Projecting LFG F-Structures from Chunks
– or (Non-)Configurationality from a Different View-Point –

Anette Frank

Language Technology Lab
German Research Center for Artificial Intelligence
DFKI GmbH
66123 Saarbrücken, Germany
Anette.Frank@dfki.de

Proceedings of the LFG03 Conference

University at Albany, State University of New York

Miriam Butt and Tracy Holloway King (Editors)

2003

CSLI Publications

http://csli-publications.stanford.edu/

Projecting LFG F-Structures from Chunks
– or (Non-)Configurationality from a Different View-Point –

Anette Frank
Language Technology Lab
German Research Center for Artificial Intelligence
DFKI GmbH
66123 Saarbrücken, Germany
Anette.Frank@dfki.de

Abstract

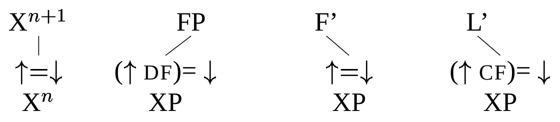
In this paper we pursue two related goals: First, we establish a conceptual link between chunk-based syntactic structures as typically assumed in shallow parsing approaches, as opposed to principle-based syntactic structures as assumed in theoretical linguistics research. This conceptual link emerges from the study of configurational vs. non-configurational languages, their analysis within the LFG framework, and the observation of diverse strategies for ambiguity resolution across this spectrum of (non-)configurational language types. Second, we show how shallow analyses as usually employed in practical NLP applications can be refined to deliver full-fledged syntactic representations, by designing an architecture for LFG f-structure projection from chunk-based syntactic analyses.

In line with our two-fold goal we will demonstrate that principles for f-structure projection from chunks are similar – modulo specific attachment constraints – to the LFG analysis of non-configurational languages. In essence, then, besides the design of a new style of robust LFG processing from chunk-based analyses, our investigation offers theoretical insight into the kind of abstraction (i.e. underspecification) employed in shallow analysis, and how it can be formalised within the LFG framework. In particular, we will show how to adapt the LFG analysis of non-configurational case-stacking languages in terms of inside-out functionality to the projection of full-fledged f-structures from chunk-based analyses of configurational languages.

1 Introduction

LFG theory has reached a high degree of sophistication and coverage, both at the level of theoretical linguistic research into diverse languages and language types, and in the area of computational processing, by providing efficient algorithms and system implementations, as well as wide-coverage computational LFG grammars (cf. [Bresnan, 2001, Dalrymple, 2001, Butt et al., 2002, Riezler et al., 2002, Cahill et al., 2003]). Still, none of the computationally tractable grammatical frameworks – be it LFG, HPSG, TAGs, or CG – is usually employed in practical NLP applications. The main reason being that despite tremendous achievements, we haven't, as of today, reached full coverage of natural language, as found in actual usage. Likewise, while computational processing has made enormous progress in speed, robustness, and analysis selection, efficiency and robustness are often (feared to be) not sufficient to lend themselves to practical NLP applications.

Starting with [Abney, 1996], we observe the emergence of a paradigm of shallow syntactic processing that restricts itself to the detection of base constituents (NP, PP, etc.), so-called 'chunks'. This type of shallow syntactic processing achieves robustness and speed by abstraction from fine-grained and ambiguity-prone syntactic distinctions, such as the specification of local attachment relations between chunks, or long-distance relationships. We are thus confronted with a



$\text{DF} = (\ \text{TOP} \mid \text{FOC} \mid \text{SUBJ}\ )$
$\text{CF} = (\ \text{SUBJ} \mid \text{OBJ} \mid \text{OBJ}_\Theta \mid \text{OBL}_\Theta \mid \text{COMP}\ )$

Figure 1: Structure-function-mappings [Bresnan, 2001]

tension between linguistically motivated 'deep' syntactic analysis on the one hand, and 'shallow' syntactic analysis, which is developing largely independently from theoretical linguistic research.

In this paper we pursue two related goals: First, we establish a conceptual link between chunk-based structures as assumed in shallow parsing, as opposed to the principle-based syntactic structures assumed in theoretical linguistic research. This link emerges from the study of configurational vs. non-configurational languages, the analysis of these languages in the LFG framework, and the observation of diverse strategies for ambiguity resolution within this spectrum of (non-)configurational language types. Second, we show how shallow analysis as usually employed in practical NLP applications can be refined to deliver full-fledged syntactic representations, by designing an architecture for LFG f-structure projection from chunk-based analyses.

In line with our two-fold goal, we will demonstrate that principles for f-structure projection from chunks are similar – modulo specific attachment constraints – to the LFG analysis of non-configurational languages. In essence, then, besides the design of a new style of robust LFG processing from shallow analyses, our investigation offers theoretical insight into the kind of abstraction (i.e. underspecification) employed in shallow analysis, and how it can be formalised within the LFG framework.

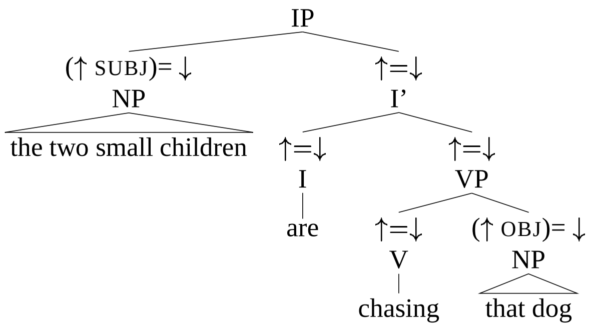
The remainder of this paper is structured as follows. In Section 2 we review the analysis of configurational and non-configurational languages in the LFG framework, considering in particular the interactions of (non-)configurationality, morphological marking and ambiguity resolution. In Section 3 we briefly characterise the complementary natures of shallow vs. 'deep' syntactic analysis in computational linguistics. Building on an existing cascaded shallow parsing architecture that combines a stochastic topological parser for German with chunk parsing, we develop a novel account to combine the complementary shallow and deep paradigms, by designing an architecture to project full-fledged LFG f-structures from chunk-based shallow analyses. In Section 4 we show how to resolve the underspecified attachment of chunks in a fully specified (disjunctive) f-structure, by use of inside-out functional uncertainty equations. In contrast to typical non-configurational languages, though, these are subject to specific adjacency constraints. Section 5 presents some conclusions.

2 (Non-)Configurationality and Ambiguity

Lexical-Functional Grammar accounts for the analysis of a wide spectrum of language types, ranging from configurational to non-configurational languages. Within its multi-level projection architecture, the c-structure allows for the flexible encoding of a wide variety of surface syntactic properties across languages, while the f-structure representation encodes functional syntactic properties that are largely shared across typologically distinct languages. General principles of c-structure encoding (X-bar theory) and principles of structure-function mappings (cf. Fig. 1) encode a principle-based mapping between c-structure and f-structure representations.

2.1 Morphology competes with Syntax

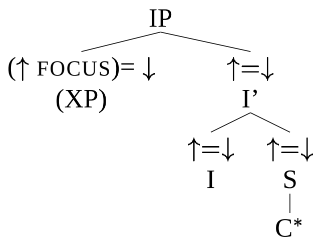
Configurational languages typically exhibit rather rigid word order constraints, and do in general not permit discontinuous realisation of constituents. Moreover, configurational languages usually don't possess overly rich systems of morphological marking. Endocentric c-structure and structure-function mapping principles jointly account for these characteristic properties of configurational languages, through a predominantly structural encoding of grammatical functions via language specific structure-function associations and ordering principles. Thus, in these languages the c-structure–f-structure mapping is largely determined by positional criteria, such as the association of the SUBJ function with the specifier position of IP in languages like English (cf. Fig. 2).



$$\mathrm{IP} \rightarrow \underset{(\uparrow \textsc{subj})=\downarrow}{\mathrm{XP}} \quad \mathrm{I'}$$

Figure 2: Structural identification of GFs in configurational languages

Besides X-bar theoretic, endocentric c-structure principles LFG admits exocentric c-structure realisations, to account for the much more flexible word order properties of so-called 'non-configurational' languages – languages that exhibit free word order, discontinuous constituents, or null anaphora. As established by the work of, i.a., [Simpson, 1991] and [Nordlinger, 1998], extensive morphological marking is the most striking characteristics of these 'non-configurational' languages. Here, the identification of grammatical functions is predominantly determined morphologically, by principles that associate morphological marking, such as case or verbal affixes with functional information (cf. Fig. 3).



$$C = \underset{\uparrow=\downarrow}{X^0} \quad \text{or} \quad \underset{(\uparrow (\mathrm{GF}))=\downarrow}{\mathrm{NP}}$$

Dependent-marking: $(\downarrow \textsc{case})=\mathrm{K} \Rightarrow (\uparrow \textsc{gf})=\downarrow$

Head-marking: $(\downarrow \textsc{agr})=(\uparrow \textsc{gf agr}) \Rightarrow (\uparrow \textsc{gf})=\downarrow$ [Nordlinger, 1998]

Figure 3: Exocentric phrase structure and morphological identification of GFs

The complementary, but graded distribution of predominantly morphologically vs. predominantly structurally determined identification of grammatical functions across a wide spectrum of language types is described as “morphology competing with syntax”: languages exhibit different mixtures of morphological and/or structural marking of functional information, yielding a continuous scale along the dimension of (non-)configurationality (cf. [Nordlinger, 1998]).

[Nordlinger, 1998] provides a typologically motivated LFG analysis of non-configurational languages that accounts for head-marking and dependent-marking languages in a uniform way. Morphological marking is viewed as *constructive*, being able to define a syntactic context. The constructional nature of case is formalised by way of inside-out designators that define an embedding functional context. This analysis accounts for the morphology-driven identification of grammatical functions, and the flexible word order properties typically found in these languages.

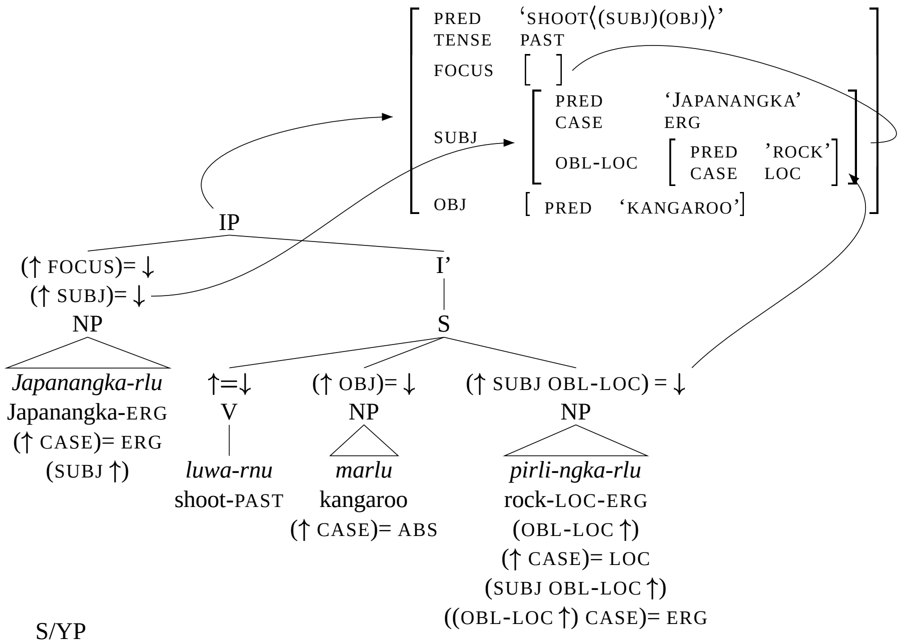
Especially striking are case stacking phenomena in dependent-marking languages, where a constituent encodes its embedding syntactic context by way of multiple case marking. This is illustrated in (1.a), an example from Warlpiri. The case marking on *pirli-ngka-rlu* (rock-LOC-ERG) marks it as a LOCative phrase that is functionally embedded within the (ERGative-marked) subject, which in (1.a) is discontinuously realised.

(1) a.

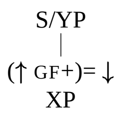
Japanangka-rlu	*luwa-rnu*	*marlu*	*pirli-ngka-rlu*
japanangka-ERG	shoot-PAST	kangaroo	rock-LOC-ERG

’Japanangka shot the kangaroo on the rock’ ([Simpson, 1991])

b.



c.



with GF a variable over grammatical functions

According to Nordlinger’s theory of constructive case, in (1.b) the stacked cases on *pirli-ngka-rlu* introduce the inside-out equations (OBL-LOC ↑) and (SUBJ OBL-LOC ↑).[1] Jointly with the

[1]This is defined by way of the *Principle of Morphological Composition* (cf. [Nordlinger, 1998]).

ERGative case marking on 'Japanangka', which projects the SUBJ function, this enforces the locative phrase to be analysed as a modifier of the discontinuous subject, to be read as: 'Japanangka on the rock'.

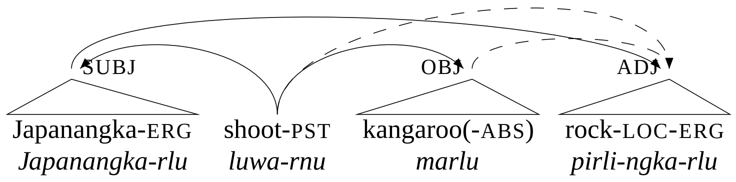
The formal LFG analysis of constructive case necessitates a considerable relaxation of functional descriptions on c-structure categories. The functional descriptions of the NP phrases displayed in (1.b) are mere instantiations of very general (underspecified or disjunctive) functional descriptions as given in (1.c). In languages where grammatical functions are primarily determined by constructive case marking, both the choice of grammatical function, and – with stacking – the depth of functional embedding is determined by the morphological marking on lexical items. The instantiation of underspecified functional path descriptions as in (1.c) is obtained through resolution of the morphologically triggered inside-out functional descriptions (cf. (1.b)).

2.2 Strategies for Ambiguity Resolution

The complementary mechanisms for functional marking in configurational vs. non-configurational languages go along with different strategies for ambiguity resolution. Case marking in general and case stacking on discontinuous phrases in particular provides an excellent means for ambiguity resolution in non-configurational languages – while not necessarily leading to fully disambiguated analyses.[2] Configurational languages, by contrast, can to a certain extent, employ structural means for the resolution of ambiguities. In (2) and (3) we illustrate these distinct strategies for ambiguity resolution by morphological vs. structural encoding.

(2) displays the possible attachments for the case-marked constituents in our Warlpiri example (1).[3] The adjunct's attachment is fully determined by stacked case marking: the ERGative case enforces functional attachment of the LOCative phrase to the SUBJect, disallowing alternative readings with attachment to the OBJect or the verb.

(2)



In a configurational language like English, a reading where *on the rock* is functionally embedded within a discontinuous SUBJect phrase is unavailable for the corresponding sentence (3). By contrast, configurational languages exhibit a systematic structural/functional attachment ambiguity: *on the rock* can be analysed as an independent phrase, and, accordingly, must be analysed

[2] In (i) the stacked cases on *coolamon*, coolamon-LOC-DAT, unambiguously identify the locative phrase as functionally embedded within the dative-marked object – 'the baby in the coolamon'. In (ii), *coolamon* bears default ABSolutive case. In the absolutive reading, the coolamon is to be analysed as embedded within the OBJect – 'the food in the coolamon'. Since absolutive is a default case, it can be interpreted as optional, which leads to the (implausible) analysis of *coolamon* as a verbal adjunct – 'the giving is in the coolamon'.

(i.**)** Karnta-ngku ka-rla kurdu-ku miyi yi-nyi parraja-rla-ku.
woman-ERG PRES-3DAT baby-DAT food(ABS) give-NPST coolamon-LOC-DAT
'The woman is giving food to the baby (who is) in the coolamon.

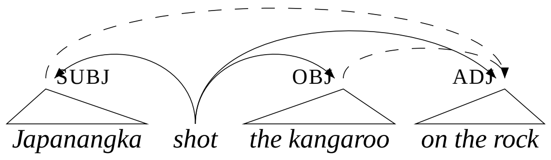
(ii.**)** Karnta-ngku ka-rla kurdu-ku miyi yi-nyi parraja-rla.
woman-ERG PRES-3DAT baby-DAT food(ABS) give-NPST coolamon-LOC(ABS)
'The woman is giving the baby food (which is) in the coolamon. [Nordlinger, 1998]

[3] Solid lines display available dependencies, while dashed lines indicate unavailable readings.

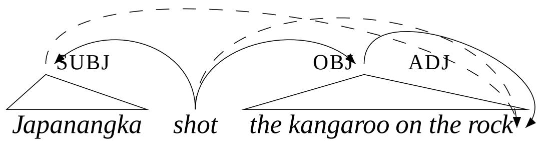
as an adjunct of the verb as in (3.a), or as c-structurally embedded within the phrase *the kangaroo* as in (3.b), in which case it is determined, by principles of structure-function mapping, as functionally embedded within the OBJect.

This kind of structural/functional ambiguity cannot be resolved without further semantic or contextual information, or world knowledge.

(3) a.



b.



In sum, the distinct strategies for identification of grammatical functions in configurational and non-configurational languages – structural vs. morphological identification – lead to distinct configurations and strategies for ambiguity resolution. We will come back to this observation in Section 4, when considering constraints for modifier attachment in a chunk-based analysis of German.

3 From Shallow Parsing to LFG F-Structures

In this section, we briefly review the complementary natures of shallow as opposed to linguistically motivated 'deep' syntactic analysis in computational linguistics. We characterise the problem of integrated shallow and deep syntactic analysis, and present a novel account to integrate these complementary types of analyses, by the design of an LFG projection architecture for shallow syntactic analysis.

We build on an existing cascaded shallow parsing architecture that combines a stochastic topological field parser for German with chunk parsing [Frank et al., 2003a].[4] To the output of this parser we apply a variant of previously developed methods for f-structure projection from context-free grammars and trees in [Frank et al., 2003b], in order to project LFG f-structures from these flat, chunk-based topological trees.

In Section 4 we examine how to bridge the fundamentally distinct natures of a chunk-based c-structure analysis and a corresponding full-fledged f-structure projection with fully specified attachments. This will bring us back to the fundamentally distinct disambiguation strategies of configurational vs. non-configurational languages and their analysis in an LFG framework. In particular, we will show how to adapt Nordlinger's LFG analysis of case-stacking languages to the projection of f-structures from chunk-based analyses of configurational languages.

3.1 The Shallow–Deep Mapping Problem

The two paradigms of shallow vs. deep syntactic analysis in computational linguistics are complementary in various respects:

[4]This type of 'divide and conquer' approach was first proposed by [Peh and Ting, 1996]. Similar parsing architectures that combine topological field parsing with cascaded chunk parsing are described, e.g., in [Wauschkuhn, 1996, Neumann et al., 2000, Hinrichs et al., 2002, Crispi, 2003, Schiehlen, 2003].

Shallow (chunk-based) processing provides *partial analyses* by abstraction from fine-grained linguistic distinctions and contextual constraints. It is therefore *highly robust*, but *less precise and accurate*. Yet, due to the lower complexity of analysis – and thus weaker formalisms – it is *highly efficient*.

Deep syntactic processing delivers *fine-grained* analyses where constraints are resolved within larger, sometimes long-distance syntactic contexts. It is *highly precise*, but inherently *less robust*. Due to the higher complexity of analysis and formalisms employed, deep syntactic processing is *less efficient*.

Integration of shallow and deep processing Recently, attempts have been made to combine shallow and deep syntactic processing, in order to obtain the virtues of both paradigms: *fine-grainedness and precision of deep syntactic analysis* as well as *robustness and efficiency of shallow processing* – while diminishing their respective weaknesses.

Integration of shallow and deep analysis has proven successful for the integration of shallow lexical processing, to complement lexical gaps in a deep grammar ([Grover and Lascarides, 2001, Crysmann et al., 2002, Kaplan and King, 2003]). Integration at the phrasal level can be used to improve processing speed and robustness, by using information from shallow parsing to make the deep parsing process more efficient, or to recover fragments from a failed parse.

Integration at the phrasal level is, however, more complex and problematic ([Daum et al., 2003, Frank et al., 2003a, Kaplan and King, 2003]). First, since in shallow parsing phrasal attachment is not made explicit, shallow and deep analyses cannot be directly mapped to each other. This is illustrated in (4): the flat attachments in (4.b) do not match the explicit embedding structure of (4.a). Second, bottom-up chunk parsing is restricted to a limited syntactic context, and is easily trapped in configurations like (5).

(4) a. [$_{\text{CL}}$ There was [$_{\text{NP}}$ a rumor [$_{\text{CL}}$ it was going to be bought by [$_{\text{NP}}$ a French company [$_{\text{CL}}$ that competes in supercomputers]]]]].

b. [$_{\text{CL}}$ There was [$_{\text{NP}}$ a rumor]] [$_{\text{CL}}$ it was going to be bought by [$_{\text{NP}}$ a French company]] [$_{\text{CL}}$ that competes in supercomputers].

(5) Peter drinks [$_{\text{NP}}$ wine and Mary] eats oranges.

3.2 Cascaded Stochastic Topological Parsing for German

Recently, [Becker and Frank, 2002] developed a non-lexicalised probabilistic parsing approach for German that is based on the theory of topological fields.[5] The *topological field model* of (German) syntax (cf. [Höhle, 1983]) divides basic clauses (CL) into distinct fields – *pre-* (VF), *middle-* (MF), and *post-fields* (NF) – delimited by verbal or sentential markers that occupy the left (LB) and right (RB) sentence bracket positions. This model of clause structure is underspecified, or *partial* as to non-sentential constituent boundaries, but provides a linguistically well-motivated, theory-neutral model of sentence *macro-structure*.

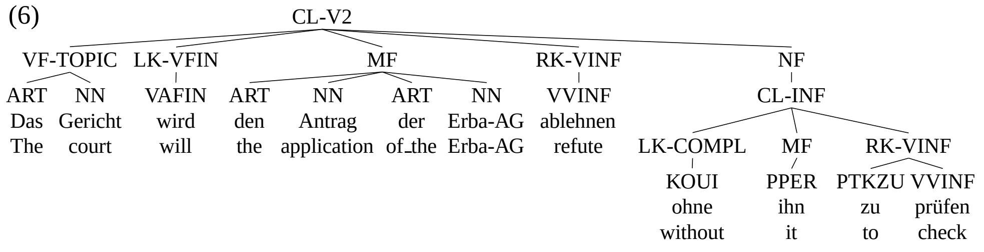
As seen in (6), the topological trees abstract away from non-sentential constituency – phrasal fields MF and VF (pre- and middle-field) expand to flat sequences of PoS tags. By contrast,

[5][Becker and Frank, 2002] explored a corpus-based stochastic approach to topological field parsing, by training a non-lexicalised PCFG on a topologically structured corpus that was derived from the NEGRA treebank. Measured against an evaluation corpus, the parser achieves nearly 100% coverage. Accuracy measures of labelled precision and recall are around 93%. The rate of perfect matches (i.e., full tree identity as compared to the gold standard evaluation corpus) is around 80% (see [Becker and Frank, 2002] for detailed evaluation).

[Veenstra et al., 2002] follow a similar approach, but restrict evaluation to (LB/RB) field *demarcations*, whereas [Becker and Frank, 2002] measure labeled constituency, i.e. the complete embedding structure.

they perfectly render the clausal skeleton and embedding structure of complex sentences. Parameterised node labels encode larger syntactic contexts, or ‘constructions’, such as clause type (CL-V2, -SUBCL, -REL), or inflectional patterns of the verb cluster (RB-VINF,-VFIN, -VPART,..).

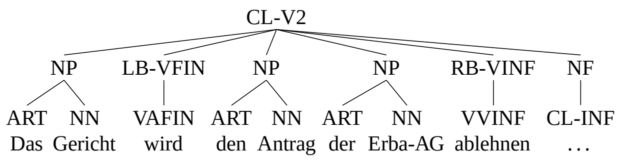
(6)



Due to its linguistic underpinning, the topological field model provides a pre-partitioning of complex sentences that is highly compatible with deep syntactic analysis, and thus maximally effective to increase parsing efficiency if interleaved with deep syntactic analysis. Partiality regarding the constituency of non-sentential material ensures robustness, coverage, and processing efficiency. These properties make topological structures perfect candidates for tight integration with deep syntactic analysis.

By cascaded chunk parsing of flat phrasal fields (VF,MF,NF) – using an off-the shelf chunk parser – we can further refine the topological tree structures to combine explicit sentential embedding with sub-sentential chunk constituents (7).

(7)



In [Frank et al., 2003a] (cascaded) stochastic topological parsing was employed for phrasal integration with a German HPSG grammar, to achieve improvement of parsing efficiency – using hand-coded mappings to bridge between distinct constituency of flat topological structures on the one hand, and the more fine-grained linguistic structures as encoded in an HPSG grammar on the other. In this integration architecture, the pre-partitioning of sentences by way of topological field parsing led to significant efficiency improvements of the HPSG parser, while purely chunk-based information was rather ineffective, or even harmful, due to the mapping problem sketched in (4).

3.3 F-structure Projection from Topological Trees

In this paper, we explore an architecture for integration of shallow and deep analysis, where the aim is to derive *maximally constrained* ’deep’ syntactic representations from shallow analyses, to obtain compatibility between independent shallow and deep parsing processes at the representational level.[6] We thus need to design an architecture for LFG f-structure construction that applies f-structure projection principles to the output of cascaded topological parsing. Despite the underspecified nature of the underlying shallow analysis, the resulting f-structures should be maximally constrained, and compatible with f-structures produced by classical ’deep’ LFG parsing.

Due to the flat chunk-based constituent analysis, special attention needs to be payed to the problem of reconciling chunk analyses with explicit embedding structures as delivered by deep

[6]A similar approach is pursued in related work of [Copestake, 2003], for integration of shallow and deep analysis at the level of semantic representations.

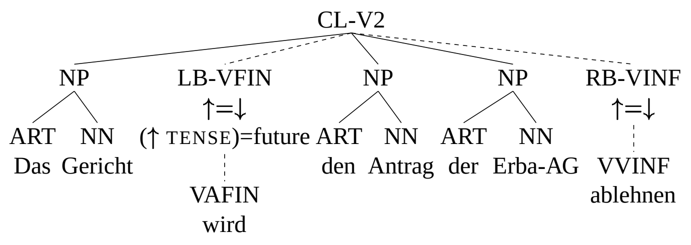
syntactic representations. Thus, the challenge of this approach is to project full-fledged, maximally specified (disjunctively) embedding f-structures from flat, chunk-based constituent structures.

To realise this architecture, we apply previously established methods for automatic annotation of context-free treebank (grammar)s with LFG f-structures. In particular, we can enrich context-free trees or grammars with f-structure projection principles or f-descriptions, to be resolved in a subsequent constraint resolution phase. Different variants of this method have been developed in [Frank, 2000, Sadler et al., 2000, Frank et al., 2003b, Cahill et al., 2002].[7]

Here we employ a variant where the output of the shallow parser, a *context-free tree*, is enriched with *functional descriptions*. These functional descriptions are resolved by deterministically reparsing the 'sentence grammar' that is read off the annotated topological tree.[8]

F-structure annotation operating on trees provides access to non-local syntactic contexts or 'configurations' (i.e., subtrees of depth greater than one), which is especially suited for annotation of flat, chunk-based trees from shallow analysis. This is illustrated in (8). In German, the tree configuration seen in (8) – a finite form of the auxiliary "werden" (VAFIN) in the left sentence bracket position (LB) combined with an infinitival main verb (VVINF) as last verbal element in the right sentence bracket cluster (RB-VINF) – is indicative of future tense. The bits of information that charaterise this 'configuration' are distributed over two levels of embedding. In a tree-based annotation approach we can state a general annotation principle that tests the tree for such a configuration, and associates the corresponding left (LB-VFIN) and right sentence bracket positions (RB-VINF) with f-descriptions ↑=↓ and (↑ TENSE)=future, as displayed in (8).[9]

(8)



[7]Note that projection of LFG f-structures from a grammar encoding topological field structures is not novel either. A hand-coded topological field grammar for German LFG has been presented in [Clement et al., 2002].

[8]This is effectively a combination of tree-based annotation in [Frank, 2000] and the reparsing architecture of [Sadler et al., 2000].

[9]In [Frank, 2000] annotation principles applying to trees are defined by way of a tree description language with basic predicates for tree branches (`arc`), precedence relations (`prec`), and lexical leaf nodes (`lex`). The arguments of `arc` and `lex` record the node identifiers, category labels and lexical form of these nodes. Annotation rules are processed by a term rewriting system, which takes as input the term description for a given tree, and checks it for satisfaction of the left-hand side conditions of an annotation rule. New predicates can be introduced on the right-hand side of a rule, here indicated by the prefix '+'.

The configuration marked in (8) by dashed lines is concisely stated in terms of tree description predicates on the left-hand side of (i). The predicate `f_desc` records the annotation of nodes with f-descriptions, here the functional descriptions for future tense.

```
(i.)     arc(A,'CL','V2',B,'LB',_), arc(B,_,_,C,'VAFIN',_), lex(C,_,'werden'),
         arc(A,'CL','V2',D,'RB','VINF'), arc(D,_,_,E,'VVINF',_)
    ⇒    +f_desc(B,'↑=↓ (↑ TENSE)=future'),
         +f_desc(D,'↑=↓').
```

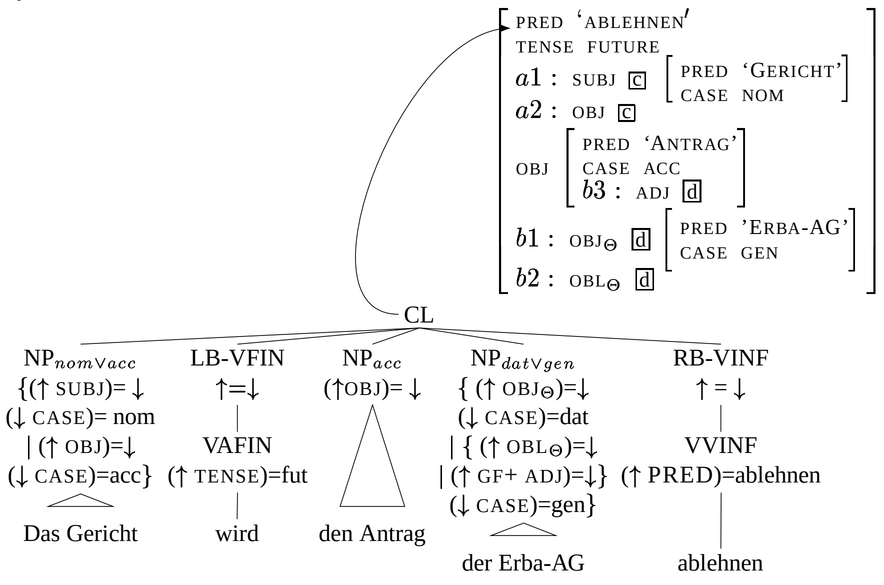
Interaction of morphological and functional constraints F-structure annotation principles can be defined to encode general structure-function mapping principles as displayed in Figs. 1 and 2 (cf. [van Genabith et al., 2001]). Yet, in a language like German, a non-configurational language with moderate case marking and – accordingly – moderately free word order, structural position is not indicative of grammatical function information. Instead, morphological information can provide *partial* functional identification. Thus, we can define annotation principles that (disjunctively) associate morphologically marked NPs with grammatical functions, as illustrated in (9).[10] These annotation principles are clearly reminicent of Nordlinger's general description of morphological identification of grammatical functions in dependent-marking languages in Fig. 3.

(9)

$$\begin{array}{lll} \text{NP}_{nom\vee acc} \Rightarrow & \{ (\uparrow \text{SUBJ}){=}\downarrow & \\ & \quad (\downarrow \text{CASE}){=}\ \text{nom} & \\ & \mid (\uparrow \text{OBJ}){=}\downarrow & \\ & \quad (\downarrow \text{CASE}){=}\ \text{acc}\ \} & \\ \text{NP}_{acc} \Rightarrow & (\uparrow \text{OBJ}){=}\downarrow & \\ & (\downarrow \text{CASE}){=}\ \text{acc} & \end{array}
\qquad
\begin{array}{ll} \text{NP}_{dat\vee gen} \Rightarrow & \{ (\uparrow \text{OBJ}_{\Theta}){=}\downarrow \\ & \quad (\downarrow \text{CASE}){=}\ \text{dat} \\ & \mid \{ (\uparrow \text{OBL}_{\Theta}){=}\downarrow \\ & \quad\ \mid (\uparrow \text{GF+ ADJ}){=}\downarrow \} \\ & \quad (\downarrow \text{CASE}){=}\ \text{gen}\ \} \end{array}$$

Applied to the case-marked NP constituents in (10), the annotation principles in (9) yield a tree decorated with functional annotations. By reparsing the given tree structure, we obtain a highly disjunctive f-structure.[11]

(10)



This disjunctive f-structure can be further resolved by applying general well-formedness conditions for functional structures. Functional bi-uniqueness, e.g., eliminates the disjunctive context $a2$, given that the OBJect function for *Antrag* is in the TRUE context. This yields the partially resolved structure (11.a).[12]

[10]The equations for genitive-marked NPs make use of an uncertainty path description GF+, defining the NP as a possessor adjunct of some accessible function GF+. For more detail see below and Section 4.

[11]We represent disjunctive f-structures as f-structure charts where context variables $a1, a2, ..b1, b2, ..$ identify disjunctive readings (cf. [Maxwell and Kaplan, 1989]). For ease of exposition, we don't represent adjuncts as set-valued features here. We discuss special problems – and solutions – for the analysis of set-valued adjuncts in Section 4.2.

[12]If the annotations do not provide lexical subcategorisation information, as in (10) and (11.a), reparsing must be relaxed to allow violation of the coherence condition.

By use of subcategorisation information from external lexica, we can further restrict the number of readings, by checking for completeness and coherence conditions. In (11.b), with *ablehnen* subcategorising for SUBJ and OBJ, contexts $b1$ and $b2$ are eliminated by violation of coherence.[13]

(11) a. Partial disambiguation by function–argument bi-uniqueness

$$\left[\begin{array}{ll} \text{PRED} & \text{`ABLEHNEN'} \\ \text{TENSE} & \text{FUTURE} \\ \text{SUBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`GERICHT'}\\ \text{CASE} & \text{NOM}\end{array}\right] \\ \text{OBJ} & \left[\begin{array}{ll} \text{PRED} & \text{`ANTRAG'} \\ \text{CASE} & \text{ACC} \\ b3: & \text{ADJ}\ \boxed{d}\end{array}\right] \\ b1: \ \text{OBJ}_{\Theta}\ \boxed{d} & \left[\begin{array}{ll}\text{PRED} & \text{'ERBA-AG'}\\ \text{CASE} & \text{GEN}\end{array}\right] \\ b2: \ \text{OBL}_{\Theta}\ \boxed{d} & \end{array}\right]$$

b. Partial disambiguation by coherence and completeness conditions

$$\left[\begin{array}{ll} \text{PRED} & \text{`ABLEHNEN}\langle(\text{SUBJ})(\text{OBJ})\rangle\text{'} \\ \text{TENSE} & \text{FUTURE} \\ \text{SUBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`GERICHT'}\\ \text{CASE} & \text{NOM}\end{array}\right] \\ \text{OBJ} & \left[\begin{array}{ll} \text{PRED} & \text{`ANTRAG'} \\ \text{CASE} & \text{ACC} \\ b3: \ \text{ADJ}\ \boxed{d} & \left[\begin{array}{ll}\text{PRED} & \text{'ERBA-AG'}\\ \text{CASE} & \text{GEN}\end{array}\right]\end{array}\right] \end{array}\right]$$

Uncertain attachment from flat structures While (11.b) seems fully disambiguated, there is in fact a final source of ambiguity that we disregarded up to this point: the annotation of a genitive NP as an embedded possessive modifier, by the functional uncertainty equation ($\uparrow$ GF+ ADJ)= $\downarrow$:

$$\begin{array}{lll} \text{NP}_{gen} & \Rightarrow & \{\ (\uparrow \text{OBL}_{\Theta})=\downarrow \\ & & \ \mid (\uparrow \text{GF+ ADJ})=\downarrow\ \} \\ & & (\downarrow \text{CASE})= \text{gen} \end{array}$$

This functional uncertainty equation accommodates for the embedding of a genitive marked NP chunk as a possessive adjunct of a discontinuously realised NP or PP chunk, here the NP *Antrag* that was identified as OBJ. Yet, in its current form the equation allows the modifier to be embedded within *any* of the locally accessible grammatical functions. Thus, by instantiating GF+ to SUBJ, we finally obtain the f-structure (12), with the additional disjunct $b4$. Attachment to the discontinuously realised SUBJect (*Gericht*) is, however, not a valid reading of the sentence.

(12)
$$\left[\begin{array}{ll} \text{PRED} & \text{`ABLEHNEN}\langle(\text{SUBJ})(\text{OBJ})\rangle\text{'} \\ \text{TENSE} & \text{FUTURE} \\ \text{SUBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`GERICHT'}\\ \text{CASE} & \text{NOM} \\ b4: & \text{ADJ}\ \boxed{d}\end{array}\right] \\ \text{OBJ} & \left[\begin{array}{ll} \text{PRED} & \text{`ANTRAG'} \\ \text{CASE} & \text{ACC} \\ b3: \ \text{ADJ}\ \boxed{d} & \left[\begin{array}{ll}\text{PRED} & \text{'ERBA-AG'}\\ \text{CASE} & \text{GEN}\end{array}\right]\end{array}\right] \end{array}\right]$$

[13] Alternative accounts that assign function-argument structure on the basis of cascaded shallow parsing, such as [Wauschkuhn, 1996, Hinrichs and Trushkina, 2002, Crispi, 2003, Schiehlen, 2003] exploit similar strategies of morphologically guided function assignment. In contrast to these approaches our analysis is based on independently motivated functional syntactic principles, and supported by algorithms for resolution of functional constraints. Functional constraints can be specified in a declarative formalism that allows to express non-local dependencies.

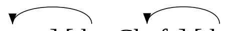
4 Projecting LFG F-Structures from Chunks

The novel aspect of our cascaded shallow-to-deep parsing architecture is the annotation of chunk-based constituent structures to project LFG f-structures that exhibit explicit (while possibly disjunctive) embedding relations between phrases that are not as such represented in the flat c-structure backbone. As seen in the previous section, this can be obtained by annotating potentially embedded phrases with uncertainty path descriptions – similar to what we find in non-configurational languages that license discontinuous constituents (cf. Section 2, example (1.c)).

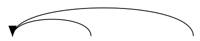
However, unlike case-marking languages where embedding relations between discontinuously realised phrases are indicated by way of (stacked) case marking, chunk analyses for configurational languages are artificial constructs, lacking extensive morphological marking to identify potential attachment relations. However, as discussed in Section 2.2, example (3), configurational languages exhibit structural adjacency constraints on adjunct embedding. Functional uncertainty equations that accommodate for potential embedding of adjunct chunks must therefore be constrained to obey adjacency conditions that rule out ungrammatical readings, such as the reading $b4$ in (12), with attachment of the modifier to a discontinuously realised SUBJ in the sentence vorfeld position.

4.1 Functional Embedding from Flat C-Structures

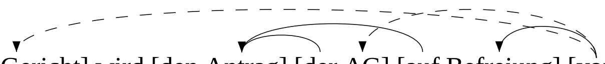
Strict and parallel embedding – adjacency constraints NP or PP chunks that are not selected by a subcategorising head, i.e. free-floating modifier chunks, can be functionally attached to a preceding chunk in one of two ways: by strict or parallel embedding, as illustrated in (13.a) and (13.b), respectively. The structural restrictions for functional attachment to a preceding (or following) chunk constituent are illustrated in (13.c) – with dashed lines indicating illicit readings. As can be observed from the corresponding deep syntactic bracketings in the glosses, functional attachment of a modifier chunk to some other chunk constituent is restricted to configurations where – in the corresponding deep syntactic representation – the attached constituent and its functional antecedent phrase are contained within a minimal contiguous phrase. That is, in the corresponding deep syntactic representation the functionally embedded constituent must be c-structurally embedded within the phrase to which it is functionally attached.



(13) a. Das Gericht wird [den Antrag] [des Chefs] [der Erba-AG] ablehnen
The court will [the application [of the head [of the Erba-AG]]] refute



b. Das Gericht wird [den Antrag] [der AG] [auf Steuerbefreiung] ablehnen
The court will [the application [of the AG] [for tax exemption]] refute



c. [Das Gericht] wird [den Antrag] [der AG] [auf Befreiung] [von Steuern] ablehnen
[The court] will [the application [of the AG] [for exemption [from tax]]] refute

We will model this contiguity restriction of the corresponding 'deep' syntactic constituent structure by defining the functional attachment of a modifier chunk as 'anaphoric' to the functional embedding path of its directly preceding left (resp. following right) sister node.

In analogy to the $\uparrow$ and $\downarrow$ metavariables, the left/right-pointing arrow in a functional description refers to the f-structure of the left/right-adjacent sister node of the current node.[14] Similar to standard inside-out functional descriptions, where (GF$*\uparrow$) identifies an uncertain embedding path of grammatical functions, starting from the f-structure of the mother of the current node, we can make use of the left/right-pointing arrow for inside-out descriptions starting from the f-structure of the left/right-adjacent sister node of the current node.

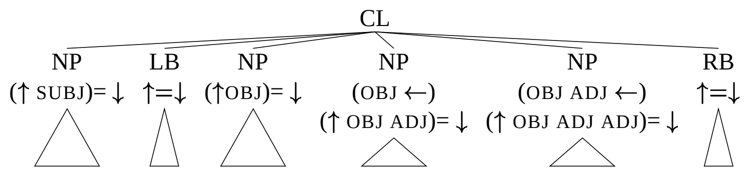
$\leftarrow$	f-structure of the left-adjacent sister node of the current node
(GF$*\leftarrow$)	inside-out functional path starting from left sister of current node
(PATH $\leftarrow$)	id., with PATH = GF$*$

Version I With this formal device, we can annotate potentially embedded NP/PP chunks as stated in (14). The annotation refers to the f-structure ($\leftarrow$) and functional embedding path of the left-adjacent constituent by the inside-out designator (PATH $\leftarrow$), with PATH a variable for the chosen instantiation of the uncertain embedding path GF$*$. The adjunct is then defined as embedded relative to this embedding path, by ($\uparrow$ PATH ADJ)= $\downarrow$.

(14) NP/PP
($\underline{\text{PATH}}$ $\leftarrow$)
($\uparrow$ $\underline{\text{PATH}}$ ADJ)= $\downarrow$

This analysis naturally precludes functional emeddings that violate the c-structural contiguity condition in a corresponding deep syntactic analysis: As each potentially embedded chunk is forced to pick up the functional embedding path of its directly adjacent sister node, functional embedding is required to proceed in a cascade, effectively preventing crossing dependencies.

However, the annotation in (14) only allows for strict embeddings of sequences of chunks, as illustrated in (15). Parallel embedding relations as in (13.b) are precluded: given the embedding of the first NP adjunct under the OBJ function (as in (15)), the second adjunct NP can only be embedded relative to the left sister's embedding path OBJ ADJ. For parallel embedding (= high attachment) of the second adjunct NP, however, the left-sister's embedding path would have to be OBJ. Thus, parallel embedding is not captured by the annotation in (14).

(15)


Version II We slightly modify the previous version, to accommodate for variable strict or parallel embedding of adjunct chunks. This is obtained by splitting the functional embedding path of the adjacent sister node into variable, possibly empty subpaths: a shared embedding path, and a variable path suffix that may be omitted, or skipped for the embedding of the adjunct chunk in question, to allow for parallel embedding relative to a common prefix embedding path.

That is, in (16) we identify the functional embedding path of the adjacent sister node by the inside-out designator (PATH GF$*\leftarrow$), splitting it into variable prefix and suffix subpaths. The adjunct's embedding is then defined with reference to the prefix path, by ($\uparrow$ PATH ADJ)= $\downarrow$, which is thus shared between the adjacent sister and the current adjunct chunk.

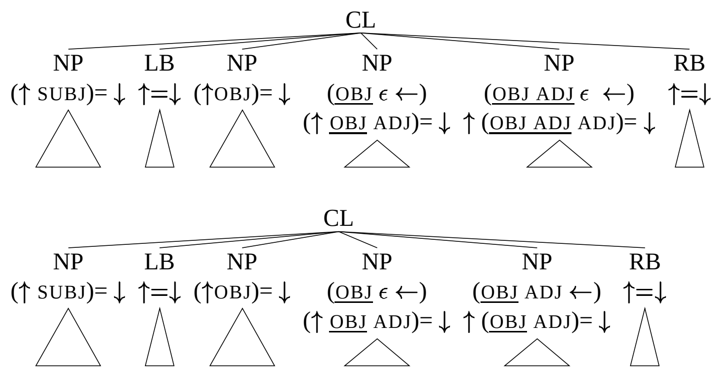
[14]The left/right-pointing arrow was used, e.g., in [Nordlinger, 1998] for an alternative definition of the *Principle of Morphological Composition*.

(16) NP/PP
$(\underline{\text{PATH}}\ \text{GF}{*}\ \leftarrow)$
$(\uparrow \underline{\text{PATH}}\ \text{ADJ}) = \downarrow$

This allows for variable strict and parallel embedding for sequences of chunks, depending on the choice for the suffix $\text{GF}{*}$: We derive strict embedding by setting $\text{GF}{*}$ to the empty string. Parallel embedding (of variable depth) is obtained by choosing the suffix $\text{GF}{*}$ to be nonempty.

Based on this analysis, annotation of sequences of chunks as in (17) yields alternative readings for strict (17.a) vs. parallel (17.b) embedding.

(17) a. b.

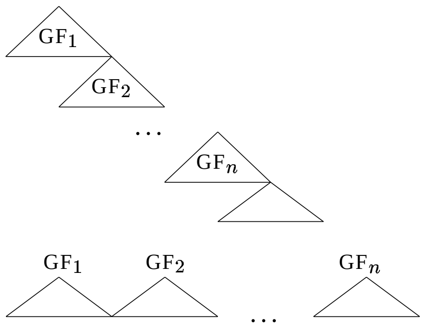


The analysis is illustrated in a more abstract way in Figs. 4 and 5. Here we contrast the structure-function associations for traditional (hierarchical) c-structures with those for flat c-structures of non-embedded sequences of chunks.

Strict functional embedding from flat sequences of chunks as in Fig. 4 can be modelled rather straightforwardly, by transposing the hierarchical analysis of functional embedding to a sequence-based approach. Thus, a given chunk ch_n in a sequence of chunks $ch_1 ... ch_n$ can be strictly embedded relative to the function GF_{n-1} projected by its preceding constituent ch_{n-1}, by referring to the functional embedding of this adjacent constituent, as in (16) with $\text{GF}{*}=\epsilon$. We obtain a strict embedding relation that is in accordance with the corresponding deep syntactic contiguity condition.

Parallel functional embedding from flat sequences of chunks as illustrated in Fig. 5 is less straightforward. In a hierarchical c-structure, a constituent c_{k+1} that is high attached to some constituent c_2 is in general directly c-structurally embedded within this latter constituent. In a sequentialised, flat sequence of constituents, we cannot directly access the corresponding chunk ch_2, but somehow need to 'skip' the intervening (strictly embedded resp. preceding) series of chunks $ch_3 ... ch_k$, to be able to state direct functional embedding of GF_{k+1} relative to GF_2.

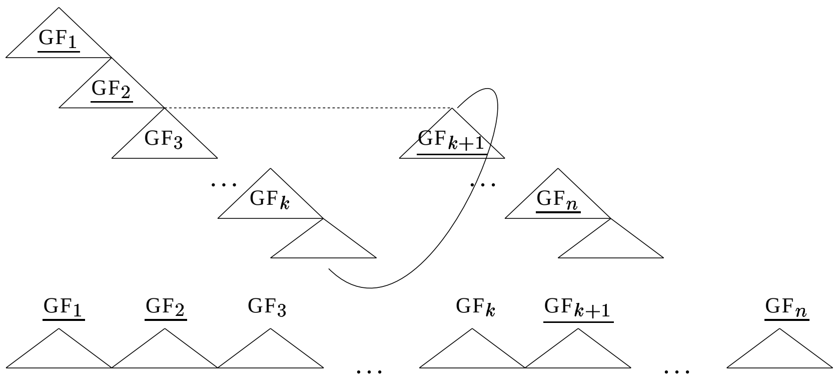
Both configurations are captured by the annotation in (16). By using the full functional embedding path of the left-adjacent constituent we obtain strict embedding of a given adjunct chunk; by 'skipping' a variable-length suffix of its adjacent constituent, we access a higher functional embedding level for parallel attachment of the given adjunct's f-structure. If PATH is instantiated to the empty string, we obtain high attachment of the modifier at the level of its local clause nucleus.



$(\underline{\mathrm{GF}_1\ \mathrm{GF}_2 \ldots \mathrm{GF}_{n-1}}\ \epsilon \leftarrow)$
$(\uparrow \underline{\mathrm{GF}_1\ \mathrm{GF}_2 \ldots \mathrm{GF}_{n-1}}\ \mathrm{GF}_n) = \downarrow$

$$\left[\mathrm{GF}_1 \left[\mathrm{GF}_2 \ \ldots \left[\mathrm{GF}_n \left[\ \right] \right] \right] \right]$$

Figure 4: Strict embedding from flat structures



$(\underline{\mathrm{GF}_1\ \mathrm{GF}_2}\ \mathrm{GF}_3 \ldots \mathrm{GF}_k \leftarrow)$
$(\uparrow \underline{\mathrm{GF}_1\ \mathrm{GF}_2}\ \mathrm{GF}_{k+1}) = \downarrow$

$$\left[\underline{\mathrm{GF}_1} \left[\underline{\mathrm{GF}_2} \left[\begin{array}{ll} \mathrm{GF}_3 \ \ldots & \left[\mathrm{GF}_k \left[\ \right] \right] \\ \underline{\mathrm{GF}_{k+1}} \ \ldots & \left[\underline{\mathrm{GF}_n} \left[\ \right] \right] \end{array} \right] \right] \right]$$

Figure 5: Parallel embedding from flat structures

Contiguity In this analysis it is (i) the access to the f-structure of the left/right-adjacent sister node via the left/right-pointing arrow, and (ii) the shared (prefix) functional embedding path for strict and parallel embedding that jointly prevent functional attachment of a modifier to a constituent that – in a corresponding deep syntactic representation – would be non-contiguous.

For each modifier, functional embedding is required to be stated relative to the functional embedding path of its directly adjacent sister node. This prevents direct access to a grammatical function that is *not a prefix* of the sister's embedding path, such as the SUBJ in (17), or the OBJ in case the first modifier is attached to the verb. Thus, functional embedding relations that violate the contiguity condition are ruled out by the fact that the functional embedding of a node is strictly dependent on the functional embedding of its left- or right-adjacent node: it is possible to skip the lower parts of the sister's functional embedding, to yield high attachment, but it is impossible to select a distinct embedding path which is not contained in the path of the adjacent sister node.

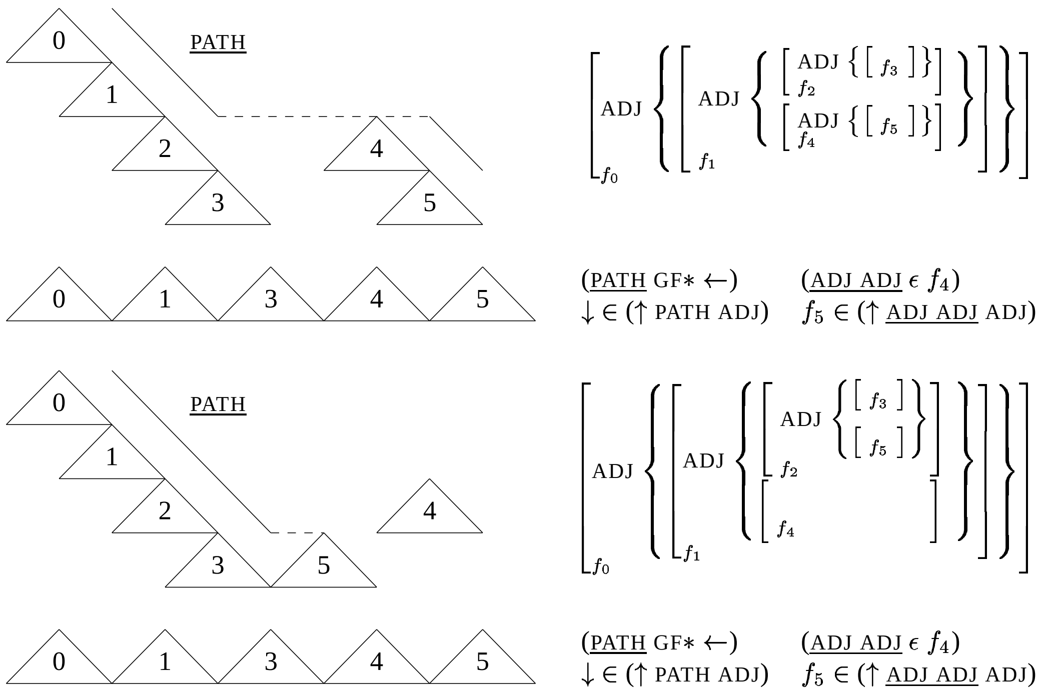

PATH
0 1 2 3 4 5
ADJ f0 ADJ f1 ADJ f2 f3 ADJ f4 f5
(PATH GF∗ ←) ↓ ∈ (↑ PATH ADJ)
(ADJ ADJ ϵ f4) f5 ∈ (↑ ADJ ADJ ADJ)

Figure 6: Adjunct sets: indeterminate reference to set elements

4.2 The Nitty-Gritty: Adjunct Sets

Up to this point our analysis ignored the complicating details of a set-based analysis of ADJuncts. While we can easily modify the annotations in (16) to define adjuncts as set-valued functions (i.e., by $\downarrow \in (\uparrow$ PATH GF$*$)), there is in fact a deeper problem lurking in the analysis of Version II, which is due to the inherent non-determinism of outside-in reference to elements of a set.

The problem is illustrated in Fig. 6, where we focus on the attachment of chunk ch_5 with the associated f-structure f_5. In the upper configuration, we define f_5 to be embedded as an ADJunct of the f-structure f_4 (of chunk ch_4), by picking up the embedding path of its left sister ch_4 (i.e., starting from f_4), and instantiating PATH to ADJ ADJ. The resulting f-structure corresponds to the attachment configuration displayed in the corresponding hierarchical structure.

Now, since f_1 and f_2 are set-valued, the description $\downarrow \in (\uparrow$ ADJ ADJ ADJ) on chunk ch_4 of the flat c-structure analysis alternatively defines the f-structure displayed in the lower part of Fig. 6. Here, f_5 is attached to f_2. We end up with an f-structure that corresponds to a hierarchical structure where the constituent is attached to the wrong antecedent – violating the contiguity condition.

Version III This unwarranted indeterminacy can be avoided if the embedding is strictly defined by inside-out functional equations. In fact, we can reformulate (16) by avoiding the outside-in equation that leads to indeterminate reference. Splitting the inside-out embedding path of the left-/right-adjacent sister into prefix PATH and suffix SKIP-PATH, it is effectively only the suffix SKIP-PATH (= GF*) that is needed to define parallel or strict embedding of chunks: setting SKIP-PATH to the empty string yields strict embedding; a nonempty SKIP-PATH defines the depth of functional embedding that is 'skipped' to define parallel attachment to a 'higher' constituent. Thus, uncertain modifier attachment from a flat sequence of chunks now reads as in (18). Applied to the example of Fig. 6 the description $f_5 \in ((\textsc{gf}^*\ f_4)\ \textsc{adj})$, with $\textsc{gf}^*{=}\epsilon$ yields the (single) intended embedding of f_5 as an ADJunct of f_4.

(18) $\dfrac{\text{NP/PP}}{\downarrow \in ((\textsc{skip-path} \leftarrow)\ \textsc{adj})}$ with $\textsc{skip-path} = \textsc{gf}*$.

5 Conclusion

We presented an LFG architecture that bridges the gap between flat, chunk-based shallow parsing and deep syntactic analysis, by defining LFG f-structure projection from chunk-based topological trees. We argued that f-structure projection from chunk-based structures is conceptually related to the LFG analysis of non-configurational languages. While related, chunk analyses for configurational languages are artificial constructs that lack extensive morphological marking. Instead, we showed how structural adjacency constraints for functional embedding that are most typical for configurational languages can be modeled by inside-out functional descriptions – similar to morphologically guided attachment of discontinuous constituents in non-configurational languages.

In contrast to previous approaches to shallow dependency parsing that apply collections of syntactic 'heuristics', our projection architecture builds on a well-established linguistic formalism and well-defined syntactic principles. In particular, we could show that linguistic insights from typological syntactic research can be applied to model and formalise the kind of underspecification that is characteristic of shallow parsing approaches employed in computational linguistics.

An integration model that provides compatible representations for shallow and deep analysis allows for flexible combination and cross-validation of concurrent systems [Copestake, 2003]. Moreover, due to compatible representations, disambiguation models developed for 'deep' LFG grammars can be applied to resolve remaining ambiguities from chunk-based processing.

Acknowledgements I would like to thank the audiences of the LFG conference, of the Universities of Stuttgart and Konstanz, as well as the members of the Parc NLTT and QCA research groups for discussion and comments. Special thanks go to John Maxwell for implementation of the left/right-pointing arrow within the XLE system. I am particularly grateful to the Deutsche Fördergemeinschaft (DFG), who supported the presentation of this work at the LFG Conference.

References

[1] Abney, S. (1996). Partial Parsing via Finite-state Cascades. In *ESSLLI Workshop on Robust Parsing*, Prague.

[2] Becker, M. and Frank, A. (2002). A Stochastic Topological Parser of German. In *Proceedings of COLING 2002*, pages 71–77, Taipei, Taiwan.

[3] Bresnan, J. (2001). *Lexical-Functional Syntax*. Blackwell Publishers, Oxford.

[4] Butt, M., Dyvik, H., King, T. H., Masuichi, H., and Rohrer, C. (2002). The Parallel Grammar Project. In *Proceedings of the Workshop on Grammar Engineering and Evaluation, COLING 2002*, Taipei, Taiwan.

[5] Cahill, A., Forst, M., McCarthy, M., O'Donovan, R., Rohrer, C., van Genabith, J., and Way, A. (2003). Treebank-Based Multilingual Unification-Grammar Development. In *ESSLLI'03 Workshop on Ideas and Strategies in Multilingual Grammar Development*.

[6] Cahill, A., McCarthy, M., van Genabith, J., and Way, A. (2002). Automatic Annotation of the Penn-Treebank with LFG F-Structure Annotation. In Lenci, A., Montemagni, S., and Pirelli, V., editors, *Proceedings of the LREC 2002 Workshop on Linguistic Knowledge Acquisition and Representation - Bootstrapping Annotated Language Data*, Gran Canaria, Spain.

[7] Clement, L., Gerdes, K., and Kahane, S. (2002). An LFG-type Grammar for German based on the Tolological Model. In Butt, M. and King, T., editors, *Proceedings of the LFG 2002 Conference*, pages 116–129, Athens, Greece. CSLI Publications, Stanford, CA.

[8] Copestake, A. (2003). Report on the Design of RMRS. Technical Report D1.1a, University of Cambridge, University of Cambridge, UK. 20 pages.

[9] Crispi, C. (2003). Linguistische Annotierung mit flachen Grammatiken. Master's thesis, Universität des Saarlandes.

[10] Crysmann, B., Frank, A., Kiefer, B., Müller, S., Neumann, G., Piskorski, J., Schäfer, U., Siegel, M., Uszkoreit, H., Xu, F., Becker, M., and Krieger, H.-U. (2002). An Integrated Architecture for Deep and Shallow Processing. In *Proceedings of ACL 2002*.

[11] Dalrymple, M. (2001). *Lexical-Functional Grammar*, volume 34 of *Syntax and Semantics*. Academic Press.

[12] Daum, M., Foth, K., and Menzel, W. (2003). Constraint-based Integration of Deep and Shallow Parsing Techniques. In *Proceedings of EACL 2003*, Budapest, Hungary.

[13] Frank, A. (2000). Automatic F-structure Annotation of Treebank Trees. In Butt, M. and King, T., editors, *Proceedings of the LFG00 Conference*, CSLI Online Publications, Stanford, CA.

[14] Frank, A., Becker, M., Crysmann, B., Kiefer, B., and Schäfer, U. (2003a). Integrated Shallow and Deep Parsing: ToPP meets HPSG. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics, ACL 2003*, pages 104–111, Sopporo, Japan.

[15] Frank, A., Sadler, L., van Genabith, J., and Way, A. (2003b). From Treebank Resources to LFG F-Structures. Automatic F-Structure Annotation of Treebank Trees and CFGs Extracted from Treebanks. In Abeille, A., editor, *Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers, The Netherlands.

[16] Grover, C. and Lascarides, A. (2001). XML-based Data Preparation for Robust Deep Parsing. In *Proceedings of ACL/EACL 2001*, pages 252–259, Toulouse, France.

[17] Hinrichs, E., Kübler, S., Müller, F., and Ule, T. (2002). A Hybrid Architecture for Robust Parsing of German. In *Proceedings of the LREC 2002 Conference*, Las Palmas, Spain.

[18] Hinrichs, E. and Trushkina, J. (2002). Getting a Grip on Morphological Disambiguation. In Busemann, S., editor, *Proceedings of KONVENS 2000, 6. Konferenz zur Verarbeitung natürlicher Sprache*, Saarbrčken, Germany.

[19] Höhle, T. (1983). Topologische Felder. Unpublished manuscript, University of Cologne.

[20] Kaplan, R. and King, T. (2003). Low-level Markup and Large-scale LFG Grammar Processing. In Butt, M. and King, T., editors, *Proceedints of the LFG 2003 Conference*, Saratoga Springs, New York.

[21] Maxwell, J. T. I. and Kaplan, R. M. (1989). An overview of disjunctive constraint satisfaction. In *Proceedings of IWPT*, pages 18–27.

[22] Neumann, G., Braun, C., and Piskorski, J. (2000). A Divide-and-Conquer Strategy for Shallow Parsing of German Free Texts. In *Proceedings of ANLP*, pages 239–246, Seattle, Washington.

[23] Nordlinger, R. (1998). *Constructive Case. Evidence from Australian Languages*. CSLI Publications, Stanford, California.

[24] Peh, L. and Ting, C. (1996). A Divide-and-Conquer Strategy for Parsing. In *Proceedings of the International Workshop on Parsing Technology (IWPT)*.

[25] Riezler, S., King, T. H., Kaplan, R. M., Crouch, R., Maxwell, J. T. I., and Johnson, M. (2002). Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the ACL'02*, Philadelphia, PA.

[26] Sadler, L., van Genabith, J., and Way, A. (2000). Automatic F-Structure Annotation from the AP Treebank. In Butt, M. and King, T., editors, *Proceedings of the LFG00 Conference*, University of California, Berkley, CSLI Online Publications, Stanford, CA.

[27] Schiehlen, M. (2003). A Cascaded Finite-State Parser. In *Proceedings of EACL*, Budapest, Hungary.

[28] Simpson, J. (1991). *Warlpiri Morpho-Syntax*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

[29] van Genabith, J., Frank, A., and Way, A. (2001). Treebank vs. X-bar based Automatic F-Structure Annotation. In Butt, M. and King, T., editors, *Proceedings of the LFG 2001 Conference*, Hong Kong, China. CSLI Publications, Stanford,CA.

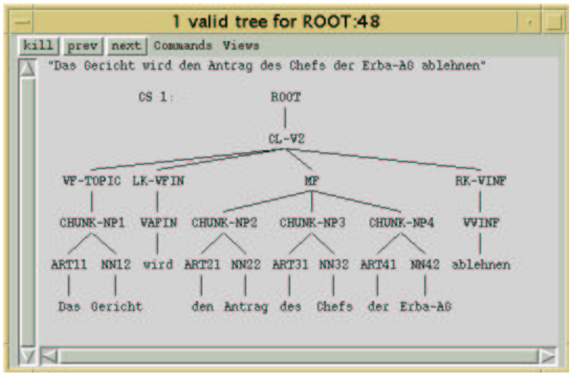
[30] Veenstra, J., Müller, F., and Ule, T. (2002). Topological Fields Chunking for German. In *Proceedings of CoNLL 2002*, pages 56–62, Taipei, Taiwan.

[31] Wauschkuhn, O. (1996). Ein Werkzeugzur partiellen syntaktischen Analyse deutscher Textcorpora. In Gibbon, D., editor, *Natural Language Processing and Speech Technology. Results of the 3rd KONVENS Conference*, pages 356–368. Mouton de Gruyter, Berlin.

Appendix

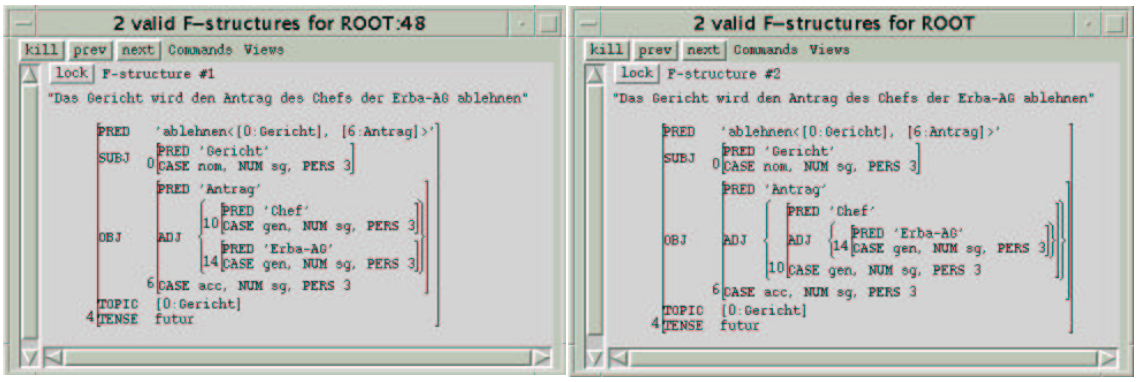
An example from a toy implementation in the XLE grammar development and processing system:

Das Gericht wird den Antrag des Chefs der Erba-AG ablehnen
The court will the application of the head of the Erba-AG refute



"Das Gericht wird den Antrag des Chefs der Erba-AG ablehnen"

Figure 7: C-structure from cascaded topological and chunk parsing


2 valid F–structures for ROOT:48
kill
prev
next
Commands
Views
lock
2 valid F–structures for ROOT
kill
prev
next
Commands
Views
lock

```
F-structure #1
"Das Gericht wird den Antrag des Chefs der Erba-AG ablehnen"

  PRED    'ablehnen<[0:Gericht], [6:Antrag]>'
  SUBJ   0[PRED 'Gericht'
          [CASE nom, NUM sg, PERS 3]
          [PRED 'Antrag'
          [        [PRED 'Chef'
          [      10[CASE gen, NUM sg, PERS 3]
  OBJ     [ADJ  {
          [        [PRED 'Erba-AG'
          [      14[CASE gen, NUM sg, PERS 3]
         6[CASE acc, NUM sg, PERS 3
  TOPIC   [0:Gericht]
 4TENSE   futur
```

```
F-structure #2
"Das Gericht wird den Antrag des Chefs der Erba-AG ablehnen"

  PRED    'ablehnen<[0:Gericht], [6:Antrag]>'
  SUBJ   0[PRED 'Gericht'
          [CASE nom, NUM sg, PERS 3]
          [PRED 'Antrag'
          [        [PRED 'Chef'
  OBJ     [ADJ  {  [ADJ  {  [PRED 'Erba-AG'
          [        [      14[CASE gen, NUM sg, PERS 3]
          [      10[CASE gen, NUM sg, PERS 3
         6[CASE acc, NUM sg, PERS 3
  TOPIC   [0:Gericht]
 4TENSE   futur
```

Figure 8: F-structures 1 and 2 for parallel and strict embedding