of log-linear models for disambiguation. As already mentioned, we plan to carry out experiments with alternative ways of calculating the effective score of the auxiliary properties on the basis of the auxiliary distributions. In addition, we hope to introduce more auxiliary distributions that capture more kinds of dependencies. E.g., it may be the case that it is not enough to have an auxiliary property like `fs_attr_val NOUN-PP-PROB %X`, but that this auxiliary property must be counterbalanced by auxiliary properties referring to alternative attachments of PPs, most importantly attachments to verbs and adjectives.

9.3 Evaluation

The results in terms of F-score and error reduction achieved on our test set of 1,497 TiGer DB structures with the best-performing model based on both the XLE template-based and the newly introduced properties are shown in Table 9.3. The parameters for combined property selection and regularization (see Chapter 10) were adjusted on the 371 TiGer DB structures of our held-out set. For comparison, we repeat the figures achieved with the log-linear model presented in Chapter 8, which exclusively uses the XLE template-based properties.

We observe that the overall F-score is improved significantly; overall error reduction increases from 34.5% to 51.0%. What is particularly interesting is the considerably better error reduction for the core grammatical functions *sb* (subject) and *oa* (accusative object). But also for *rcs* (relative clauses) and *mos* (modifiers or adjuncts), which are notoriously difficult for disambiguation due to PP and ADVP attachment ambiguities, we observe an improvement in F-score.

9.4 Comparison to similar systems

Although it is difficult to compare the results just reported to the results reported for other systems, we make an attempt here. Two types of systems are interesting for comparison: On the one hand, these are parsers for German for which a dependency-based evaluation on corpus data has been carried out. On the other hand, we compare our results, and in particular the error reduction we achieve, to the results reported for other unification-based grammars which can parse free text and for which an evaluation has been performed in terms of the lower bound and the upper bound defined by the symbolic grammars as well as the F-score achieved by means of stochastic parse selection.

The best results achieved so far in a dependency-based evaluation of a German parser are reported in Foth et al. (2004). Their dependency parser based on weighted constraints (WCDG) achieves an accuracy of 87.0% on a dependency-based gold standard that was semi-automatically constructed from a part of the NEGRA Treebank (Daum et al. 2004). This accuracy being considerably higher than both our overall and our PREDs only F-scores, we would like to stress that the WCDG parser does not perform lemmatization nor does it decompose compounds, so that mismatches due to differences in lemmatization that do occur in our evaluation cannot affect their figures. Furthermore, it should be noted that their gold standard does not encode reentrancies, which may also be a type of dependency for which our system performs below average.

Property design for the disambiguation of German LFG parses

relation/feature	all properties		template-based pr.	
	F-score	error red.	F-score	error red.
all	83.01	51.0	82.17	34.5
PREDS only	75.74	46.5	74.69	31.0
app (close apposition)	60	63	61	75
app_cl (appositive clause)	53	100	52	86
cc (comparative complement)	19	-29	19	-29
cj (conjunct of coord.)	68	50	67	25
da (dative object)	63	67	62	58
det (determiner)	91	50	91	50
gl (genitive in spec. pos.)	88	75	88	75
gr (genitive attribute)	84	56	84	56
mo (modifier)	63	36	62	27
mod (non-head in compound)	89	29	89	29
name_mod (non-head in compl. name)	80	33	81	67
number (number as determiner)	81	33	81	33
oa (accusative object)	75	77	69	31
obj (arg. of prep. or conj.)	88	50	87	25
oc_fin (finite cl. obj.)	64	0	64	0
oc_inf (infinite cl. obj.)	82	0	82	0
op (prepositional obj.)	54	40	54	40
op_dir (directional argument)	23	13	23	13
op_loc (local argument)	49	29	49	29
pd (predicative argument)	60	50	59	25
pred_restr	87	62	84	38
quant (quantifying determiner)	68	33	68	33
rc (relative clause)	62	20	59	0
sb (subject)	73	63	71	38
sbp (logical subj. in pass. constr.)	63	62	61	46
case	85	75	83	50
comp_form (complementizer form)	72	0	74	100
coord_form (coordinating conj.)	86	100	86	100
degree	88	50	87	0
det_type (determiner type)	95	-	95	-
fut (future)	86	-	86	-
gend (gender)	90	60	89	40
mood	90	-	90	-
num (number)	89	50	89	50
pass_asp (passive aspect)	80	100	79	0
perf (perfect)	85	0	86	100
pers (person)	84	83	82	50
pron_form (pronoun form)	73	-	73	-
pron_type (pronoun type)	70	0	71	100
tense	91	0	91	0

Table 9.3: F-scores (in %) in the 1,497 TiGer DB examples of our test set

Schiehlen (2003) reports a dependency F-score of 85.20% on the NEGRA Treebank for an approach that combines information from several parsers, and Schiehlen (2004) reports a dependency F-score of 81.69%, also on the NEGRA Treebank, for a context-free grammar that was induced from a version of the treebank that was augmented with additional information. Dubey (2004), finally, who also worked principally with the NEGRA Corpus and extracted from it a history-based parser for German similar to the Collins parser for English, reports a labeled dependency F-score of 82.2%. All these figures are higher than our PREDS only F-score, but again it should be noted that the level of granularity in the TiGer Dependency Bank structures is higher than the level of granularity in the dependencies on which these parsers were evaluated, so that the task of producing TiGer DB structures can arguably be judged as more difficult than the task of identifying NEGRA style dependencies.

A different approach to identifying syntactic dependencies in free German text is presented in Cahill et al. (2005). In this paper, an LFG approximation induced from the TIGER Treebank is presented and evaluated. It achieves an F-score of 71% on 100 semi-manually annotated gold standard structures that are very similar to the structures in the TiGer Dependency Bank, and an F-score of 74.61% is reported for an evaluation on 2,000 f-structures that were produced by automatically f-annotating the corresponding TIGER Treebank trees.³⁶ The results of the German *ParGram* LFG compare favorably with these results, even if we base this claim on the PREDS only F-score.

In summary, the German *ParGram* achieves competitive results in the task of identifying grammatical relations in free German text, even if it performs less well than some other systems, in particular Foth et al. (2004). However, we believe that its analyses are interesting for the high level of linguistic detail contained in them. Moreover, it is the only grammar for German that both achieves broad coverage on free text when parsing and can be used in generation.

As for hand-crafted unification-based grammars for other languages that have been evaluated on free text, there are only two systems that we are aware of: the English *ParGram* LFG and the HPSG-inspired *Alpino* parser for Dutch. Riezler et al. (2002) report an F-score of 78.6% corresponding to an error reduction of 36% for an evaluation on the entire PARC 700 Dependency Bank (King et al. 2003). For basically the same system evaluated on half of the PARC 700 DB (the other half being used for adjusting hyperparameters), Riezler & Vasserman (2004) report an F-score of 79.3%, but give no clear indication as to the error reduction that this F-score represents.³⁷ The F-scores achieved by the

³⁶The latter evaluation may thus be slightly biased in favor of this approach, as the same types of erroneous annotations may give rise to the evaluated f-structures and to the gold standard f-structures.

³⁷The lower bound F-score of the system reported in Riezler & Vasserman (2004) is so much lower than the lower bound F-score reported in Riezler et al. (2002) that we suspect it of having

German *ParGram* LFG are comparable to the F-score figures reported for the English *ParGram* LFG; the overall F-score of 83.01% is clearly better, but the PREDs only F-score of 75.74% is lower, and it has to be taken into account that the PARC 700 DB encodes fewer morphosyntactic features than the TiGer DB. With respect to error reduction, the German system compares favorably to the English system, since it achieves an error reduction of 51% compared to 36%. These figures indicate that both LFG as a formalism for hand-crafted broad-coverage grammars and the data-driven disambiguation techniques employed port well from English to German, which differs considerably from English in such aspects as configurationality, type/token ratio etc., which seem to have a serious impact on other parsing models.

For the Dutch *Alpino* parser, Malouf & van Noord (2004) report an F-score of 85.78% on the dependencies annotated in the Alpino Corpus, and van Noord (2006) even reports an F-score of 88.5%. Since these figures exceed the upper bound of our system, we assume that they are mainly due to extremely careful and targeted error mining in the symbolic part of the grammar (van Noord 2004). However, the disambiguation module may also contribute to these extremely good results, as Malouf & van Noord (2004) report an error reduction of 78%,³⁸ even if we have to keep in mind that the *Alpino* system does not employ an OT-mark-driven pre-filter and that, hence, the lower bound figure on which the calculation of the reported error reduction is based is considerably lower than the lower bound F-score of the German *ParGram* LFG.

9.5 Summary

In this chapter, we have presented new properties that are defined by means of ‘transfer’ rules as they are processed by XLE’s term-rewriting system. Most of these properties refer to information that is actually present in the output representations of the grammar, but that cannot be accessed directly by means of the XLE property templates. These properties have been developed in a linguistically informed way, i.e. by trying to capture all kinds of factors that are known or assumed to play a rule in German word order and the identification of grammatical relations. In addition, we have experimented with auxiliary properties based on probabilities computed on a very large corpus that was analyzed by a shallow parser. By providing these different kinds of new properties, we improve the error reduction of the log-linear model used for syntactic disambiguation from 34.5% to 51.0%.

been miscalculated and would thus not want to base our calculation of error reduction on it.

³⁸measured on concept accuracy, not F-score

Chapter 10

Property selection and regularization

This chapter discusses several regularization and property selection techniques, whose purpose is to develop more accurate models by preventing them from overfitting the training data and more compact models by discarding irrelevant and often redundant properties.

10.1 The problem of overfitting

As we have seen so far, log-linear models can be trained on data that are partially labeled and the learning features or properties used need not be linearly independent. This may lead to the assumption that, as a developer of a disambiguation component based on a log-linear model, you just extract from your data any kind of property you imagine might be relevant for disambiguation and retrain the model with these additional properties.

However, log-linear models tend to overfit the training data, in particular when the number of properties used is far greater than the amount of data available. Since the number of properties that we initially extract from parses is far greater than the number of packed c- and f-structure representations in our training corpus, we are in this situation. This means that a training regime without any type of property selection or regularization produces property weights that are so closely adapted to the training data that they generalize poorly to unseen data. This is particularly likely to happen with weights of sparse properties, but more frequent properties may also be associated with extreme weights that are due to some coincidence in the training data and are not suitable for the application of the model to unseen data.

The problem of overfitting can be addressed by two related strategies, namely property selection and regularization. The following sections present these strategies in detail.

10.2 Property selection

Property selection consists of choosing, among the initial set of properties, the ones that are really useful for disambiguation. By reducing the number of properties, it helps to avoid overfitting, but in addition, there is a further aspect that makes property selection attractive: efficiency. As we have stated already in Subsection 3.2.4, it is reasonable to assume that the time necessary for property extraction is proportional to the number of properties extracted. Furthermore, the time needed for computing the probability of a reading is reduced when using a simpler model instead of a more complex model. Property selection being important for both the quality and the efficiency of the models trained, we believe that property selection, alongside property design, is the key problem in the development of log-linear models.

10.2.1 Frequency-based cutoffs

A popular and relatively straightforward way to address the problem of overfitting is to use a frequency-based cutoff that indicates how often a property has to occur in the training data in order to be considered for training. Nevertheless, there are a number of variants of ‘the’ frequency-based cutoff reported in the literature, and although one might think that the differences are minor, we believe that they have considerable consequences for the ability of the cutoff to reduce the number of properties and to minimize overfitting.

Riezler & Vasserman (2004) report figures for a frequency-based cutoff c that checks whether a given property occurs at least c times in the unlabeled training data, summing over all readings in the unlabeled packed c - and f -structure representations. This cutoff thus directly considers the frequency of the properties available. Let us consider an example: Let c be 16, which is the optimal cutoff established on their held-out set in Riezler & Vasserman (2004). Our property `fs_attr_val ADD-PROP DEP22_VTYPE_ab#1ehlenen_OBJ_common_Kritik` occurs once in 20 unintended readings out of the 46 possible analyses of the TIGER Corpus sentence # 22,581; it ‘survives’ the frequency-based cutoff. In contrast, our property `fs_attr_val ADD-PROP DEP22_VTYPE_bestehen_OBJ_common_Gefahr` is discarded on the basis of this frequency-based cutoff because it occurs less than 16 times in all readings of all sentences. Actually, it occurs once in 2 unintended readings out of the 6 possible analyses of TIGER Corpus sentence # 13,066, once in 5 unintended analyses out of the 15 possible ones of sentence # 24,460 and once in 1 unintended reading out of the 4 possible solutions of sentence # 35,669.

Considering that `fs_attr_val ADD-PROP DEP22_VTYPE_bestehen_OBJ_common_Gefahr` is discriminative in 3 sentences, while `fs_attr_val ADD-PROP`

DEP22_VTYPE_ab#1ehnen_OBJ_common_Kritik is discriminative in only 1 sentence, the effect of the cutoff seems problematic. Actually, `fs_attr_val ADD-PROP DEP22_VTYPE_ab#1ehnen_OBJ_common_Kritik` only ‘survives’ the frequency-based cutoff as defined by Riezler & Vasserman (2004) because it occurs in a relatively highly ambiguous sentence, whereas `fs_attr_val ADD-PROP DEP22_VTYPE_bestehen_OBJ_common_Gefahr` is discarded because it occurs in more, but less ambiguous, sentences. Note that most of the ambiguities that cause the relatively high number of readings in sentence # 22,581 have nothing to do with and, hence, cannot be resolved by the property `fs_attr_val ADD-PROP DEP22_VTYPE_ab#1ehnen_OBJ_common_Kritik`.

We are convinced that this behavior of the cutoff is a serious disadvantage. In our opinion, there is no reason for a property that happens to occur in many analyses of a single highly ambiguous sentence to be preferred over a property that occurs only once in one analysis (or a few analyses) of a few sentences. In fact, this makes it very difficult, if not impossible, to interpret such a frequency-based cutoff. As there are so many ways (not) to satisfy a cutoff of, say, 16, this number does not tell us much (if anything) about the frequency with which a property should occur in order to allow for the reliable estimation of the corresponding weight.

Furthermore, this type of cutoff can only be applied to properties that are themselves frequency-based. As soon as the set of properties comprises auxiliary distributions (which usually take values between 0 and 1), properties that record (potentially negative) distances or length differences between constituents etc., it can no longer be used. Finally, due to the fact that many frequency-based properties occur several times in a single parse, this frequency-based cutoff does little to reduce the number of properties that are kept for training. Riezler & Vasserman (2004) report a compression of the number of properties by 18.4%, which is surprisingly little for a cutoff of 16 (compare Table 10.1 on p. 202) and can only be explained by its somewhat peculiar definition.

Malouf & van Noord (2004) report figures for a different frequency-based cutoff c that discards all properties that are relevant in the parses of fewer than $c + 1$ sentences. In order to be relevant for a sentence, a property must have different values for at least two competing analyses of the sentence under consideration. This kind of frequency-based cutoff is insensitive to the frequency of properties within parses, but this feature is in fact rather an advantage than a disadvantage, since this kind of cutoff can hence be applied to any kind of property and not only to frequency-based ones. However, if the definition of *relevant* is really the one in Malouf & van Noord (2004), this cutoff actually does not ensure that the property under consideration can contribute to discriminating the intended reading(s) from the unintended ones, as the value of a *relevant* property may differ between any pair of competing analyses, not necessarily between the intended analysis and one or several unintended ones.

We have therefore adopted a very similar, but slightly more elaborate frequency-based cutoff c , namely the one presented in Charniak & Johnson (2005). It consists in discarding all properties whose values differ between the intended tree and at least one of the competing unintended trees of fewer than c sentences. In other words, the property has to contribute to disambiguation in at least c sentences, c being set to 5 in their case. We then adapted this cutoff for our purposes, which was necessary because Charniak & Johnson (2005) have fully labeled data available, whereas we only have partially labeled data. Our redefinition of the cutoff is as follows: For at least c sentences, the average value of a given property in the intended analyses must differ from the average value of this same property in the unintended analyses.

Table 10.1 (p. 202) shows the effect of this kind of frequency-based cutoff on the number of properties considered for training. As can easily be seen, the compression rates are very high for this kind of cutoff; with our set of properties, it is above 80% for a cutoff as low as 3 and above 90% for a cutoff as low as 5.¹ In addition, the table also shows that this frequency-based cutoff helps us to avoid overtraining, since the F-scores obtained with the reduced models that result from the application of a cutoff of, e.g., 4 is significantly better than the F-score of the model that includes all 57,934 properties.²

A somewhat surprising result of our experiments with different cutoffs is that the model resulting from a cutoff of 12 achieves the best result among these unregularized models, and that the cutoff can be set as high as 180 without adversely affecting F-score (in comparison with the model including all properties). It is only when increasing the cutoff from 180 to 200 that we observe a significant drop in F-score between the resulting models.³

These observations lead us to the conclusion that there is an enormous amount of redundancy in the set of properties and that, hence, many properties are actually not needed or can even be a hindrance for high-quality disambiguation. Our frequency-based cutoff can effectively cut down the number of properties and thus allows for the development of more compact models that generalize better to unseen data. Nevertheless, a frequency-based cutoff is, as Perkins et al. (2003) convincingly argue, always ad hoc in nature, since the frequency of a property tells us rather little about its ability to discriminate

¹The compression rates indicated in Table 10.1 are calculated on the basis of the 57,934 properties that contribute to disambiguation in at least one sentence from the training set. It should be noted that the number of properties formulated in our list of properties exceeds 300,000 and that more than 90,000 out of these are active in the parses of the training set.

²The model resulting from the application of a cutoff of 4 is better than the model including all properties at a confidence level of more than 95%. The difference is less significant for the models resulting from the application of cutoffs of 2 and 3. They are better than the model including all properties only at a confidence level of 90%.

³The model resulting from the application of a cutoff of 180 is better than the model resulting from the application of a cutoff of 200 at a confidence level of more than 95%.

between intended and unintended analyses.⁴ What the frequency does give us an indication of, however, is whether for a given property, there are sufficient data for the reliable estimation of its weight. We therefore advocate the use of a frequency-based cutoff, but argue for being ‘conservative’ in its use. In our concrete case, this means that, in all subsequent experiments, we worked with the data resulting from the application of cutoffs of 3 and 4, 3 being the cutoff most similar in its definition to the one applied by Malouf & van Noord (2004) and 4 being, among the cutoffs below 10, the one that results in the best-performing model.

Of course, the exact compression rate at which a given frequency-based cutoff of this kind reduces the number of properties considered for training depends very much on the exact nature of the properties. In order to illustrate this, we present the compression rates for cutoffs 1 through 6 broken down according to property ‘families’ in Table 10.2 (p. 203). For the template-based properties (Table 10.2a), i.e. the kinds of properties that are also used in the disambiguation component of the English *ParGram* LFG, compression rates are relatively low. Even with a cutoff of 6, the compression rate stays below 50% for all template-based property families except the lexicalized property family based on the template `lex.subcat`. Within the template-based property families, we observe that, e.g., the compression rate is higher for the property family based on the template `cs_adjacent_label` than it is for the `cs_label` properties, which is plausible because `cs_adjacent_label` properties, which are parameterized for two c-structure categories, are more specific than `cs_label` properties, which are parameterized for just one c-structure category. As for the additional properties that were specially designed for the disambiguation of German LFG parses (Tables 10.2b and 10.2c), we also notice that the compression rates observed depend very much on the degree of specialization of the properties, in particular on lexicalization. Entirely unlexicalized property families, such as the one that comprises properties like `isDef_SUBJ` (see column `isDef_SUBJ` in Table 10.2b), are hardly affected by the frequency-based cutoff. Mildly lexicalized property families, such as the one that contains `VerbsWithAccObjAndGenObjTh.OBJ_precedes.OBJ-TH` (see column `VerbsWith` in Table 10.2b), or property families that are lexicalized for frequent lexical items like prepositions, such as the one that contains `VADJUNCT_nach_precedes_über` (see column `[ANV]ADJUNCT` in Table 10.2b), are

⁴The frequency-based cutoff used by Riezler & Vasserman (2004) actually does not take into account any information with respect to the discriminative power of properties. The frequency-based cutoff presented in Malouf & van Noord (2004) does to some extent, although the definition of a *relevant* property does not ensure that such a property is discriminative. The cutoff proposed by Charniak & Johnson (2005), finally, of which our frequency-based cutoff is a variant, does take the discriminativeness of properties into account. This being said, the mere amount of data for which a property is discriminant is, of course, only vaguely related to the quality of a property as a predictor.

cutoff	# properties	compression	F-score	error reduction
1	57,934	–	83.89	34.8%
2	16,422	71.7%	84.21	42.0%
3	9,157	84.2%	84.12	40.0%
4	6,378	89.0%	84.34	44.8%
5	4,898	91.5%	84.25	42.8%
6	3,997	93.1%	84.20	41.7%
7	3,430	94.1%	84.23	42.4%
8	3,009	94.8%	84.20	41.7%
9	2,667	95.4%	84.26	43.1%
10	2,460	95.8%	84.29	43.7%
11	2,249	96.1%	84.36	45.2%
12	2,088	96.4%	84.38	45.8%
20	1,445	97.5%	84.23	42.4%
30	1,129	98.1%	84.22	42.3%
50	812	98.6%	84.14	40.5%
70	679	98.8%	84.27	43.3%
90	610	98.9%	84.10	39.4%
100	569	99.0%	84.07	38.8%
120	518	99.1%	83.79	32.5%
140	480	99.2%	84.03	38.1%
160	447	99.2%	83.96	36.4%
180	412	99.3%	84.02	37.7%
200	388	99.3%	83.64	29.3%

Table 10.1: F-scores for different numbers of properties resulting from different frequency-based cutoffs on the 387 example sentences of our held-out set

affected to an intermediate degree. Finally, highly specialized property families that encode bilexical dependencies like the DEP22 property family (see Table 10.2c) are very strongly affected by this kind of frequency-based cutoff.

A further observation we can make from these figures is that the labelling in the data must make properties discriminative in order to allow them to ‘survive’ our frequency-based cutoff, which takes the discriminativeness of properties into account. The property families <lemma>_MOD_<lemma> (see column `vor#sitzen_MOD.SPD` in Table 10.2c) and MOD_<lemma> (see column `MOD_SPD` in Table 10.2c) are lexicalized, the former even with respect to two lemmas, which makes these properties rather sparse, but moreover the labelling in the data often does not make them discriminative. This is due to the fact that the dependency MOD encodes the relationship between a compound head and its non-head(s), but that the decomposition of compounds is not annotated in the TIGER Treebank and, hence, the labelling in our training data does not allow us to discriminate between alternative decompositions of compounds.

(a) Template-based property families

cutoff	cs_label		cs_num_children		cs_adjacent_label		fs_attrs		fs_attr_val		fs_adj_attr		lex_subcat	
	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.
1	212	-	181	-	673	-	113	-	184	-	293	-	985	-
2	185	13%	161	11%	527	22%	109	4%	156	15%	260	11%	417	58%
3	172	19%	148	18%	453	33%	106	6%	144	22%	249	15%	246	75%
4	168	21%	147	19%	425	37%	105	7%	136	26%	244	17%	158	84%
5	159	25%	141	22%	393	42%	102	10%	132	28%	230	22%	114	88%
6	152	28%	136	25%	363	46%	99	12%	127	31%	226	23%	89	91%

(b) Unlexicalized and rather mildly lexicalized property families

cutoff	isDef.SUBJ		DEP11		DEPTH-OF		PATH		VerbsWith		[ANV]ADJUNCT		ab#bauen.OBJ	
	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.
1	61	-	108	-	12	-	40	-	6	-	681	-	2,072	-
2	59	3%	100	7%	12	-	34	15%	3	50%	413	39%	875	58%
3	57	7%	97	10%	9	25%	29	28%	3	50%	284	58%	539	74%
4	57	7%	93	14%	9	25%	26	35%	3	50%	216	68%	354	83%
5	56	8%	89	18%	9	25%	24	40%	3	50%	181	73%	250	88%
6	54	11%	87	19%	9	25%	21	48%	3	50%	146	79%	195	91%

(c) Relatively heavily lexicalized property families

cutoff	DEP21		F2		DEP12		ACTIVE		DEP22		vor#sitzen.MOD_SPD		MOD_SPD	
	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.	# pr.	compr.
1	12,194	-	1,589	-	10,060	-	312	-	27,793	-	181	-	142	-
2	4,638	62%	587	63%	3,544	65%	107	66%	4,134	85%	30	83%	29	80%
3	2,605	79%	306	81%	1,957	81%	46	85%	1,648	94%	10	94%	7	95%
4	1,725	86%	217	86%	1,311	87%	27	91%	909	97%	2	99%	4	97%
5	1,226	90%	171	89%	949	91%	17	95%	604	98%	2	99%	4	97%
6	925	92%	136	91%	752	93%	9	97%	425	98%	1	99%	0	100%

Table 10.2: Numbers of discriminative properties broken down according to property families and different frequency cutoffs

10.2.2 Incremental property selection

Another strategy for reducing the number of properties included in the final log-linear model is incremental property selection. This idea goes back to Tibshirani (1996) and Della Pietra et al. (1997), was refined by Perkins et al. (2003) and successfully applied to log-linear models for syntactic disambiguation by Riezler & Vasserman (2004). The view in all the works cited is that property selection should not, or at least not only, be performed prior to training, but that the relevance of properties should be assessed somehow during training and their weight should only be adjusted away from zero, which is assumed to be the initial weight of all properties, if the property makes a relevant contribution to the model.

The advantage of incremental property selection over a frequency-based cut-off is that it makes a selection among all types of properties regardless of their frequency. It can thus discard properties that are relatively frequent, but do not contribute much to the classification task the model is trained for. Limitations of the incremental property selection techniques mentioned above are that (i) they are insensitive to data sparsity and (ii) they presuppose that the property set is only moderately redundant. We discuss below how we may take advantage of incremental property selection while addressing these limitations.

10.2.3 Correlational analysis

(Semi-)Automatic property construction on the basis of property templates gives rise to property sets that are highly redundant. In particular when the amount of training data is limited, a non-negligible number of properties are correlated very highly or are even entirely collinear. Properties are necessarily collinear when they are instantiations of different property templates, but happen to refer to the same tree or AVM configuration. Nevertheless, properties can also correlate very highly or even be collinear when they are only mildly related or even unrelated; this is then due to peculiarities of the training data which may be coincidental or in turn be due to peculiarities of the grammar that produced the training data.

An example of a pair of properties that are defined with the help of different property templates, but that refer to the same tree configuration are `cs_embedded VP[v,fin] 1` and `cs_sub_label VP[v,fin] VP[v,fin]`. Both properties count the number of `VP[v,fin]` nodes in a given c-structure that dominate another `VP[v,fin]` node; the dominance relation does not need to be immediate in this case. They are thus collinear by definition, which means that they would also be collinear if the grammar were modified or if we had substantially more training data.

An example of a pair of properties that are collinear due to constraints that the grammar imposes on the analyses are `cs_label CPfreerel[dp]` and

`cs_adjacent_label CPfreerel[dp] CPfreerelx[dp]`. The way the c-structure rules for nominal free relative clauses are written enforces all `CPfreerel[dp]` nodes to immediately dominate a `CPfreerelx[dp]` node and, vice versa, all `CPfreerelx[dp]` nodes to be immediately dominated by a `CPfreerel[dp]` node. The collinearity of these properties thus depends on the grammar, and they might result in no longer being collinear if the grammar were modified.

Finally, an example of a pair of properties that are ‘accidentally’ collinear in our set of training data are `cs_conj_nonpar 9` and `cs_conj_nonpar 10`. The reason for them being collinear is that no coordinated constituent in the training data is parallel at a depth of 9 levels below the coordinated node and, hence, none is at the depth of 10 levels below the coordinated node either. This is a ‘peculiarity’ of our training data, since the two properties might no longer be collinear in a larger set of training data, even if this specific ‘peculiarity’ is rather likely to be stable across sets of training data of varying sizes.

Although it is generally seen as one of the strong points of log-linear models that they do *not* require properties to be linearly independent, we see two reasons for which a less redundant and hence more compact set of properties can be considered to be superior to a more redundant (and hence less compact) one: efficiency and suitability for incremental property selection. The first reason holds for all kinds of models, regardless of the training regime that is applied for the estimation of their parameters. The second reason obviously holds only if incremental property selection is used in the training process. As we do employ incremental property selection (see Section 10.4), we thus believe that it is worth looking into ways of reducing redundancy in the set of properties used in our final log-linear model, since incremental property selection works best on “linguistic features sets with moderate redundancy” (Riezler & Vasserman 2004).

Detecting redundancies in a set of properties is straightforward. It is sufficient to perform a correlational analysis of the data resulting from property extraction. What is less straightforward is determining the measures to be taken in case a high correlation between two properties is observed. Nevertheless, we have developed a principled solution to this issue. While the solution is fully automatic for highly correlated but not entirely collinear properties, manual intervention by the grammar/property developer is required for entirely collinear properties.

This is due to the fact that, for pairs of properties that are entirely collinear in our set of training data, there is no information in the property values that would allow us to systematically retain one property and discard the other one. We thus suggest that a list of detailed criteria be established on the basis of which the grammar/property developer evaluates properties against each other. These criteria are based on the definition of the collinear properties and should obey more general principles, such as simplicity, on the basis of which the list of criteria can be completed if the need arises. Below is a tentative list of cri-

teria for choosing among collinear template-based properties. It would have to be completed according to the principles of simplicity and f-structure predominance if it were to be applied to all pairs of collinear properties that can be found in our set of properties.

- **Simplicity**

For efficiency considerations, we prefer properties that involve less structure over properties that involve more structure. Concretely, this means the following:

- A `cs_conj_nonpar` property of lesser depth of embedding is chosen over a `cs_conj_nonpar` property of greater depth.
- Properties based on `cs_label` are chosen over properties based on `cs_num_children`, `cs_adjacent_label`, `fs_attr_val`, `fs_adj_attrs` and `lex_subcat`.
- Properties based on `fs_attrs` are chosen over properties based on `cs_num_children`, `cs_adjacent_label`, `fs_attr_val`, `fs_adj_attrs` and `lex_subcat`.
- Properties based on `fs_attr_val` are chosen over properties based on `cs_num_children`, `cs_adjacent_label`, `fs_adj_attrs` and `lex_subcat`.
- Properties based on `cs_num_children` are chosen over properties based on `cs_adjacent_label`, `fs_adj_attrs` and `lex_subcat`.
- Properties based on `cs_adjacent_label` are chosen over properties based on `lex_subcat`.
- Properties based on `fs_adj_attrs` are chosen over properties based on `lex_subcat`.

- **F-structure rather than c-structure**

Since applications that may build on the output of the grammar will probably make use of f-structures and disregard c-structures and since we evaluate our f-structures rather than our c-structures, we prefer f-structure-based properties over c-structure-based properties. Concretely, this means the following:

- Properties based on `fs_attrs` are chosen over properties based on `cs_label`.
- Properties based on `fs_adj_attrs` are chosen over properties based on `cs_adjacent_label`.

- **Further criteria**

There will (probably) always be pairs of collinear properties among which it is impossible to choose on the basis of the aforementioned criteria. The most evident case is the one of collinear properties that belong to the same property family. Since there is no well-founded way of preferring one of these properties over the other, we just arbitrarily retain the one that comes first in alphabetical order and discard the other one.

For pairs of properties that are highly correlated, i.e. correlated at a level above a previously fixed threshold, but not entirely collinear, there is information in the property values available that can be used to systematically retain one property and discard the other. One possible criterion is the average correlation with all the remaining properties, in which case the property that is less correlated with the remaining properties would be retained and the other one, discarded; the motivation behind this approach would be to have a compact set of maximally dissimilar properties. Another possible criterion is correlation with the probability of the analysis, in which case the property that, considered on its own, is the better predictor would be chosen over the one that is less correlated with the probability of the analysis. Given that the calculation of the correlations between several thousands of properties is already extremely expensive in computational resources, we believe that the latter selection mechanism, which avoids summing over the correlations between all properties and the probability of the analysis under consideration, is more realistic.

Although we have developed a clear idea of how we can reduce the high redundancy observed in our set of properties and even implemented this idea, there is an important obstacle to obtaining results that would allow us to state (i) to what extent our set of properties is reduced by the elimination of collinear properties and (ii) what effect the choice of the correlation cutoff for the selection among highly correlated properties has on the compactness and the accuracy of the resulting models. This obstacle is computing time. Running our Perl implementation of the correlational analysis and the related measures to be taken on one CPU would take months, so that we will have to reimplement the code in C and vectorize it, so that it can be processed in parallel on a cluster of CPUs. The correlation analysis will thus be one of the first steps in future work on the log-linear model for disambiguation, but cannot be accomplished as part of this dissertation.

10.3 Regularization

Another way of addressing the problem of overfitting is to regularize property weights, i.e. to impose a prior on property weights during training. One commonly used prior is the Gaussian prior, which assumes that property weights are

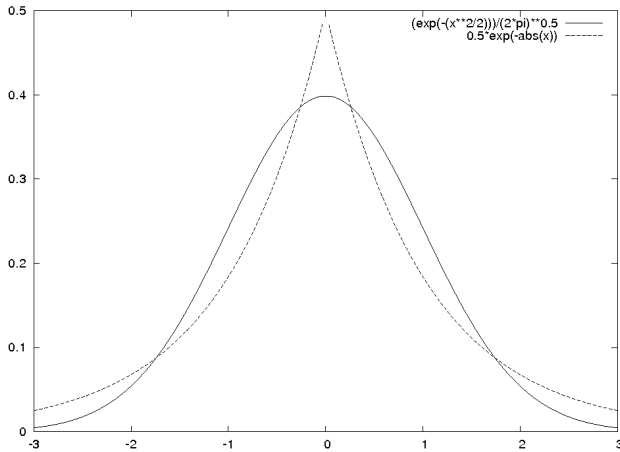


Figure 10.1: Gaussian distribution and Laplacian distribution with $\mu = 0$ and $\sigma = 1$

distributed in a Gaussian bell curve. Since a standardized zero-mean Gaussian bell curve like the one in Figure 10.1 is very flat around zero, the application of a Gaussian prior usually does not reduce the number of properties that are used for disambiguation, but it helps to prevent properties from being associated with extreme weights that are tailored too tightly to the training data. As Malouf & van Noord (2004) state it, “a Gaussian prior is used for more accurate models, and a frequency cutoff is used for more compact models”.

Riezler & Vasserman (2004) present alternative ways of regularizing log-linear models. These consist in imposing an upper limit on the number of non-zero-weighted properties (l_0 regularization), which is related to the frequency-based cutoff(s) discussed in Subsection 10.2.1, or in imposing a double-exponential Laplacian prior on property weights (l_1 regularization). The double-exponential prior is better suited for combined property selection and regularization, since it puts more probability mass near zero (and into the tails) than the Gaussian prior, as is illustrated in Figure 10.1. In other words, it causes more property weights to be estimated as zero, while not smoothing extreme weights as strongly as the Gaussian prior.

10.4 Combined regularization and property selection

For a number of systems, property selection and regularization have been combined (Riezler & Vasserman 2004, Malouf & van Noord 2004, van Noord 2006). However, there is an important difference in the exact way this is done in Riezler & Vasserman (2004) on the one hand and Malouf & van Noord (2004), van Noord (2006) on the other hand.

Malouf & van Noord (2004) and van Noord (2006) simply combine a frequency-based cutoff (of 2 according to their definition, which would be a cutoff of 3 according to our definition) and regularization by means of a Gaussian prior (with σ set to 1,000). In contrast, Riezler & Vasserman (2004) do not make use of any frequency-based cutoff in their best model because of the cutoff's ad hoc character, but employ n -best grafting combined with l_1 regularization, which in their experiments outperforms l_2 regularization, even if the latter is coupled with incremental feature selection.

The results of our experiments do not confirm that the combination of gradient-based property selection and regularization proposed by Riezler & Vasserman (2004) outperforms regularization by means of a Gaussian prior in terms of the accuracy of the resulting models, at least not if the latter model is trained on data to which a frequency-based cutoff was applied, as it is the case in Malouf & van Noord (2004) and van Noord (2006). Instead, the resulting models are equivalent in terms of the F-score measured on the held-out set (see Models 2 and 3 in Table 10.3 on p. 210), and on the test set (see Table 10.4 on p. 210), Model 3, i.e. the model resulting from the application of a frequency-based cutoff and regularization by means of a Gaussian prior, even performs significantly better than Model 2, which was trained along the lines of Riezler & Vasserman (2004). However, both Model 2 and Model 3 outperform the unregularized Model 1 significantly.

What may be the reason for this apparent contradiction between the results reported in Riezler & Vasserman (2004) and our results? We believe that it is the application of a frequency-based cutoff in (or rather prior to) the training of Model 3 as opposed to no such frequency-based cutoff in the training of Model 2. Model 2 may suffer from property weights that could not be estimated reliably because the corresponding properties occur too rarely in the training data. Although Riezler & Vasserman (2004) oppose a frequency-based cutoff on the one hand and gradient-based property selection on the other hand as 'competitors', these two techniques may actually be complementary. The frequency-based cutoff could ensure that the properties considered during training occur frequently enough for the estimation of the corresponding weights, which is something the gradient-based property selection mechanism is not sensitive to, while gradient-based property selection can further cut down the number of

	# prop.	compr.	F-score	error red.
unregularized MLE on				
standardized values (Model 1)	57,934	–	83.89	34.8%
n -best grafting combined				
with l_1 regularization (Model 2)	18,000	68.9%	84.26	43.0%
frequency-based cutoff of 3				
and l_2 regularization (Model 3)	9,152	84.2%	84.26	43.1%
frequency-based cutoff of 4 and n -best				
grafting with l_1 regularization (Model 4)	4,340	92.5%	84.45	47.3%

Table 10.3: Overall F-score (and corresponding error reduction) achieved by four different systems on the 371 TiGer DB structures of our held-out set

	# prop.	compr.	F-score	error red.
unregularized MLE on				
standardized values (Model 1)	57,934	–	82.55	42.0%
n -best grafting combined				
with l_1 regularization (Model 2)	18,000	68.9%	82.74	45.6%
frequency-based cutoff of 3				
and l_2 regularization (Model 3)	9,152	84.2%	83.08	52.3%
frequency-based cutoff of 4 and n -best				
grafting with l_1 regularization (Model 4)	4,340	92.5%	83.01	51.0%

Table 10.4: Overall F-score (and corresponding error reduction) achieved by four different systems on the 1,597 TiGer DB structures of our test set

properties retained in the resulting model by discarding frequent, but hardly discriminative properties.

The comparison of Model 4, i.e. the model for which a frequency-based cutoff was coupled with gradient-based property selection, with the two aforementioned models shows that the frequency-based cutoff does indeed have a role to play in the development of a compact model that generalizes well to unseen data. On the held-out set, Model 4 performs better than Model 3, even if the difference is only significant at a confidence level of 90%, and on the test set, Model 3 and Model 4 perform equally well, even with a confidence level as low as 70%. Both these models are significantly superior to the unregularized Model 1, and on the test set, we also observe a significant performance difference between Models 3 and 4 on the one hand and Model 2, which is regularized, but was trained on data to which no frequency-based cutoff was applied.

10.4 Combined regularization and property selection

Given that Models 3 and 4 do not differ significantly (at the generally used confidence level of 95%) in terms of accuracy, we may want to judge them with respect to their size. As noted already several times, more compact models are generally considered superior to less compact ones because they are more efficient when being applied. Considering this criterion, Model 4 is clearly superior to Model 3, as it comprises less than half as many properties.⁵ Although our results suggest that n -best grafting combined with l_1 regularization (Model 4) and l_2 regularization (Model 3) are equivalent with respect to the accuracy of the resulting models, there is thus a strong argument in favor of the combined property selection and l_1 regularization approach. Most importantly, we notice that the prior application of a frequency-based cutoff to the training data is essential for these approaches to yield models that generalize well to unseen data.

Besides the sparsity of properties, extreme redundancy in the set of properties is a further problem where the incremental property selection mechanism by Riezler & Vasserman (2004) needs to be supplemented. As these authors state, incremental property selection works only on moderately redundant property sets. However, the property set that we start out with is highly redundant; it would thus be desirable to reduce this redundancy, e.g. by the strategy that we have proposed in Subsection 10.2.3, prior to training proper.

Table 10.5 (p. 212) illustrates the effect of the different property selection strategies applied in the four models on different property families. Let us first consider the effect of gradient-based property selection without the prior application of a frequency-based cutoff, i.e. Model 2. We observe that the compression rates for the property families `cs_adjacent_label`, `fs_adj_attrs`, `isDef.SUBJ`, `DEP11`, `PATH`, `VerbsWith`, `[ANV]ADJUNCT`, `DEP21` and `DEP22` are below the overall compression rate of 68.9%. This means that properties belonging to these nine families are (on average) better predictors for the training data than properties that belong to one of the remaining families.

When we compare, for each property family, the compression rates of Model 2 and Model 4, we observe that the compression rate of Model 2 is superior for `cs_label`, `cs_num_children`, `cs_adjacent_label`, `fs_attrs`, `fs_attr_val`, `fs_adj_attrs`, `isDef.SUBJ`, `DEP11`, `VerbsWith` and `MOD_SPD`, whereas the inverse is true for the remaining property families. This is plausible, since properties belonging to the families just mentioned are relatively weakly specialized and are thus not likely to be heavily affected by a frequency-based cutoff. As the remaining, more specialized, property families are, in particular `DEP22`, they actually take over ‘work’ from the strongly discriminative but sparse properties that are discarded by means of the frequency-cutoff.

⁵It may be argued that the difference in the number of properties in the models is mainly due to the different frequency-based cutoffs applied for Models 3 and 4. This is partly true. However, incremental property selection does compress the model based on properties that satisfy a frequency-based cutoff of 4 by 32%.

(a) Template-based property families

model	cs_label # pr. compr.	cs_num_children # pr. compr.	cs_adjacent_label # pr. compr.	fs_attrs # pr. compr.	fs_attr_val # pr. compr.	fs_adj_attrs # pr. compr.	lex_subcat # pr. compr.
1	212 –	181	673 –	113 –	184 –	293 –	985 –
2	54 75%	38 79%	234 79%	25 65%	35 81%	95 81%	150 68%
3	171 19%	148 18%	453 18%	106 33%	142 23%	249 23%	246 15%
4	78 63%	68	257 62%	61 62%	62 66%	154 66%	106 47%

(b) Unlexicalized and rather mildly lexicalized property families

model	isDef.SUBJ # pr. compr.	DEP11 # pr. compr.	DEPTH-OF # pr. compr.	PATH # pr. compr.	Verbswith # pr. compr.	[ANV] ADJUNCT # pr. compr.	ab#bauen.DB J # pr. compr.
1	61 –	108	12 –	40 –	6 –	681 –	2,072 –
2	27 56%	58 46%	3 75%	27 33%	2 67%	325 52%	332 332
3	57 7%	97 10%	9 25%	29 28%	3 50%	284 58%	542 542
4	44 28%	71 34%	3 75%	21 48%	3 50%	186 73%	271 271

(c) Relatively heavily lexicalized property families

model	DEP21 # pr. compr.	F2 # pr. compr.	DEP12 # pr. compr.	ACTIVE # pr. compr.	DEP22 # pr. compr.	vor#sitzen.MOD_SPD # pr. compr.	MOD_SPD # pr. compr.
1	12,194 –	1,589	10,060 –	312 –	27,793 –	181 –	142 –
2	3,852 68%	246 85%	2,299 85%	31 90%	10,128 64%	11 94%	2 99%
3	2,605 79%	306 81%	1,957 81%	44 86%	1,648 94%	10 94%	7 95%
4	1,221 90%	103 94%	895 91%	10 97%	689 98%	2 99%	4 97%

Table 10.5: Numbers of discriminative properties broken down according to property families and different models

When we compare, for each property family, the compression rate of Model 3 to the compression rates of Models 2 and 4, we observe the same tendency of less specialized property families taking over ‘work’ from the more specialized families, but, as is to be expected in a model where the frequency-based cutoff is almost the only means of discarding properties, the exact compression rates for the different property families depends almost solely on the degree of specialization of the respective families and much less on their discriminative power. An example of a weakly specialized property family with relatively little discriminative power is the property family `fs_attrs`. Whereas Model 3 makes use of 94% of the properties belonging to this family, Model 4 only uses 54% of these properties.

10.5 Summary

This chapter has discussed several ways of preventing log-linear models from overfitting the training data, which is an important issue in the development of log-linear models based on tens or even hundreds of thousands of properties. We have shown that the usefulness of a frequency-based cutoff depends crucially on its exact definition and that a well-defined frequency-based cutoff can make an interesting contribution to the development of models that generalize well to unseen data by ensuring that properties occur often enough for reliable parameter estimation. Furthermore, our experiments confirm that regularized models, i.e. models during whose training a Gaussian or Laplacian distribution is assumed for property weights, outperform unregularized models and that iterative property selection can further reduce the number of properties used in a given model, without adversely affecting accuracy. Our best model in terms of accuracy and compactness/efficiency results from the application of a frequency-based cutoff of 4, followed by a training regime that combines regularization by means of a Laplacian prior and iterative property selection as described in Riezler & Vasserman (2004).

Part V
Conclusions

Chapter 11

Summary and Conclusions

11.1 Summary

In this dissertation, we have presented the disambiguation architecture used in the German *ParGram* LFG, which is now a stochastic unification-based grammar, and all steps involved in the development of the final system.

The first of these steps is the development of training and test data, presented in Chapters 4 and 5 respectively. We have shown how an existing tree-bank resource, namely the TIGER Corpus, can be used for the automatic construction of LFG training data and that, with some manual intervention, the resulting data can be completely disambiguated and hence be used as test data.

The next step consists in the fine-tuning of a filter for syntactic analyses based on so-called OT marks, which are elements of an additional projection specified by the grammar writer(s) as part of the symbolic grammar. Chapter 6 justifies this approach and documents the state of this filter at the inception of our work and Chapter 7 presents a method based on the Gradual Learning Algorithm (Boersma 1998) for fine-tuning the filter on the basis of corpus data. We have shown in this context that it is possible to adapt the filter in such a way that its filter fidelity achieves more than 95%, while filter efficiency is maintained at more than 60%. This is important because the purpose of the OT-mark-based disambiguation module shifts from being the only disambiguation device used in the system to merely reducing the number of analyses to be considered in the final disambiguation step and thus acting as a *pre-filter*.

The third step, finally, is the development of a log-linear model that selects the (*n*) best parse(s) among the ones that are preserved by the pre-filter. We have started this development by building a first model along the lines of Riezler et al. (2002) and Riezler & Vasserman (2004), i.e. a model based on the same types of properties and trained according to the same training regime as the log-linear model used in the English *ParGram* LFG. Then we have turned to ways of improving this first model. The machine learning machinery being

relatively well established, we have focussed on the importance of property design in this context. In a linguistically inspired manner, we have defined new properties that are intended to capture typical ambiguities that occur in German LFG parses. Nevertheless, we have also addressed the question of what the best strategies are to avoid overfitting, once the set of properties that are provided to the system is established. In this context, we have discussed in particular how (and why) a sound frequency-based cutoff can complement the iterative property selection technique proposed by Riezler & Vasserman (2004).

The two different sets of properties (the template-based properties and the extended set of properties) combined with different training regimes have given rise to a multitude of log-linear models. Table 11.1 repeats the figures concerning the size and the performance of the four models that are most important for our argumentation.

	# prop.	F-score	error red.
XLE template-based properties, <i>n</i> -best grafting combined with l_1 regularization	4,660	82.17	34.5%
all properties, unregularized MLE (Model 1)	57,934	82.55	42.0%
all properties, <i>n</i> -best grafting with l_1 regularization (Model 2)	18,000	82.74	45.6%
all properties that survive a frequency-b. cutoff of 4, <i>n</i> -best grafting with l_1 regularization (Model 4)	4,340	83.01	51.0%

Table 11.1: Overall F-score (and corresponding error reduction) achieved by four different systems on the 1,497 TiGer DB structures of our test set

11.2 Conclusions

We have shown that the TIGER Treebank, despite the hybrid annotation combining elements of phrase structure and dependencies, can successfully be used for the construction of labeled training data in the format that the German *Par-Gram* LFG produces. Nevertheless, it has to be kept in mind that the resolution of ambiguities that could only be resolved on the basis of information that is not available in the TIGER Treebank annotations (information concerning the decomposition of compounds, adjunct vs. argument status of a constituent annotated as having the function M0 (modifier) in the TIGER Treebank, etc.) can generally not be learned from these labeled data. All these aspects also hold

for other corpora annotated in the same format and according to the same or similar guidelines as the TIGER Treebank, such as the NEGRA Treebank.

Furthermore, we have created a dependency-based gold standard for German parsers, the TiGer Dependency Bank. Like the the construction of the training data, its creation involved the fully automatic conversion of TIGER Treebank graphs into f-structure charts. However, in order to obtain fully disambiguated representations with the same granularity of information as is encoded in the grammar output, the f-structure charts then had to be disambiguated, which involved several person months of manual labor. Therefore, the TiGer DB was limited to the corpus section from sentence 8,001 to sentence 10,000.

As for the disambiguation by means of OT marks, we have shown that the Gradual Learning Algorithm can be used to automatically learn the ranking of the OT marks from corpus data, but we have also found that the relative ranking of the OT marks does actually not have a big influence on the quality of the filter that the OT marks constitute. Fortunately, though, the Gradual Learning Algorithm also proved to be efficient as a means of identifying OT marks that are not reliable enough to be used in a pre-filter. With this method, the OT-mark-driven pre-filter can be tuned so that it preserves the intended analyses for more than 95% of the sentences, while still being a very efficient means for reducing the number of analyses produced for an average sentence.

In the context of stochastic disambiguation, we have demonstrated that property design is of utmost importance in the development of a disambiguation module. Our results indicate that it is a good idea to carry out property design in a linguistically inspired fashion, i.e. by referring to the theoretical literature that deals with soft constraints that are active in the language for which the system is developed. Property design thus requires a profound knowledge of the language under consideration (and the theoretical literature that deals with its syntax), and since the disambiguation module operates on the output of the symbolic grammar, a good knowledge of the grammar is certainly desirable, if not necessary, as well.

Weighting against each other the contributions of the different measures taken for improving the log-linear model used for disambiguation to the final model, we can conclude that property design is more important than property selection and/or regularization. Even the worst model based on all properties, which is the entirely unregularized Model 1 (see Table 11.1), performs significantly better than the best-performing model among those that are based on the template-based properties only. Moreover, property design can be carried out in a targeted way, i.e. properties can be designed in order to improve the disambiguation of grammatical relations that, so far, are disambiguated particularly poorly or that are of special interest for the task that the system's output is used for. In contrast, automatic property selection and regularization can only improve the overall accuracy of the model that is developed. By demonstrating that property design is the key to good log-linear models for syntactic disam-

bification, our work confirms that “specifying the features of a SUBG [stochastic unification-based grammar] is as much an empirical matter as specifying the grammar itself” (Johnson et al. 1999).

Regarding property selection and regularization, we conclude that both of these measures improve log-linear models for syntactic disambiguation significantly. According to our results and contrarily to what Riezler & Vasserman (2004) observe, there is no significant difference in terms of accuracy between models that were regularized using a Gaussian prior and models for which regularization was done with a Laplacian exponential prior. However, the exponential prior lends itself particularly well to the combination of regularization and property selection, which results in more compact models, which are to be favored for efficiency considerations.

We also conclude that a well-defined frequency cutoff does have a role to play in property selection, although some authors shun it altogether for being of an ad hoc nature. This argument has some validity, of course, but a ‘conservative’ cutoff of 3 or 4, apart from compressing models considerably, can actually prevent property weights from being learnt from sparse data and thus allow for significantly improved accuracy. For property selection beyond the application of such a ‘conservative’ cutoff, we argue in favor of iterative property selection as proposed by Riezler & Vasserman (2004). In their approach, property selection is actually a part of parameter estimation, so that it is not ad hoc, but has a solid mathematical foundation.

The two main challenges in the development of log-linear models used for the disambiguation of deep syntactic analyses are thus property design, i.e. the definition of the properties that are provided in the first place, and property selection, i.e. choosing among the properties provided the ones that are relevant for the disambiguation task. Property selection is something that can be integrated into the learning procedure, although we have shown that there is still some room for improvement. How to improve property selection is thus a topic to be addressed by machine learning experts. Property design, however, is, in our opinion, an activity that needs expert knowledge of the language dealt with and linguistic intuition, so that it should be carried out by (computational) linguists. Ideally, it is even the same person(s) who develop (or maintain) the symbolic grammar and the properties used for disambiguation.

11.3 Outlook to future work

In future work, we plan to carry out more experiments with properties based on auxiliary distributions and possibly other properties based on external resources (see Section 9.2). These experiments will consist in integrating more auxiliary distributions into the model, so that statistical information relating to more different kinds of dependencies will be available to the model, and in

trying out different ways of merging and normalizing the statistical information concerning individual dependencies. Our suspicion is that the exact way in which auxiliary distributions are integrated into a log-linear model as properties matters a lot for the usefulness of these properties.

In the field of property selection, we plan to carry out a correlational analysis of the properties as they are observed in our training data and then reduce the redundancy, which we know to exist in the data, according to the method presented in Subsection 10.2.3. We already know that this will at least allow for a further compaction of the model. In the best case, accuracy will also be improved, since we expect incremental property selection to work better on data that are less redundant than our current training data.

Finally, we are going to apply the same approach to the task of realization ranking, i.e. we are going to train a log-linear model for the selection of the most probable surface string generated from a given *f*-structure. This work will be heavily inspired by Velldal et al. (2004) and Velldal & Oepen (2005). For a start, we will base this new model on the same properties as are used for parse disambiguation, but, as purely *f*-structural properties are obviously irrelevant in the context of generation and all other properties have so far only been defined with parse disambiguation in mind as well, we will certainly define new properties expressly for realization ranking. Many of them will be very surface-oriented, so that they will hardly be relevant for parse disambiguation. Nevertheless, with all the ones that refer to *c*-structure configurations or to *c*-structure-to-*f*-structure mappings, we plan to train new log-linear models for parse disambiguation as well. The idea is to create crosstalk between the property design for parse disambiguation on the one hand and property design for realization ranking on the other hand and to evaluate what properties are relevant for both tasks and to what extent.

Summary in German – Zusammenfassung

Desambiguierung für einen linguistisch präzisen deutschen Parser

Diese Dissertation beschäftigt sich mit Methoden für die Auflösung von Mehrdeutigkeiten (Desambiguierung) in linguistisch präzisen und detaillierten syntaktischen Analysen, die von handgeschriebenen unifikationsbasierten Grammatiken wie der deutschen *ParGram*-LFG ausgegeben werden. Sie stellt die Architektur der Desambiguierungskomponente der deutschen *ParGram*-LFG vor, durch die diese ursprünglich rein symbolische Grammatik eine stochastische oder probabilistische unifikationsbasierte Grammatik (SUBG oder PUBG) geworden ist, sowie alle Schritte, die für die Entwicklung dieser Desambiguierungskomponente nötig waren.

Handgeschriebene Grammatiken sind zunächst immer rein symbolischer Natur. Dies bedeutet, dass sie keine Information hinsichtlich der Wahrscheinlichkeiten enthalten, mit denen die in ihnen enthaltenen Regeln angewandt werden. Aufgrund der vielfältigen und überaus häufigen syntaktischen Ambiguitäten (Mehrdeutigkeiten), die in den meisten Sätzen realer Texte vorkommen, gibt eine handgeschriebene unifikationsbasierte Grammatik für einen durchschnittlichen Satz immer mehrere Analysen aus. Für viele Sätze werden sogar Hunderte, Tausende oder gar Millionen verschiedener möglicher Analysen ermittelt.

Die allermeisten Anwendungen, für die Grammatiken entwickelt werden, wie z.B. Maschinelle Übersetzung, Informationsextraktion, Frage-Antwort-Systeme usw., können auf der Basis solch stark mehrdeutiger Analysen nicht weiterarbeiten. Daher werden für handgeschriebene Grammatiken Desambiguierungskomponenten benötigt, die es erlauben, aus allen Analysen, welche die symbolische Grammatik produziert, die wahrscheinlichste oder die n wahrscheinlichsten zu ermitteln. Der am weitesten verbreitete Ansatz für die Entwicklung solcher Desambiguierungskomponenten besteht darin, mit Hilfe von syntaktisch annotierten Korpusdaten, sogenannten Baumbanken, die inzwischen für eine ganze Reihe von Sprachen, u.a. das Deutsche, in beachtlichem

Zusammenfassung

Umfang zur Verfügung stehen, statistische Modelle zu trainieren, die aus den Korpusdaten auf der Basis von vorher definierten Lernmerkmalen Präferenzen für oder Dispräferenzen gegen gewisse (Teil-)Strukturen in den Analysen lernen. Auf der Basis dieser Präferenzen und Dispräferenzen kann dann eine Analyse (oder eine Menge von n Analysen) als die wahrscheinlichste selektiert werden.

Diese Dissertation führt zunächst in den Kapiteln 2 und 3 die Grundlagen für die dokumentierte Arbeit ein. Dann beschreibt sie in den Kapiteln 4 bis 10 die verschiedenen Arbeitsschritte, die zur Entwicklung der letztendlichen Desambiguierungskomponente erforderlich waren. Den Abschluss bilden im letzten Kapitel eine Zusammenfassung, Schlussfolgerungen sowie ein Ausblick auf zukünftige Arbeiten. Diese Zusammenfassung folgt dieser Gliederung.

Die deutsche *ParGram*-LFG

Die in Kapitel 2 vorgestellte deutsche *ParGram*-LFG ist eine komputationelle Grammatik zur Analyse freien deutschen Textes. Sie ist im Formalismus der Lexikalisch-Funktionalen Grammatik (LFG) (Kaplan & Bresnan 1982) definiert und wird mit Hilfe der Grammatikentwicklungs- und -verarbeitungsplattform *XLE* (Crouch et al. 2006) implementiert und verarbeitet.

Wie der Name schon sagt, ist die deutsche *ParGram*-LFG Mitglied der Familie der *ParGram*-Grammatiken (Butt et al. 2002). *ParGram* ist eine Initiative zur Entwicklung paralleler Lexikalisch-Funktionaler Grammatiken für eine Reihe typologisch und genetisch unterschiedlicher Sprachen (momentan Arabisch, Chinesisch, Deutsch, Englisch, Französisch, Japanisch, Madegassisch, Norwegisch, Türkisch, Ungarisch, Urdu, Vietnamesisch und Walisisch), wobei 'parallel' hier bedeutet, dass sich die Ausgaberepräsentationen von übersetzungsäquivalenten Sätzen in verschiedenen Sprachen so wenig wie möglich unterscheiden.

Die LFG ist ein Grammatikformalismus, der zwei Repräsentationsebenen benutzt, um syntaktische Eigenschaften von Sätzen auszudrücken: die Konstituentenstruktur (c-Struktur) und die funktionale Struktur (f-Struktur). C-Strukturen sind kontextfreie Bäume, die die Konstituenten und ihre lineare Abfolge im jeweiligen Satz festhalten. F-Strukturen repräsentieren grammatische Relationen und morphosyntaktische Merkmale. In übereinzelsprachlicher Hinsicht wird angenommen, dass übersetzungsäquivalente Sätze in verschiedenen Sprachen sich auf der c-Struktur-Ebene evtl. erheblich unterscheiden, während die entsprechenden f-Strukturen sich sehr ähnlich sind. Betrachten wir als Beispiel die c- und f-Strukturen, die die englische und die deutsche *ParGram*-LFG jeweils für die folgenden beiden Sätze ausgeben:

(11.1) Der Brief wird morgen angekommen sein.

(11.2) The letter will have arrived tomorrow.
 Der Brief wird haben angekommen morgen.

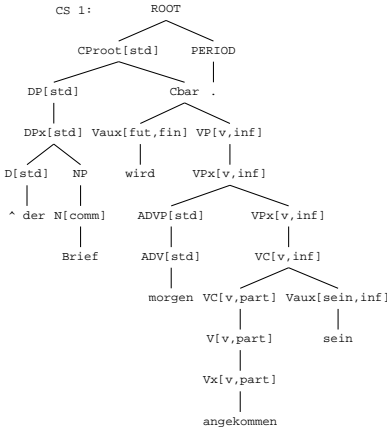


Abbildung 11.1: C-Struktur zu (11.1)

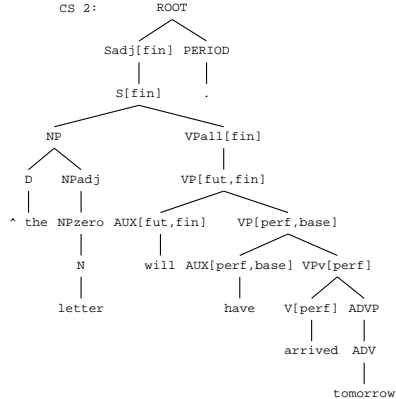


Abbildung 11.3: C-Struktur zu (11.2)

"Der Brief wird morgen angekommen sein."

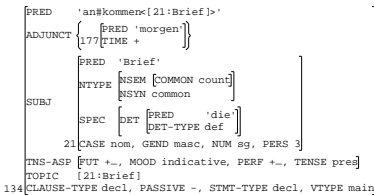


Abbildung 11.2: F-Struktur zu (11.1)

"The letter will have arrived tomorrow."

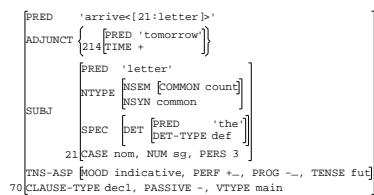


Abbildung 11.4: F-Struktur zu (11.2)

Die deutsche *ParGram*-LFG wurde zunächst *phänomenbasiert* entwickelt. Diese Arbeit ist zum Teil in Butt et al. (1999) und sehr detailliert in Dipper (2003) dokumentiert. Die phänomenbasierte Entwicklung der Grammatik führte dazu, dass für die allermeisten Konstruktionen der deutschen Syntax linguistisch fundierte Analysen produziert werden konnten. Allerdings hatte die Grammatik, wenn sie auf Zeitungskorpora angewendet wurde, nur eine Abdeckung (Anteil der Sätze, für die mindestens eine den ganzen Satz

überspannende Analyse ermittelt werden kann) zwischen 35% (Dipper 2003) und 55% (Rohrer 2003, pers. Mitt.). Diese niedrige Abdeckung stellte ein ernsthaftes Hindernis für die Verwendung der Grammatik in Anwendungen dar. Darum wurde dann eine Phase der *korpusbasierten* Grammatikentwicklung eingeleitet, in der die Abdeckung auf über 80% erhöht wurde (Rohrer & Forst 2006a,b).

Neben der niedrigen Abdeckung war die erhebliche Ambiguität durchschnittlicher Ausgaberepräsentationen ein zentrales Hindernis für die Verwendbarkeit der Grammatik. Um hier für Abhilfe zu sorgen, musste daher die ursprüngliche deutsche *ParGram*-LFG durch eine Desambiguierungskomponente ergänzt werden. Die einzelnen Schritte in der Entwicklung dieser Desambiguierungskomponente sind Gegenstand der vorliegenden Arbeit.

Desambiguierung für unifiktionsbasierte Grammatiken

Die Desambiguierung ist in unifiktionsbasierten Grammatiken immer eine Art Nachverarbeitung, da auf Grund der nicht-lokalen Abhängigkeiten, die durch die Unifikation entstehen können, die Grammatikregeln selber nicht mit Wahrscheinlichkeiten assoziiert werden können, wie es z.B. in probabilistischen kontextfreien Grammatiken der Fall ist. Die Desambiguierung für unifiktionsbasierte Grammatiken besteht also letztlich darin, aus allen durch den symbolischen Teil der Grammatik spezifizierten Analysen eines Satzes die wahrscheinlichste (oder die n wahrscheinlichsten) zu selegieren.

Auf dem Gebiet der Desambiguierung für unifiktionsbasierte Grammatiken gibt es zwei Ansätze, die in der letztlichen Version der deutschen *ParGram*-LFG beide zum Einsatz kommen. Zum einen handelt es sich dabei um die optimalitätstheoretisch inspirierte Desambiguierung, die in Frank et al. (2001) vorgestellt wird, zum anderen um die Verwendung eines log-linearen Modells, das auf der Basis von Lernmerkmalen, die aus einem Trainingskorpus extrahiert werden, trainiert wird. Der optimalitätstheoretisch inspirierte Ansatz besteht darin, dass der Grammatikschreiber Lexikoneinträge und Regeln mit sogenannten Optimalitätsmarks (OT-Marks) versieht. Sowohl die OT-Marks selbst als auch ihre Position in der OT-Mark-Hierarchie werden dabei ursprünglich vom Grammatikschreiber auf der Grundlage seiner Intuition spezifiziert. Verbreiteter ist der probabilistische Ansatz, der darin besteht, auf Korpusdaten ein log-lineares Modell zu trainieren, das auf der Basis von Lernmerkmalen und diesen Merkmalen zugeordneten Gewichten die Wahrscheinlichkeit alternativer Analysen ermittelt.

Die Idee, probabilistische Versionen unifiktionsbasierter Grammatiken zu entwickeln, geht zurück auf Eisele (1994) und Brew (1995). Allerdings sind

die darin vorgeschlagenen Modelle keine sauberen Wahrscheinlichkeitsmodelle und die aus den Ansätzen resultierenden Schätzwerte für Regelwahrscheinlichkeiten maximieren nicht die Wahrscheinlichkeit des jeweils verwendeten Trainingskorpus. Abney (1997) schlägt daher vor, log-lineare Modelle der folgenden Form für die Desambiguierung unifikationsbasierter Analysen zu verwenden:

$$P_{\lambda}(\omega) = \frac{e^{\sum_{j=1}^m \lambda_j \cdot f_j(\omega)}}{\sum_{\omega' \in \Omega} e^{\sum_{j=1}^m \lambda_j \cdot f_j(\omega')}}$$

$\omega = (x, y)$ und $\omega' = (x', y')$ sind dabei Paare von Analysen und Zeichenketten. $\lambda = (\lambda_1 \dots \lambda_m)$ ist ein Vektor von m Lernmerkmalsgewichten, $f = (f_1 \dots f_m)$ ist ein Vektor von m Lernmerkmalsfunktionen und Ω ist die Menge aller Paare von Analysen und Zeichenketten, die zur von der Grammatik definierten Sprache gehören.

Johnson et al. (1999) argumentieren überzeugend, dass die Parameterschätzung für ein solches Modell, das die gemeinsame Wahrscheinlichkeit von Zeichenketten und Analysen maximiert, bei unifikationsbasierten Grammatiken mit großer Abdeckung nicht machbar ist, da die von diesen Grammatiken definierten Sprachen nicht endlich sind und somit nicht über alle ihre Elemente summiert werden kann. Als Alternative schlagen sie vor, log-lineare Modelle zu verwenden, die die bedingte Wahrscheinlichkeit von Analysen bei gegebenen Zeichenketten maximieren. Diese Modelle haben folgende Form:

$$P_{\lambda}(x|y) = \frac{e^{\sum_{j=1}^m \lambda_j \cdot f_j(x,y)}}{\sum_{x' \in X(y)} e^{\sum_{j=1}^m \lambda_j \cdot f_j(x',y)}}$$

x ist dabei eine mögliche Analyse der Zeichenkette y , λ ist wieder ein Vektor mit m Merkmalsgewichten, f ist wieder ein Vektor von Merkmalsfunktionen und $X(y)$ ist die Menge aller möglichen Analysen der Zeichenkette y .

Abgesehen davon, dass die Parameter $\lambda_1 \dots \lambda_m$ dieser Modelle auch für sehr große Grammatiken auf der Basis von Korpusdaten geschätzt werden können, gibt es auch gute Argumente dafür, dass sie für diese Aufgabe angemessener sind, denn Zeichenketten sind ja tatsächlich gegeben, wenn es darum geht zu desambiguieren. Aus diesen beiden Gründen finden diese bedingten Wahrscheinlichkeitsmodelle nun standardmäßig Anwendung in Desambiguierungsmodulen für Grammatiken mit tiefer Analyse.

Der Rest von Kapitel 3 befasst sich mit Arbeiten, die zum Ziel hatten, zwei wichtige Einschränkungen der gerade vorgestellten log-linearen Modelle zu umgehen: die beschränkte Verfügbarkeit von Trainingsdaten und die Überanpassung des Modells an die Trainingsdaten.

Zusammenfassung

Johnson & Riezler (2000) zeigen, dass es problemlos möglich ist, sogenannte Auxiliarverteilungen in überwacht trainierte log-lineare Modelle einzubinden. Auxiliarverteilungen sind statistische Informationen, die auf der Basis von mit flachen Verarbeitungsverfahren annotierten Korpora akquiriert wurden. Diese flach annotierten Korpora können in ihrem Umfang weit über den des detaillierter annotierten Trainingskorpus hinaus gehen. Schließlich sei noch angemerkt, dass Auxiliarverteilungen keine Wahrscheinlichkeitsverteilungen sein müssen, auch wenn ihr Name dies suggeriert.

Die Resultate von Johnson & Riezler (2000) sprechen nicht unbedingt dafür, dass die Einbindung von Auxiliarverteilungen in ein log-lineares Modell die Desambiguierungsqualität tatsächlich signifikant verbessert. Allerdings sind Auxiliarverteilungen in log-linearen Modellen für die Realisierungsauswahl bei der Generierung (Velldal & Oepen 2005, Nakanishi et al. 2005) sehr erfolgreich eingesetzt worden, so dass die Möglichkeit, statistische Informationen aus unannotierten Daten in ein überwacht trainiertes Modell zu integrieren, sicherlich als interessante Option zu betrachten ist.

Dem Problem der Überanpassung des Modells an die Trainingsdaten kann durch zwei verwandte Strategien begegnet werden: durch die Regularisierung von potentiell extremen Merkmalsgewichten und durch das Auswählen einer Teilmenge aus der Menge der anfänglich zur Verfügung gestellten Lernmerkmale. Die Regularisierung extremer Merkmalsgewichte wird in den meisten Systemen, wie bereits in Johnson et al. (1999), durch die Anwendung eines Gaußschen Priors realisiert. Eine Auswahl möglichst relevanter Lernmerkmale findet bei der Entwicklung der meisten log-linearen Modelle für die syntaktische Desambiguierung jedoch nicht statt. Nur in der Entwicklung von Systemen, die auf sehr vielen, d.h. Zehntausenden, Hunderttausenden oder gar Millionen, Lernmerkmalen beruhen, wird manchmal zur Reduzierung ein Schwellwert angewandt; Merkmale, deren Frequenz den Schwellwert nicht übersteigt, werden beim Training nicht berücksichtigt.

Riezler & Vasserman (2004) stellen einen kombinierten Regularisierungs- und Merkmalsauswahlmechanismus vor, der auf einem Laplaceschen Prior beruht. Der große konzeptionelle Vorteil dieser Methode gegenüber der Anwendung eines frequenzbasierten Schwellwerts und der anschließenden Regularisierung durch einen Gaußschen Prior besteht darin, dass die Auswahl der relevantesten Lernmerkmale als Teil des Trainings betrachtet wird und somit eine Auswahl unter allen Merkmalen, unabhängig von ihrer Frequenz, getroffen wird.

Baumbankumwandlung für die Erstellung von Trainingsdaten

Um ein log-lineares Modell überwacht zu trainieren, benötigt man annotierte Trainingsdaten, wobei es ausreicht, wenn eine echte Teilmenge der von der Grammatik gelieferten Analysen als annotationskompatibel identifiziert werden können. Man benötigt nicht unbedingt ein vollständig desambiguiertes Korpus. Allerdings muss das Trainingskorpus natürlich dasselbe Repräsentationsformat verwenden wie die Daten, auf die das darauf trainierte Modell schließlich angewendet werden soll.

Als Ausgangsbasis für die Erstellung unserer Trainingsdaten dient die TIGER-Baumbank (Brants et al. 2003). Allerdings unterscheiden sich TIGER-Graphen teilweise erheblich von den entsprechenden f-Strukturen, so dass eine recht aufwändige Umwandlung von TIGER-Graphen in f-Strukturen stattfinden muss (siehe auch Forst (2003a,b)). Diese wird zusätzlich erschwert durch Unterschiede in der Granularität der Repräsentationen und durch verschiedene Lemmatisierungskonventionen. Eine vollautomatische Eins-zu-eins-Abbildung von TIGER-Baumbank-Graphen in f-Strukturen ist daher nicht möglich. Stattdessen wird eine Umwandlung von TIGER-Graphen in potentiell ambige f-Struktur-Repräsentationen, sogenannte f-Struktur-Charts, vorgenommen. Diese geschieht in folgenden Schritten :

- Umwandlung der in TIGER XML kodierten TIGER-Graphen in Attribut-Wert-Matrizen mit dem gleichen Informationsgehalt, dargestellt im Prolog-Interface-Format von XLE. Bei diesem Schritt werden noch keine Ambiguitäten eingeführt.
- Lemmatisierung mit Hilfe eines Perl-Programms, das sich der von der deutschen *ParGram*-LFG verwendeten Finite-State-Morphologie bedient. Bei Lemmatisierungsambiguitäten werden alle Alternativen in der gepackten Repräsentation festgehalten.
- Umwandlung der lemmatisierten im Prolog-Interface-Format von XLE repräsentierten Attribut-Wert-Matrizen in f-Struktur-Charts durch die Anwendung von zum Teil komplexen Termersetzungsregeln, wie sie das Termersetzungs-system von XLE benutzt. TIGER-Kantenlabel, die mehreren f-Struktur-Attributen entsprechen können, werden durch optionale Regeln gehandhabt. Dies hat den Vorteil, dass keine unfundierten Entscheidungen für die eine oder die andere Entsprechung gefällt werden müssen. Andererseits führt die Anwendung von optionalen Regeln dazu, dass die resultierenden Repräsentationen oft sehr ambig sind.

Nach der Erstellung von f-Struktur-Charts auf der Basis von TIGER-Graphen findet schließlich ein Abgleich statt zwischen diesen Repräsentationen und

den von der Grammatik produzierten Analysen. Dabei können sich folgende Umstände ergeben: (i) Sind alle Analysen mit mindestens einer der aus der TIGER-Annotation abgeleiteten f-Strukturen kompatibel, kann der Satz nicht für das Training verwendet werden, da die TIGER-Annotation keine Unterscheidung von erwünschten und unerwünschten Lesarten zulässt. (ii) Genauso verhält es sich, wenn keine der Grammatikanalysen mit einer der aus der TIGER-Annotation abgeleiteten f-Strukturen kompatibel ist. (iii) Die Anzahl der Lesarten in den aus der TIGER-Annotation abgeleiteten Repräsentationen kombiniert mit der Anzahl der Lesarten in den Grammatikanalysen ist so hoch, dass ein Abgleich nicht mit überschaubarem Aufwand machbar ist. (iv) Das Produkt der Anzahl der Lesarten in den aus der TIGER-Annotation abgeleiteten Repräsentationen und der Anzahl der Lesarten in den Grammatikanalysen überschreitet einen festgelegten Schwellwert nicht und aus der Menge der Grammatikanalysen ist eine echte Teilmenge mit jeweils mindestens einer der aus der TIGER-Annotation abgeleiteten f-Strukturen kompatibel. Von den 37 546 Sätzen der 50 000 TIGER-Korpus-Sätze, die die damals verwendete Grammatikversion parsen konnte, erfüllten 9269 diese Bedingung. Da einige davon jedoch zu dem Korpusabschnitt gehören, den wir für die Evaluierung verwenden, erhalten wir ein Trainingskorpus von 8881 f-Struktur-Charts.

Die TiGer Dependency Bank – Ein dependenz-basierter Gold Standard für deutsche Parser

Für die Evaluierung der verschiedenen log-linearen Modelle, mit denen wir experimentiert haben, sowie für die Evaluierung des Gesamtsystems benötigen wir Testdaten. Diese sollten hinsichtlich der Granularität der in ihnen repräsentierten Information mit den Analysen, die die deutsche *ParGram*-LFG und evtl. andere deutsche Parser mit tiefer Analyse ausgeben, möglichst vergleichbar sein.

Für den zufällig ausgewählten Abschnitt des TIGER-Korpus von Satz 8001 bis Satz 10 000 wurde daher auf der Basis der zuvor aus den TIGER-Graphen abgeleiteten gepackten f-Struktur-Repräsentationen die TiGer Dependency Bank Forst et al. (2004) erstellt. Sie umfasst 1868 Strukturen, die aus sogenannten Dependenztripeln bestehen; für 132 Sätze konnten keine Strukturen erstellt werden, weil die 'Sätze' entweder nur aus einem Wort oder aus syntaktisch nicht zusammenhängenden Wortfolgen bestehen. Die Erstellung erfolgte durch die manuelle Desambiguierung der aus den TIGER-Graphen abgeleiteten f-Struktur-Charts, wobei für einen Teil der Sätze die Ambiguität der f-Struktur-Charts auch schon zuvor durch den Abgleich mit den von der Grammatik produzierten Analysen reduziert werden konnte. Die manuelle Arbeit war nötig, um die bei der Lemmatisierung und der Umwandlung von aus LFG-Sicht

ambigen TIGER-Kantenlabeln eingeführten Ambiguitäten wieder aufzulösen. Die durch die vollständige Desambiguierung der f-Struktur-Charts entstehenden f-Strukturen konnten dann schließlich vollautomatisch in Abhängigkeitstripel überführt werden.

Die Strukturen in der TiGer Dependency Bank benutzen zum Großteil die gleichen Label für grammatische Relationen wie die TIGER-Baumbank. Allerdings gibt es auch einige Unterschiede, wovon die wichtigsten hier zusammengefasst sind.

- Komposita werden bei der Lemmatisierung zerlegt.
- Für die Festlegung von Subjekten von funktional kontrollierten Infinitiven und prädikativ verwendeten Elementen wird Koindizierung verwendet.
- Die Relation NK wird zugunsten von feineren Unterscheidungen aufgegeben. Determinanten werden abhängig von ihrer Subkategorie unter der Relation det oder quant aufgehängt, attributive Adjektive unter der Relation mo, enge Appositionen unter der Relation app und quantifizierte Elemente unter der Relation measured.
- Alle prädikativen Argumente werden, unabhängig vom Element, das sie subkategorisiert, unter der Relation pd aufgehängt.
- Präpositionale und adverbiale Argumente direktonaler, lokaler und modaler Natur werden statt unter der Kante MO unter den Relationen op_dir, op_loc und op_manner aufgehängt.

Die Satz (11.3) zugeordnete Beispielstruktur auf Seite 232 illustriert einige dieser Unterschiede zum entsprechenden TIGER-Graphen.

(11.3) Privatmuseum muß weichen
Private museum must leave
'Private museum must leave'¹

Manuell definierte OT-Mark-Hierarchien für die Desambiguierung

Die Desambiguierung auf der Basis manuell definierter OT-Mark-Hierarchien (Frank et al. 2001) ist in der Ausgangsgrammatik der einzige verwendete Desambiguierungsmechanismus. Er besteht darin, dass in Grammatikregeln

¹s8595

Zusammenfassung

```
case(Museum~1, nom),
compd_form(Museum~1, Privatmuseum),
gend(Museum~1, neut),
mod(Museum~1, privat~1001),
mood(müssen~0, indicative),
num(Museum~1, sg),
oc_inf(müssen~0, weichen~3),
pers(Museum~1, 3),
sb(müssen~0, Museum~1),
sb(weichen~3, Museum~1),
tense(müssen~0, pres)
tiger_id(müssen~0, 2)
```

Abbildung 11.5: Dependenztripelrepräsentation zu (11.3)

und Lexikoneinträgen durch vom Grammatikschreiber spezifizierte Annotationen sogenannte OT-Marks projiziert werden und diese nach dem Aufbau aller möglichen Analysen für den jeweiligen geparsten Satz zur Evaluierung der verschiedenen Lesarten herangezogen werden. Bei den OT-Marks kann es sich sowohl um Dispräferenz- als auch um Präferenzmarks handeln. Sie sind alle hierarchisch zueinander geordnet, wobei sie auch in Mengen von gleich starken OT-Marks zusammengefasst werden können. Zur Bewertung der Analysen wird dann die OT-Mark-Hierarchie von oben nach unten durchlaufen, und bei jedem Schritt bleiben die Analysen als potentiell optimale übrig, die am wenigsten Vorkommen (im Falle eines Dispräferenzmarks) bzw. am meisten Vorkommen (im Falle eines Präferenzmarks) haben. Dieser Ansatz ist der einzige uns bekannte Ansatz zur Desambiguierung, der vollkommen manuell spezifiziert wird.

Dieses Verfahren ist sehr erfolgreich in verschiedenen *ParGram*-Grammatiken eingesetzt worden, um die Zahl der produzierten Analysen zu reduzieren. Allerdings können OT-Marks nur sehr lokale Phänomene erfassen und auch der von der Optimalitätstheorie übernommene Evaluierungsmechanismus setzt der Flexibilität des Ansatzes verhältnismäßig enge Grenzen. Schließlich sei erwähnt, dass die genauen Auswirkungen von OT-Marks und ihrer relativen Gewichtung für den Grammatikschreiber schwer vorherzusehen sind. Die Desambiguierung auf der Basis von OT-Marks ist daher nicht geeignet für eine vollständige Desambiguierung der ambigen Ausgabe des symbolischen Teils der Grammatik. Zur Reduzierung der Ambiguität dieser Ausgabe ist sie jedoch sehr geeignet.

Korpusbasiertes Erlernen von OT-Mark-Hierarchien

Angesichts der Tatsache, dass sich die Desambiguierung auf der Basis von OT-Mark-Hierarchien zwar als sehr nützlich zur Reduzierung der Ambiguität der Grammatikausgabe erweist, die manuell spezifizierte OT-Mark-Hierarchie in der Ausgangsgrammatik andererseits aber auch immer wieder erwünschte Lesarten als suboptimal beurteilt, schien die Idee vielversprechend, die Hierarchie der vom Grammatikschreiber spezifizierten OT-Marks aus Korpusdaten maschinell zu erlernen.

Wir wenden daher den Graduellen Lernalgorithmus (GLA) aus der Stochastischen Optimalitätstheorie (Boersma 1998) auf die aus unseren Trainingsdaten extrahierten OT-Tableaux an. Die aus den Daten erlernte Hierarchie führt zu einer etwas besseren Desambiguierung der Analysen unserer Grammatik, aber der Unterschied zur manuell spezifizierten Hierarchie ist nicht sehr groß. Glücklicherweise stellt sich der GLA jedoch auch als ein hervorragendes Werkzeug zur Ermittlung unzuverlässiger OT-Marks heraus. Dadurch können wir diese OT-Marks deaktivieren; dies führt zwar dazu, dass mehr Analysen als optimal bewertet werden, stellt aber gleichzeitig sicher, dass keine erwünschten Lesarten als suboptimal eliminiert werden. Schließlich wird für die verbleibenden OT-Marks mittels des GLA aus den Korpusdaten die optimale OT-Mark-Hierarchie gelernt. Der Vorfilter, den diese OT-Mark-Hierarchie darstellt, filtert für weniger als 5% der Sätze die erwünschte Lesart als suboptimal aus, d.h. seine Filterverlässlichkeit liegt über 95%, während er weiterhin über 60% der Analysen ausfiltert.

Ein erstes log-lineares Modell

In Kapitel 8 stellen wir das erste log-lineare Modell vor, das wir auf unseren Trainingsdaten nach dem Beispiel von Riezler et al. (2002) und Riezler & Vasserman (2004) trainiert haben. Wir gehen dabei insbesondere auf die verwendeten Lernmerkmale ein, berichten aber auch, welche Trainingsmethode genau angewendet wurde.

Die in diesem log-linearen Modell verwendeten Lernmerkmale beruhen alle auf den Merkmalstemplates, die XLE (Crouch et al. 2006) zur Verfügung stellt und die erlauben, die auf ihnen beruhenden Merkmale direkt aus den Grammatikanalysen zu extrahieren. Selbstverständlich wurden die Templates entsprechend den in der Grammatik verwendeten c-Struktur-Kategorien und f-Struktur-Attributen und ihren Werten instanziiert.

Das Modell wurde dann auf den 8881 f-Struktur-Chart-Paaren unseres Trainingskorpus mit dem kombinierten Regularisierungs- und Merkmalsauswahlmechanismus von Riezler & Vasserman (2004) trainiert. Die Parameter für diesen Mechanismus wurden auf den 371 TiGer DB-Strukturen unseres Held-out-

Sets angepasst. Schließlich wurde es mit Hilfe der Software von Crouch et al. (2002) auf den 1497 TiGer-Strukturen unseres Testsets evaluiert.

Die Evaluierung ergibt eine F-Metrik von 82,17% auf grammatischen Relationen und morphosyntaktischen Merkmalen bzw. von 74,69% nur auf grammatischen Relationen. Dies entspricht einer Fehlerreduktion von 34,5% bzw. 31,0%, was etwas unter der Fehlerreduktion von 36% liegt, die für das in der englischen *ParGram*-LFG verwendete log-lineare Modell berichtet wird.

Entwicklung von Lernmerkmalen für die Desambiguierung deutscher LFG-Analysen

Ausgehend von der Beobachtung, dass typischerweise in deutschen LFG-Analysen auftretende Ambiguitäten oft gar nicht von den auf XLE-Templates beruhenden Lernmerkmalen erfasst werden können, wurde in einem zentralen Arbeitsschritt der Entwicklung neuer Lernmerkmale für die Desambiguierung deutscher LFG-Analysen besondere Aufmerksamkeit geschenkt. Die Vorgehensweise war hierbei bewusst linguistisch inspiriert, d.h. es wurde die linguistische Literatur zu Wortstellungsphänomenen allgemein und zur Wortstellung im Deutschen und verwandten Sprachen im besonderen konsultiert. Dann wurden Lernmerkmale für alle potentiellen Regularitäten entwickelt, die vom symbolischen Teil der Grammatik nicht erfasst werden, weil es sich eben oft nur um statistische Tendenzen ('weiche' Beschränkungen) und nicht um kategorische Regeln ('harte' Beschränkungen) handelt. Auch wurden Auxiliarverteilungen, die auf der Basis von einem sehr großen Korpus erlernt wurden, als Lernmerkmale in das log-lineare Modell aufgenommen. All diese Lernmerkmale sind in Kapitel 9 dokumentiert.

Technisch realisiert wurde die Definition der neuen Merkmale mit Hilfe von Termersetzungsregeln, wie sie schon bei der Baubankumwandlung (Kapitel 4) zum Einsatz gekommen sind. Mit ihrer Hilfe werden die von der Grammatik erstellten c-/f-Struktur-Paare auf gewisse c-/f-Struktur-Konfigurationen abgeprüft und die gefundene Information dann so in den Strukturen repräsentiert, dass sie über das Merkmalstemplate `fs_attr_val` extrahiert werden kann. In diesem Kontext sei erwähnt, dass viele der neu eingeführten Lernmerkmale sich gleichzeitig auf die c-Struktur und auf die f-Struktur beziehen, indem sie z.B. die lineare Abfolge von grammatischen Funktionen ermitteln. Information dieser Art kann mittels der auf XLE-Templates basierenden Merkmale nicht aus Analysen extrahiert werden, obwohl sie sich, wie wir sehen, als sehr nützlich erweist für die Desambiguierung von syntaktischen Analysen in einer Sprache mit relativ freier Wortstellung wie dem Deutschen.

Training, Anpassung der Hyperparameter und Evaluierung wurden nach dem gleichen Muster durchgeführt wie beim ersten von uns trainierten Modell.

Durch die zusätzliche Information in den neu entwickelten Lernmerkmalen ergibt sich eine verbesserte F-Metrik von 83,01% auf grammatischen Relationen und morphosyntaktischen Relationen bzw. 75,74% nur auf grammatischen Relationen, was einer Fehlerreduktion von 51,0% bzw. 46,5% entspricht. Damit liegt die Fehlerreduktion eindeutig über der des ersten Modells und nun auch eindeutig über der von 36%, die für das in der englischen *ParGram*-LFG verwendete log-lineare Modell berichtet wird.

Regularisierung und Auswahl der Lernmerkmale

Neben der Entwicklung geeigneter Lernmerkmale ist die weitere zentrale Herausforderung bei der Entwicklung eines log-linearen Modells für die syntaktische Desambiguierung die Vermeidung der Überanpassung des Modells an die Trainingsdaten. Zu diesem Zweck werden bei der Entwicklung der in der Literatur erläuterten Modelle zwei Strategien angewendet: (i) die Regularisierung des Modells und (ii) das Treffen einer Auswahl unter allen zur Verfügung gestellten Merkmalen. Die Regularisierung verhindert durch die Annahme einer Gaußschen oder Laplaceschen Verteilung der Merkmalsgewichte, dass Lernmerkmale mit extremen Gewichten assoziiert werden. Die Auswahl einer Teilmenge von Merkmalen aus der Menge aller zur Verfügung gestellten trägt durch die oft beträchtliche Reduzierung der Anzahl der letztlich verwendeten Merkmale zur Vermeidung der Überanpassung bei. Außerdem ermöglicht sie die Entwicklung kompakterer und somit effizienterer Modelle.

Eine üblicherweise verwendete Strategie zur Auswahl einer Teilmenge von Merkmalen besteht darin, eine Mindestfrequenz festzulegen, mit der Merkmale in den Trainingsdaten vorkommen müssen. Wir stellen mehrere Varianten dieser Strategie aus der Literatur vor und kommen zu dem Schluss, dass diese Strategie, wenn sie bis ins Detail sinnvoll definiert ist, sehr erfolgreich dafür sorgen kann, dass Merkmalsgewichte nicht auf der Basis von zu wenigen und daher oft nicht repräsentativen Merkmalsvorkommen gelernt werden.

Eine weitere Strategie, die unserer Kenntnis nach bisher allerdings nur in Riezler & Vasserman (2004) zum Einsatz gekommen ist, besteht darin, Regularisierung und Merkmalsauswahl zu kombinieren. Gegenüber der Anwendung einer Mindestfrequenz, die in gewisser Weise ad hoc ist und nicht garantiert, dass die beibehaltenen Lernmerkmale tatsächlich bessere Indikatoren für die gewünschte Analyse eines Satzes sind als die eliminierten, hat die Kombination von Regularisierung und Merkmalsauswahl den Vorteil, dass die Auswahl unter allen Lernmerkmalen unabhängig von ihrer Frequenz und stattdessen gradientbasiert, d.h. auf der Basis ihres Beitrags zur Desambiguierung, stattfindet.

Allerdings berücksichtigt die kombinierte Regularisierungs- und Merkmalsauswahlmethode von Riezler & Vasserman (2004) die Frequenz von Lernmerkmalen überhaupt nicht. Dies ist insofern problematisch, als auch Merkmale,

Zusammenfassung

die nur in einem einzigen Satz des Trainingskorpus vorkommen, beim Training berücksichtigt werden, sofern sie in dem jeweiligen Satz gute Indikatoren für die intendierte Analyse sind. Daher schalten wir eine von uns definierte Anwendung einer Mindestfrequenz vor die Kombination von Regularisierung und Merkmalsauswahl von Riezler & Vasserman (2004). Die in Tabelle 11.2 zusammengefassten Ergebnisse zeigen, dass dieses Verfahren das Modell ergibt, das gleichzeitig am akkuratesten und am kompaktesten ist.

	Anz. Merk.	F-Metrik	Fehlerred.
auf XLE-Templates basierende Merkmale, kombinierte Regularisierung und Merkmalsauswahl	4660	82,17	34,5%
alle Merkmale, unregularisiert	57934	82,55	42,0%
alle Merkmale, kombinierte Regularisierung und Merkmalsauswahl	18000	82,74	45,6%
alle Merkmale, die in mind. 4 Sätzen diskriminant sind, kombinierte Regularisierung und Merkmalsauswahl	4340	83,01	51,0%

Tabelle 11.2: Von vier verschiedenen Systemen auf den 1497 TiGer DB-Strukturen unseres Testsets erreichte F-Metrik und entsprechende Fehlerreduktion sowie Anzahl der verwendeten Merkmale

Schlussfolgerungen und Ausblick auf zukünftige Arbeiten

Aus den Ergebnissen der verschiedenen Arbeitsschritte ziehen wir folgende Lehren:

- Das TIGER-Korpus kann zur Erstellung von Trainingsdaten für ein log-lineares Modell zur Desambiguierung deutscher LFG-Analysen verwendet werden. Allerdings stellen die Unterschiede in den Lemmatisierungskonventionen und hinsichtlich der Granularität der repräsentierten Information zwischen TIGER-Graphen einerseits und *ParGram*-f-Strukturen andererseits insofern ein Hindernis dar, als in den Trainingsdaten manche Ambiguitäten unaufgelöst bleiben und zur Erstellung der Testdaten ein nicht unerheblicher manueller Aufwand notwendig ist.
- Aus den Ergebnissen der Experimente zum Erlernen von OT-Mark-Hierarchien aus Korpusdaten schließen wir, dass die relative Gewichtung

der OT-Marks eine untergeordnete Rolle spielt. Allerdings lässt sich der Graduelle Lernalgorithmus aus der Stochastischen Optimalitätstheorie sehr erfolgreich dazu einsetzen, OT-Marks zu identifizieren, die für die Verwendung in einem Vorfilter nicht zuverlässig genug sind. Durch die Deaktivierung dieser unzuverlässigen OT-Marks steigern wir die Zuverlässigkeit des Filters auf über 95%. Gleichzeitig rechtfertigt sich die Verwendung des OT-Mark-Vorfilters durchaus noch, denn er eliminiert auch nach der Deaktivierung einiger OT-Marks noch mehr als 60% der von der Grammatik produzierten Lesarten.

- Aus den Zahlen in Tabelle 11.2 schließen wir, dass die Entwicklung von Lernmerkmalen für die Qualität des auf ihnen beruhenden log-linearen Modells für die syntaktische Desambiguierung von höchster Bedeutung ist, denn das schlechteste Modell, das die neuen Merkmale mitbenutzt, erzielt eine signifikant bessere F-Metrik als das beste der Modelle, die nur die auf XLE-Templates basierenden Merkmale benutzen.
- Eine weitere Schlussfolgerung ist, dass Regularisierung und Merkmalsauswahl nichtsdestotrotz eine bedeutende Rolle in der Entwicklung von log-linearen Modellen spielen, die auf ungesehenen Daten ähnlich gute Resultate wie auf den Trainingsdaten erzielen. Am besten schneidet in unseren Experimenten dabei die Kombination einer vorsichtig gewählten Mindestfrequenz für Lernmerkmale mit dem Regularisierungs- und Merkmalsauswahlmechanismus von Riezler & Vasserman (2004) ab. Dabei sei allerdings angemerkt, dass die genaue Definition der Mindestfrequenz eine große Rolle für die Gültigkeit dieser Aussage spielt.

Zukünftige Arbeiten sehen wir vor allem auf dem Gebiet der Einbindung von weiteren Auxiliärverteilungen als Lernmerkmalen und im Bereich der Datenvorbereitung für die inkrementelle Merkmalsauswahl durch eine Korrelationsanalyse zwischen Lernmerkmalen. Schließlich werden wir die Herangehensweise, die wir erfolgreich für die Entwicklung eines Desambiguierungsmoduls für die deutsche *ParGram*-LFG angewandt haben, auch auf die Aufgabe der Realisierungsauswahl bei der Generierung anwenden. Wir gehen davon aus, dass der veränderte Blickwinkel in diesem neuen Arbeitsschritt neue Impulse geben wird für die Entwicklung weiterer linguistisch fundierter Lernmerkmale. Idealerweise tragen diese dann wiederum auch zu einer verbesserten Desambiguierung bei.

Bibliography

- Abeillé, Anne (ed.). 2003. *Building and using syntactically annotated corpora*. Dordrecht, Netherlands: Kluwer.
- Abney, Steven. 1997. Stochastic attribute-value grammars. *Computational Linguistics* 23:597–617.
- Aissen, Judith. 2003. Differential Object Marking: Iconicity vs. Economy. *Natural Language and Linguistic Theory* 21:435–483.
- Boersma, Paul. 1998. *Functional Phonology. Formalizing the interactions between articulatory and perceptual drives*. PhD thesis, University of Amsterdam.
- Brants, Sabine, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. 2003. TIGER: Linguistic Interpretation of a German Corpus. *Journal of Language and Computation* Special Issue.
- Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol, Bulgaria.
- Brants, S., and S. Hansen. 2002. Developments in the TIGER annotation scheme and their realization in the corpus. In *Proceedings of the Third Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Spain.
- Brants, Thorsten, Brigitte Krenn, Wojciech Skut, and Hans Uszkoreit. 1997. Das NEGRA-Annotationsschema. Negra project report, Computational Linguistics, Saarland University, Saarbrücken, Germany.
- Brants, Thorsten, Wojciech Skut, and Hans Uszkoreit. 1999. Syntactic annotation of a German newspaper corpus. In *Proceedings of the ATALA Treebank Workshop*, pp. 69–76, Paris, France.
- Bresnan, Joan. 2001. *Lexical-Functional Syntax*, volume 16 of *Textbooks in Linguistics*. Oxford, UK: Blackwell.
- Brew, Chris. 1995. Stochastic HPSG. In *Proceedings of EAACL-95*.

Bibliography

- Butt, Miriam, Helge Dyvik, Tracy H. King, Hiroshi Masuichi, and Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of COLING-2002 Workshop on Grammar Engineering and Evaluation*, pp. 1–7.
- Butt, Miriam, Tracy Holloway King, María-Eugenia Niño, and Frédérique Segond. 1999. *A Grammar-Writer's Cookbook*. Stanford, CA: CSLI Publications.
- Cahill, Aoife, Michael Burke, Martin Forst, Ruth O'Donovan, Christian Rohrer, Josef van Genabith, and Andy Way. 2005. Treebank-Based Multilingual Unification-Grammar Resources. *Research in Language and Computation*.
- Cahill, Aoife, Mairead McCarthy, Josef van Genabith, and Andy Way. 2002. Automatic Annotation of the Penn-Treebank with LFG F-Structure Information. In A. Lenci, S. Montemagni, & V. Pirelli (eds.), *Proceedings of the LREC 2002 Post-Conference Workshop on Linguistic Knowledge Acquisition and Representation - Bootstrapping Annotated Language Data*, Paris, France.
- Carroll, John, Guido Minnen, and Edward Briscoe. 2003. Parser evaluation: using a grammatical relation annotation scheme. In Abeillé (2003), pp. 299–316.
- Carroll, John, Guido Minnen, and Ted Briscoe. 1999. Corpus annotation for parser evaluation. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora (LINC '99)*, Bergen, Norway.
- Charniak, Eugene, and Mark Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, Ann Arbor, MI.
- Clark, Stephen, and James R. Curran. 2004. Parsing the WSJ using CCJ and Log-Linear Models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*, Barcelona, Spain.
- Crouch, Dick, Mary Dalrymple, Ron Kaplan, Tracy King, John Maxwell, and Paula Newman. 2006. XLE documentation. Technical report, Palo Alto Research Center, Palo Alto, CA.
- Crouch, Richard, Ronald M. Kaplan, Tracy H. King, and Stefan Riezler. 2002. A comparison of evaluation metrics for a broad-coverage parser. In *Proceedings of the LREC Workshop 'Beyond PARSEVAL—Towards improved evaluation measures for parsing systems'*, pp. 67–74, Las Palmas, Spain.
- Dalrymple, Mary. 2001. *Lexical Functional Grammar*, volume 34 of *Syntax and Semantics*. New York, NY: Academic Press.

- Dalrymple, Mary, Ron Kaplan, John T. Maxwell, and Annie Zaenen (eds.). 1995. *Formal Issues in Lexical-Functional Grammar*. Stanford, CA: CSLI Publications.
- Daum, Michael, Kilian Foth, and Wolfgang Menzel. 2004. Automatic transformation of phrase treebanks to dependency trees. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-2004)*, Lisbon, Portugal.
- Della Pietra, Stephen, Vincent Della Pietra, and John Lafferty. 1997. Inducing Features of Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19:380–393.
- Dipper, Stefanie. 2003. *Implementing and Documenting Large-scale Grammars – German LFG*. PhD thesis, IMS, University of Stuttgart. Arbeitspapiere des Instituts für Maschinelle Sprachverarbeitung (AIMS), Volume 9, Number 1.
- Dorna, Michael, Anette Frank, Josef van Genabith, and Martin Emele. 1998. Syntactic and semantic transfer with f-structures. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING '98) and the 36th Annual Meeting of the Association for Computational Linguistics (ACL'98) (COLING-ACL'98)*, Montréal, Québec, Canada.
- Dubey, Amit. 2004. *Statistical Parsing for German: Modeling syntactic properties and annotation differences*. PhD thesis, Saarland University, Saarbrücken, Germany.
- Eisele, Andreas. 1994. Towards probabilistic extensions of constraint-based grammars. Technical Report Technical Report Deliverable R1.2.B, DYANA-2.
- Falk, Yehuda. 2001. *Lexical-Functional Grammar: An Introduction to Parallel Constraint-Based Syntax*. Number 126 in CSLI Lecture Notes. Stanford, CA: CSLI Publications.
- Forst, Martin. 2003a. Treebank Conversion – Creating a German f-structure bank from the TIGER Corpus. In *Proceedings of the LFG03 Conference*, Saratoga Springs, NY. CSLI Publications.
- Forst, Martin. 2003b. Treebank Conversion – Establishing a testsuite for a broad-coverage LFG from the the TIGER Treebank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora (LINC '03)*, Budapest.
- Forst, Martin, Núria Bertomeu, Berthold Crysmann, Frederik Fouvry, Silvia Hansen-Schirra, and Valia Kordoni. 2004. Towards a dependency-based gold standard for German parsers – The TiGer Dependency Bank. In *Proceedings of the COLING Workshop on Linguistically Interpreted Corpora (LINC '04)*, Geneva, Switzerland.

Bibliography

- Forst, Martin, Jonas Kuhn, and Christian Rohrer. 2005. Corpus-based learning of OT constraint rankings for large-scale LFG grammars. In *Proceedings of the 10th International LFG Conference (LFG'05)*, Bergen, Norway. CSLI Publications.
- Foth, Kilian, Michael Daum, and Wolfgang Menzel. 2004. A broad-coverage parser for German based on defeasible constraints. In *KONVENS 2004, Beiträge zur 7. Konferenz zur Verarbeitung natürlicher Sprache*, pp. 45–52, Vienna, Austria.
- Frank, Anette. 2000. Automatic F-structure Annotation of Treebank Trees. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the Fifth International Conference on Lexical-Functional Grammar*, Berkeley, CA. CSLI Publications.
- Frank, Anette. 2001a. Treebank Conversion. Converting the NEGRA treebank to an LTAG grammar. In *Proceedings of the Workshop on Multi-layer Corpus-based Analysis*, Iasi, Romania.
- Frank, Anette. 2001b. Treebank Conversion for LTAG Grammar Extraction. In *Proceedings of the Workshop on Linguistically Interpreted Corpora (LINC) 2001*, Leuven, Belgium.
- Frank, Anette, Tracy Holloway King, Jonas Kuhn, and John T. Maxwell. 2001. Optimality Theory Style Constraint Ranking in Large-Scale LFG Grammars. In Peter Sells (ed.), *Formal and Empirical Issues in Optimality Theoretic Syntax*, pp. 367–397. Stanford, CA: CSLI Publications.
- Frey, Werner. 1993. *Syntaktische Bedingungen für die semantische Interpretation*. Berlin, Germany: Akademie Verlag.
- Helbig, Gerhard, and Joachim Buscha. 2001. *Deutsche Grammatik – Ein Handbuch für den Ausländerunterricht*. Berlin and Munich, Germany: Langenscheidt.
- Hockenmaier, Julia. 2003. *Data and Models for Statistical Parsing with Combinatory Categorical Grammar*. PhD thesis, University of Edinburgh, Edinburgh, UK.
- Johnson, Mark, Stuart Geman, Stephen Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic "unification-based" grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics 1999*, College Park, MD.
- Johnson, Mark, and Stefan Riezler. 2000. Exploiting auxiliary distributions in stochastic unification based grammars. In *Proceedings of the 1st Conference of North American Chapter of the ACL*, Seattle, WA.

- Kaplan, Ronald M., and Joan Bresnan. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations*, pp. 173–281. Cambridge, MA: The MIT Press. Reprinted in Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell, and Annie Zaenen, eds., *Formal Issues in Lexical-Functional Grammar*, 29–130. Stanford: Center for the Study of Language and Information. 1995.
- Kaplan, Ronald M., Stefan Riezler, Tracy King, John T. Maxwell, Alexander Vasserman, and Richard Crouch. 2004. Speed and Accuracy in Shallow and Deep Stochastic Parsing. In *Proceedings of the Human Language Technology Conference and the 4th Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'04)*, Boston, MA.
- King, Tracy, Martin Forst, Jonas Kuhn, and Miriam Butt. 2005. The Feature Space in Parallel Grammar Writing. *Research in Language and Computation*.
- King, Tracy Holloway, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. 2003. The PARC 700 Dependency Bank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora (LINC '03)*, Budapest, Hungary.
- Kountz, Manuel. 2006. Extraktion von Dependenztripeln aus der TIGER-Baumbank. Studienarbeit, Universität Stuttgart, Institut für Maschinelle Sprachverarbeitung.
- Lenerz, Jürgen. 1977. *Zur Abfolge nominaler Satzglieder im Deutschen*. Number 5 in *Studien zur deutschen Grammatik*. Tübingen, Germany: Narr.
- Malouf, Robert, and Gertjan van Noord. 2004. Wide Coverage Parsing with Stochastic Attribute Value Grammars. In *Proceedings of the IJCNLP-04 Workshop "Beyond Shallow Analyses - Formalisms and statistical modeling for deep analyses"*.
- Manning, Christopher D. 2003. Probabilistic Syntax. In Rens Bod, Jennifer Hay, & Stefanie Jannedy (eds.), *Probabilistic Linguistics*, pp. 289–341. Cambridge, MA: MIT Press.
- Manning, Christopher D., and Hinrich Schütze. 1999. *Foundations of Statistical Language Processing*. Cambridge, MA: MIT Press.
- Maxwell, John T., and Ronald M. Kaplan. 1993. The interface between phrasal and functional constraints. *Computational Linguistics* 19(4):571 – 589.

Bibliography

- Mengel, Andreas, and Wolfgang Lezius. 2000. An XML-based representation format for syntactically annotated corpora. In *Proceedings of the International Conference on Language Resources and Evaluation*, pp. 121–126, Athens, Greece.
- Miyao, Yusuke, Takashi Ninomiya, and Jun'ichi Tsujii. 2004. Corpus-oriented grammar development for acquiring a Head-Driven Phrase Structure Grammar from the Penn Tree Bank. In *Proceedings of the First International Joint Conference on Natural Language Processing (IJCNLP-04)*, pp. 684–693, Hainan Island, China.
- Miyao, Yusuke, and Jun'ichi Tsujii. 2002. Maximum entropy estimation for feature forests. In *Proceedings of the Human Language Technology Conference*, San Diego, CA.
- Nakanishi, Hiroko, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proceedings of IWPT 2005*.
- O'Donovan, Ruth, Michael Burke, Aoife Cahill, Josef van Genabith, and Andy Way. 2005a. Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II and Penn-III Treebanks. *Computational Linguistics* pp. 329–365.
- O'Donovan, Ruth, Aoife Cahill, Josef van Genabith, and Andy Way. 2005b. Automatic Acquisition of Spanish LFG Resources from the CAST3LB Treebank. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the Tenth International Conference on LFG (LFG '05)*, Bergen, Norway. CSLI Publications.
- Oepen, Stephan, Daniel Flickinger, J. Tsujii, and Hans Uszkoreit (eds.). 2002. *Collaborative Language Engineering. A Case Study in Efficient Grammar-Based Processing*. Stanford, CA: CSLI Publications.
- Perkins, Simon, Kevin Lacker, and James Theiler. 2003. Grafting: Fast Incremental Feature Selection by Gradient Descent in Function Space. *Journal of Machine Learning Research* pp. 1333–1356.
- Prince, Alan, and Paul Smolensky. 1993. Optimality Theory: Constraint Interaction in Generative Grammar. Technical Report Technical Report 2, Rutgers University Center for Cognitive Science.
- Riezler, Stefan, Tracy Holloway King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics 2002*, Philadelphia, PA.

- Riezler, Stefan, Jonas Kuhn, Detlef Prescher, and Mark Johnson. 2000. Lexicalized Stochastic Modelling of Constraint-based Grammars Using Log-linear Measures and EM Training. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics 2000*, Hong Kong, China.
- Riezler, Stefan, and Alexander Vasserman. 2004. Gradient feature testing and l_1 regularization for maximum entropy parsing. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP'04)*, Barcelona, Spain.
- Rohrer, Christian, and Martin Forst. 2006a. Broad-coverage Grammar Development – How Far Can It Go? In Miriam Butt, Mary Dalrymple, & Tracy H. King (eds.), *Intelligent Linguistic Architectures – Variations on Themes By Ronald M. Kaplan*. CSLI Publications.
- Rohrer, Christian, and Martin Forst. 2006b. Improving coverage and parsing quality of a large-scale LFG for German. In *Proceedings of the Language Resources and Evaluation Conference (LREC-2006)*, Genoa, Italy.
- Rosén, Victoria, Koenraad de Smedt, Helge Dyvik, and Paul Meurer. 2005. TREPIL: Developing Methods and Tools for Multilevel Treebank Construction. In *Proceedings from the Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*, University of Barcelona, Spain.
- Sadler, Louisa, Josef van Genabith, and Andy Way. 2000. Automatic F-Structure Annotation from the AP Treebank. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the Fifth International Conference on Lexical-Functional Grammar*, Berkeley, CA. CSLI Publications.
- Schiehlen, Michael. 2003. Combining Deep and Shallow Approaches in Parsing German. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan.
- Schiehlen, Michael. 2004. Annotation Strategies for Probabilistic Parsing in German. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland.
- Schulte im Walde, Sabine. 2003a. A Collocation Database for German Verbs and Nouns. In *Proceedings of the 7th Conference on Computational Lexicography and Text Research*, Budapest, Hungary.
- Schulte im Walde, Sabine. 2003b. *Experiments on the Automatic Induction of German Semantic Verb Classes*. PhD thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart. Published as AIMS Report 9(2).

Bibliography

- Skut, Wojciech, Thorsten Brants, Brigitte Krenn, and Hans Uszkoreit. 1998. A Linguistically Interpreted Corpus of German Newspaper Text. In *Proceedings of the Conference on Language Resources and Evaluation LREC-98*, pp. 705–711, Granada, Spain.
- Spreyer, Kathrin, and Anette Frank. 2005a. Projecting RMRS from TIGER Dependencies. In *Proceedings of the HPSG 2005 Conference*, Lisbon, Portugal.
- Spreyer, Kathrin, and Anette Frank. 2005b. The TIGER 700 RMRS Bank: RMRS Construction from Dependencies. In *Proceedings of the Workshop on Linguistically Interpreted Corpora (LINC) 2005*, Jeju Island, Korea.
- Telljohann, Heike, Erhard Hinrichs, and Sandra Kübler. 2003. Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z). Technical report, University of Tübingen, Tübingen, Germany.
- Tesar, Bruce B., and Paul Smolensky. 1998. Learnability in Optimality Theory. *Linguistic Inquiry* 29:229–268.
- Tibshirani, Robert. 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B* 58:267–288.
- Toutanova, Kristian, Christopher D. Manning, Stuart M. Shieber, Dan Flickinger, and Stephan Oepen. 2002. Parse disambiguation for a rich HPSG grammar. In *First Workshop on Treebanks and Linguistic Theories (TLT2002)*, pp. 253–263.
- Uszkoreit, Hans. 1987. *Word Order and Constituent Structure in German*. Stanford, CA: CSLI Publications.
- van Genabith, Josef, Anette Frank, and Andy Way. 2001. Treebank vs. Xbar-based Automatic Feature-Structure Annotation. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the Sixth International Lexical-Functional Grammar Conference*, Hong Kong, China. CSLI Publications.
- van Genabith, Josef, Andy Way, and Louisa Sadler. 1999. Semi-Automatic Generation of F-Structures from Tree Banks. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the Fourth International Conference on Lexical-Functional Grammar*, Manchester, UK. CSLI Publications.
- van Noord, Gertjan. 2004. Error Mining for Wide-Coverage Grammar Engineering. In *Proceedings of ACL 2004*, Barcelona, Spain.
- van Noord, Gertjan. 2006. *At Last Parsing Is Now Operational*. In Piet Mertens, Cedrick Fairon, Anne Dister, & Patrick Watrin (eds.), *TALN06. Verbum Ex Machina. Actes de la 13e conférence sur le traitement automatique des langues naturelles*, pp. 20–42, Leuven, Belgium.

- Velldal, Erik, and Stephan Oepen. 2005. Maximum entropy models for realization ranking. In *Proceedings of the 10th Machine Translation Summit*, pp. 109–116, Phuket, Thailand.
- Velldal, Erik, Stephan Oepen, and Dan Flickinger. 2004. Paraphrasing treebanks for stochastic realization ranking. In *Proceedings of the 3rd Workshop on Treebanks and Linguistic Theories*, pp. 149–160, Tübingen, Germany.
- Wahlster, Wolfgang (ed.). 2000. *Verbmobil: Foundations of Speech-to-Speech Translation*. Heidelberg, Germany: Springer.
- Yoshida, Kazuhiro. 2005. Corpus-Oriented Development of Japanese HPSG Parsers. In *Proceedings of the 43rd ACL Student Research Workshop*, Ann Arbor, MI.
- Zinsmeister, Heike, Jonas Kuhn, and Stefanie Dipper. 2002. TIGER TRANSFER - Utilizing LFG Parses for Treebank Annotation. In Miriam Butt & Tracy H. King (eds.), *Proceedings of the LFG02 Conference*, pp. 111–120, National Technical University of Athens, Greece. CSLI.

