Separating Surface Order and Syntactic Relations in a Dependency Grammar

Norbert Bröker
Universität Stuttgart
Azenbergstr. 12
D-70174 Stuttgart
NOBI@IMS.UNI-STUTTGART.DE

Abstract

This paper proposes decoupling the dependency tree from word order, such that surface ordering is not determined by traversing the dependency tree. We develop the notion of a *word order domain structure*, which is linked but structurally dissimilar to the syntactic dependency tree. The proposal results in a lexicalized, declarative, and formally precise description of word order; features which lack previous proposals for dependency grammars. Contrary to other lexicalized approaches to word order, our proposal does not require lexical ambiguities for ordering alternatives.

1 Introduction

Recently, the concept of valency has gained considerable attention. Not only do all linguistic theories refer to some reformulation of the traditional notion of valency (in the form of θ-grid, subcategorization list, argument list, or extended domain of locality); there is a growing number of parsers based on binary relations between words (Eisner, 1997; Maruyama, 1990).

Given this interest in the valency concept, and the fact that word order is one of the main difference between phrase-structure based approaches (henceforth PSG) and dependency grammar (DG), it is valid to ask whether DG can capture word order phenomena without recourse to phrasal nodes, traces, slashed categories, etc. A very early result on the weak generative equivalence of context-free grammars and DGs suggested that DGs are incapable of describing surface word order (Gaifman, 1965). This result has recently been critizised to apply only to impoverished DGs which do not properly represent formally the expressivity of contemporary DG variants (Neuhaus & Bröker, 1997).

Our position will be that dependency relations are motivated semantically (Tesnière, 1959), and need not be projective (i.e., may cross if projected onto the surface ordering). We argue for so-called *word order domains*, consisting of partially ordered sets of words and associated with nodes in the dependency tree. These order domains constitute a tree defined by set inclusion, and surface word order is determined by traversing this tree. A syntactic analysis therefor consists of two linked, but dissimilar trees.

Sec. 2 will briefly review approaches to word order in DG. In Sec. 3, word order domains will be defined, and Sec. 4 introduces a modal logic to describe dependency structures. Sec. 5 applies our approach to the German clause and Sec. 6 relates it to some PSG approaches.

2 Word Order in DG

A very brief characterization of DG is that it recognizes only lexical, not phrasal nodes, which are linked by directed, typed, binary relations to form a dependency tree (Tesnière, 1959; Hudson, 1993). The following overview of DG flavors shows that various mechanisms (global rules, general graphs, procedural means) are generally employed to lift the limitation of projectivity and discusses some shortcomings of these proposals.

Functional Generative Description (Sgall et al., 1986) assumes a language-independent *underlying order*, which is represented as a projective dependency tree. This abstract representation of the sentence is mapped via *ordering rules* to the concrete surface realization. Recently, Kruijff (1997) has given a categorial-style formulation of these ordering rules. He assumes associative categorial operators, permuting the arguments to yield the surface ordering. One difference to our proposal is that we argue for a representational account of word order (based on valid structures representing word order), eschewing the non-determinism introduced by unary operators; the second difference is the avoidance of an underlying structure, which stratifies the theory and makes incremental processing difficult.

Meaning-Text Theory (Melc'ŭk, 1988) assumes seven *strata of representation*. The rules mapping from the unordered dependency trees of surface-syntactic representations onto the annotated lexeme sequences of deep-morphological representations include *global ordering rules* which allow discontinuities. These rules have not yet been formally specified (Melc'ŭk & Pertsov, 1987p.187f).

Word Grammar (WG, Hudson (1990)) is based on *general graphs* instead of trees. The ordering of two linked words is specified together with their dependency relation, as in the proposition "`object of verb follows it`". Extraction of, e.g., objects is analyzed by establishing an additional dependency called `visitor` between the verb and the extractee, which requires the reverse order, as in "`visitor of verb precedes it`". This results in inconsistencies, since an extracted object must follow the verb (being its `object`) and at the same time precede it (being its `visitor`). The approach compromises the semantic motivation of dependencies by adding *purely order-induced dependencies*. WG is similar to our proposal in that it also distinguishes a propositional meta language describing the graph-based analysis structures.

Dependency Unification Grammar (DUG, Hellwig (1986)) defines a tree-like data structure for the representation of syntactic analyses. Using morphosyntactic features with special interpretations, a word defines *abstract positions* into which modifiers are mapped. Partial orderings and even discontinuities can thus be described by allowing a modifier to occupy a position defined by some transitive head. The approach requires that the *parser* interpretes several features specially, and it cannot restrict the scope of discontinuities.

Slot Grammar (McCord, 1990) employs a number of rule types, some of which are exclusively concerned with precedence. So-called head/slot and slot/slot *ordering rules* describe the precedence in projective trees, referring to arbitrary predicates over head and modifiers. Extractions (i.e., discontinuities) are merely handled by a mechanism built into the *parser*.

3 Word Order Domains

Summarizing the previous discussion, we require the following of a word order description for DG:

- not to compromise the semantic motivation of dependencies,
- to be able to restrict discontinuities to certain constructions and delimit their scope,
- to be lexicalized without requiring lexical ambiguities for the representation of ordering alternatives,
- to be declarative (i.e., independent of an analysis procedure), and
- to be formally precise and consistent.

The subsequent definition of an order domain structure and its linking to the dependency tree satisify these requirements.

3.1 The Order Domain Structure

A *word order domain* is a set of words, generalizing the notion of positions in DUG. The cardinality of an order domain may be restricted to at most one element, at least one element, or – by conjunction – to exactly one element. Each word is associated with a sequence of order domains, one of which must contain the word itself, and each of these domains may require that its elements have certain features. Order domains can be partially ordered based on set inclusion: If an order domain d contains word w (which is not associated with d), every word w' contained in a domain d' associated with w is also contained in d; therefor, $d' \subset d$ for each d' associated with w. This partial ordering induces a tree on order domains, which we call the *order domain structure*.

Take the example of German "*Den Mann hat der Junge gesehen*" ("*the man$_{ACC}$ – has – the boy$_{NOM}$ – seen*"). Its dependency tree is shown in Fig.1, with word order domains indicated by dashed circles. The finite verb, "*hat*", defines a sequence of domains, $\langle d_1, d_2, d_3 \rangle$, which roughly correspond to the topological fields in the German main clause. The nouns "*Mann*"

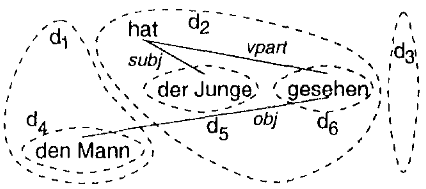

Figure 1: Dependency Tree and Order Domains for "*Den Mann hat der Junge gesehen*"

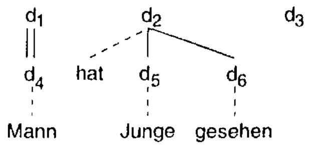

Figure 2: Order Domain Structure for "*Den Mann hat der Junge gesehen*"

and "*Junge*" and the participle "*gesehen*" each define one order domain (d_4, d_5, d_6, resp.). Set inclusion gives rise to the domain structure in Fig.2, where the individual words are attached by dashed lines to their including domains (d_1 and d_4 collapse, being identical).[1]

3.2 Surface Ordering

How is the surface order derived from an order domain structure? First of all, the ordering of domains is inherited by their respective elements, i.e., "*Mann*" precedes (any element of) d_2, "*hat*" follows (any element of) d_1, etc.

Ordering within a domain, e.g., of "*hat*" and d_6, or d_5 and d_6, is based on precedence predicates (adapting the precedence predicates of WG). There are two different types, one ordering a word w.r.t. any other element of the domain it is associated with (e.g., "*hat*" w.r.t. d_6), and another ordering two modifiers, referring to the dependency relations they occupy (d_5 and d_6, referring to `subj` and `vpart`). A verb like "*hat*" introduces two precedence predicates, requiring other words to follow itself and the participle to follow subject and object, resp.:[2]

$$\text{"}hat\text{"} \Rightarrow (<_{*} \wedge \langle \texttt{vpart} \rangle >_{\{\texttt{subj},\texttt{obj}\}})$$

[1]Note that in this case, we have not a single rooted tree, but rather an ordered sequence of trees (by virtue of ordering d_1, d_2, and d_3) as domain structure. In general, we assume the sentence period to govern the finite verb and to introduce a single domain for the complete sentence.

[2]For details of the notation, please refer to Sec. 4.

Informally, the first conjunct is satisfied by any domain in which no word precedes "*hat*", and the second conjunct is satisfied by any domain in which no subject or object follows a participle. The domain structure in Fig.2 satisfies these restrictions since nothing follows the participle, and because "*den Mann*" is not an element of d_2, which contains "*hat*". This is an important interaction of order domains and precedence predicates: Order domains define scopes for precedence predicates. In this way, we take into account that dependency trees are flatter than PS-based ones[3] and avoid the formal inconsistencies noted above for WG.

3.3 Linking Domain Structure and Dependency Tree

Order domains easily extend to discontinuous dependencies. Consider the non-projective tree in Fig.1. Assuming that the finite verb governs the participle, no projective dependency between the object "*den Mann*" and the participle "*gesehen*" can be established. We allow non-projectivity by loosening the linking between dependency tree and domain structure: A modifier (e.g., "*Mann*") may not only be inserted into a domain associated with its direct head ("*gesehen*"), but also into a domain of a transitive head ("*hat*"), which we will call the *positional head.*

The possibility of inserting a word into a domain of some transitive head raises the questions of how to require contiguity (as needed in most cases), and how to limit the distance between the governor and the modifier in the case of discontinuity. From a descriptive viewpoint, the *syntactic construction* is often cited to determine the possibility and scope of discontinuities (Bhatt, 1990; Matthews, 1981). In PS-based accounts, the construction is represented by phrasal categories, and extraction is limited by bounding nodes (e.g., Haegeman (1994), Becker et al. (1991)). In dependency-based accounts, the construction is represented by the dependency relation, which is typed or labelled to indicate constructional distinctions which are configurationally defined in PSG. Given this correspondence, it is natural to employ dependencies in the description of discontinuities as fol-

[3]Note that each phrasal level in PS-based trees defines a scope for linear precedence rules, which only apply to sister nodes.

lows: For each modifier of a certain head, a set of dependency types is defined which may link the direct head and the positional head of the modifier ("*gesehen*" and "*hat*", resp.). If this set is empty, both heads are identical and a contiguous attachment results. The impossibility of extraction from, e.g., a finite verb phrase may follow from the fact that the dependency embedding finite verbs, propo, may not appear on any path between a direct and a positional head.[4]

4 The Description Language

This section sketches a logical language describing the dependency structure. It is based on modal logic and owes much to work of Blackburn (1994). As he argues, standard Kripke models can be regarded as directed graphs with node annotations. We will use this interpretation to represent dependency structures. Dependencies and the mapping from dependency tree to order domain structure are described by modal operators, while simple properties such as word class, features, and cardinality of order domains are described by modal propositions.

4.1 Model Structures

In the following, we assume a set of words, $\mathcal{W}$, ordered by a precedence relation, $\prec$, a set of dependency types, $\mathcal{D}$, a set of atomic feature values $\mathcal{A}$, and a set of word classes, $\mathcal{C}$. We define a family of dependency relations $R_d \subset \mathcal{W} \times \mathcal{W}, d \in \mathcal{D}$ and for convenience abbreviate the union $\bigcup_{d \in \mathcal{D}} R_d$ as $R_{\mathcal{D}}$.

Def: *A* dependency tree *is a tuple* $\langle \mathcal{W}, w_r, R_{\mathcal{D}}, V_{\mathcal{A}}, V_{\mathcal{C}} \rangle$, *where* $R_{\mathcal{D}}$ *forms a tree over* $\mathcal{W}$ *rooted in* w_r, $V_{\mathcal{A}} : \mathcal{W} \mapsto 2^{\mathcal{A}}$ *maps words to sets of features, and* $V_{\mathcal{C}} : \mathcal{W} \mapsto \mathcal{C}$ *maps words to word classes.*

Def: *An* order domain (over $\mathcal{W}$) m *is a set of words from* $\mathcal{W}$ *where* $\forall w_1, w_2, w_3 \in \mathcal{W} : (w_1 \prec w_2 \prec w_3 \wedge w_1 \in m \wedge w_3 \in m) \Rightarrow w_2 \in m$.

Def: *An* order domain structure (over $\mathcal{W}$) $\mathcal{M}$ *is a set of order domains where* $\forall m, m' \in \mathcal{M} : m \cap m' = \emptyset \vee m \subseteq m' \vee m' \subseteq m$.

[4]One review pointed out that some verbs may allow extractions, i.e., that this restriction is lexical, not universal. This fact can easily be accomodated because the possibility of discontinuity (and the dependency types across which the modifier may be extracted) is described in the lexical entry of the verb. In fact, a universal restriction could not even be stated because the treatment is completely lexicalized.

Def: *A* dependency structure T *is a tuple* $\langle \mathcal{W}, w_r, R_{\mathcal{D}}, V_{\mathcal{A}}, V_{\mathcal{C}}, \mathcal{M}, V_{\mathcal{M}} \rangle$ *where* $\langle \mathcal{W}, w_r, R_{\mathcal{D}}, V_{\mathcal{A}}, V_{\mathcal{C}} \rangle$ *is a dependency tree,* $\mathcal{M}$ *is an order domain structure over* $\mathcal{W}$, *and* $V_{\mathcal{M}} : \mathcal{W} \mapsto \mathcal{M}^n$ *maps words to order domain sequences.*

Additionally, we require for a dependency structure four more conditions: (1) Each word w is contained in exactly one of the domains from $V_{\mathcal{M}}(w)$, (2) all domains in $V_{\mathcal{M}}(w)$ are pairwise disjoint, (3) each word (except w_r) is contained in at least two domains, one of which is associated with a (transitive) head, and (4) the (partial) ordering of domains (as described by $V_{\mathcal{M}}$) is consistent with the precedence of the words contained in the domains (see (Bröker, 1997) for more details).

4.2 The Language $\mathcal{L}_{\mathcal{D}}$

Fig.3 defines the logical language $\mathcal{L}_{\mathcal{D}}$ used to describe dependency structures. Although they have been presented differently, they can easily be rewritten as (multimodal) Kripke models: The dependency relation R_d is represented as modality $\langle d \rangle$ and the mapping from a word to its ith order domain as modality $\diamond^i_{\mathcal{M}}$.[5] All other formulae denote properties of nodes, and can be formulated as unary predicates -- most evident for word class and feature assignment. For the precedence predicates $<_*$ and $<_\delta$, there are inverses $>_*$ and $>_\delta$. For presentation, the relation *places* $\subset \mathcal{W} \times \mathcal{W}$ has been introduced, which holds between two words iff the first argument is the positional head of the second argument.

A more elaborate definition of dependency structures and $\mathcal{L}_{\mathcal{D}}$ defines two more dimensions, a feature graph mapped off the dependency tree much like the proposal of Blackburn (1994), and a conceptual representation based on terminological logic, linking content words with reference objects and dependencies with conceptual roles.

5 The German Clause

Traditionally, the German main clause is described using three topological fields; the initial and middle fields are separated by the finite (auxiliary) verb, and the middle and the

[5]The modality $\Box^i_{\mathcal{M}}$ can be viewed as an abbreviation of $\diamond^i_{\mathcal{M}} \Box_{\mathcal{M}}$, composed of a mapping from a word to its ith order domain and from that domain to all its elements.

Syntax (valid formulae)	Semantics (satisfaction relation)	
$c \in \mathcal{L_D}, \forall c \in \mathcal{C}$	$T, w \models c$	$:\Leftrightarrow c = V_\mathcal{C}(w)$
$a \in \mathcal{L_D}, \forall a \in \mathcal{A}$	$T, w \models a$	$:\Leftrightarrow a \in V_\mathcal{A}(w)$
$\langle d \rangle \, \phi \in \mathcal{L_D}, \forall d \in \mathcal{D}, \phi \in \mathcal{L_D}$	$T, w \models \langle d \rangle \, \phi$	$:\Leftrightarrow \exists w' \in \mathcal{W} : wR_d w' \wedge T, w' \models \phi$
$<_* \in \mathcal{L_D},$	$T, w \models <_*$	$:\Leftrightarrow \exists m \in \mathcal{M} : (V_\mathcal{M}(w) = \langle \cdots m \cdots \rangle$ $\wedge \forall w' \in m : (w = w' \vee w \prec w'))$
$<_\delta \in \mathcal{L_D}, \forall \delta \subseteq \mathcal{D}$	$T, w \models <_\delta$	$:\Leftrightarrow \neg \exists w', w'', w''' \in \mathcal{W} : places(w', w)$ $\wedge places(w', w'') \wedge w''' R_\delta w \wedge w''' \prec w$
$\uparrow_\delta \in \mathcal{L_D}, \forall \delta \subset \mathcal{D}$	$T, w \models \uparrow_\delta$	$:\Leftrightarrow \exists w', w'' \in \mathcal{W} : wR_\mathcal{D} w \wedge$ $places(w'', w) \wedge w'' R^*_\delta w' o$
$\diamond^i_\mathcal{M}\texttt{single} \in \mathcal{L_D}, \forall i \in I\!N$	$T, w \models \diamond^i_\mathcal{M}\texttt{single}$	$:\Leftrightarrow \left\| \left\{ w' \,\middle\|\, \begin{array}{l} w' \in \Phi_i(V_\mathcal{M}(w)) \wedge \\ \neg \exists w'' : (w'' R_\mathcal{D} w' \wedge \\ w'' \in \Phi_i(V_\mathcal{M}(w))) \end{array} \right\} \right\| \leq 1$
$\diamond^i_\mathcal{M}\texttt{filled} \in \mathcal{L_D}, \forall i \in I\!N$	$T, w \models \diamond^i_\mathcal{M}\texttt{filled}$	$:\Leftrightarrow \mid \Phi_i(V_\mathcal{M}(w)) \mid \geq 1$
$\Box^i_\mathcal{M} a \in \mathcal{L_D}, \forall i \in I\!N, a \in \mathcal{A}$	$T, w \models \Box^i_\mathcal{M} a$	$:\Leftrightarrow \forall w' \in \Phi_i(V_\mathcal{M}(w)) : T, w' \models a$
$\phi \wedge \psi \in \mathcal{L_D}, \forall \phi, \psi \in \mathcal{L_D}$	$T, w \models \phi \wedge \psi$	$:\Leftrightarrow T, w \models \phi$ and $T, w \models \psi$
$\neg \phi \in \mathcal{L_D}, \forall \phi \in \mathcal{L_D}$	$T, w \models \neg \phi$	$:\Leftrightarrow$ not $T, w \models \psi$

Figure 3: Syntax and Semantics of $\mathcal{L_D}$ Formulae

$$
\begin{array}{lr}
\texttt{Vfin} \Rightarrow \diamond^1_\mathcal{M}(\texttt{single} \wedge \texttt{filled}) \wedge \Box^1_\mathcal{M}\texttt{initial} & [1] \\
\quad \wedge \Box^2_\mathcal{M}(\texttt{middle} \wedge \texttt{norel}) & [2] \\
\quad \wedge \diamond^3_\mathcal{M}\texttt{single} \wedge \Box^3_\mathcal{M}(\texttt{final} \wedge \texttt{norel}) & [3] \\
\quad \wedge \texttt{V2} \Leftrightarrow (\texttt{middle} \wedge <_* \wedge \Box^1_\mathcal{M}\texttt{norel}) & [4] \\
\quad \wedge \texttt{VEnd} \Leftrightarrow (\texttt{middle} \wedge >_*) & [5] \\
\quad \wedge \texttt{V1} \Leftrightarrow (\texttt{initial} \wedge \texttt{norel}) & [6]
\end{array}
$$

Figure 4: Domain Description of finite verbs

$$
\begin{array}{lr}
\text{“\textit{hat}”} \wedge \texttt{Vfin} & [7] \\
\wedge \ \langle \texttt{subj} \rangle \, (\text{“\textit{Junge}”} \wedge \uparrow_\emptyset) & [8] \\
\wedge \langle \texttt{vpart} \rangle \, (\text{“\textit{gesehen}”} \wedge \uparrow_\emptyset & [9] \\
\qquad \wedge \neg\texttt{final} \wedge \ >_{\{\texttt{subj},\texttt{obj}\}} & [10] \\
\qquad \wedge \langle \texttt{obj} \rangle \, (\text{“\textit{Mann}”} \wedge \uparrow_{\{\texttt{vpart}\}})) & [11]
\end{array}
$$

Figure 5: Hierachical Structure

final fields by infinite verb parts such as separable prefixes or participles. We will generalize this field structure to verb-initial and verb-final clauses as well, without going into the linguistic motivation due to space limits.

The formula in Fig.4 states that all finite verbs (word class Vfin $\in \mathcal{C}$) define three order domains, of which the first requires exactly one element with the feature initial [1], the second allows an unspecified number of elements with features middle and norel [2], and the third allows at most one element with features final and norel [3]. The features initial, middle, and final $\in \mathcal{A}$ serve to restrict placement of certain phrases in specific fields; e.g., no reflexive pronouns can appear in the final field. The norel $\in \mathcal{A}$ feature controls placement of a relative NP or PP, which may appear in the initial field only in verb-final clauses. The order types are defined as follows: In a verb-second clause (feature V2), the verb is placed at the beginning ($<_*$) of the middle field (middle), and the element of the initial field cannot be a relative phrase ($\diamond^1_\mathcal{M}$norel in [4]). In a verb-final clause (VEnd), the verb is placed at the end ($>_*$) of the middle field, with no restrictions for the initial field (relative clauses and non-relative verb-final clauses are subordinated to the noun and conjunction, resp.) [5]. In a verb-initial clause (V1), the verb occupies the initial field [6].

The formula in Fig.5 encodes the hierarchical structure from Fig.1 and contains lexical restrictions on placement and extraction (the surface is used to identify the word). Given this, the order type of "*hat*" is determined as follows: The participle may not be extraposed ($\neg$final in [10]; a restriction from the lexical entry of "*hat*"), it must follow "*hat*" in d_2. Thus, the verb cannot be of order type VEnd, which would require it to be the last element in its domain ($>_*$ in [5]). "*Mann*" is not adjacent to "*gesehen*", but may be extracted across the dependency vpart ($\uparrow_{\{\texttt{vpart}\}}$ in [11]), allowing its insertion into a domain defined by "*hat*". It cannot precede "*hat*" in d_2, because "*hat*" must either begin d_2 (due to $<_*$ in [4]) or itself go into d_1. But d_1 allows only one phrase (single), leaving only the domain structure from Fig.2, and thus the order type V2 for "*hat*".

6 Comparison to PSG Approaches

One feature of word order domains is that they factor ordering alternatives from the syntactic tree, much like feature annotations do for morphological alternatives. Other lexicalized grammars collapse syntactic and ordering information and are forced to represent ordering alternatives by lexical ambiguity, most notable L-TAG (Schabes et al., 1988) and some versions of CG (Hepple, 1994). This is not necessary in our approach, which drastically reduces the search space for parsing.

This property is shared by the proposal of Reape (1993) to associate HPSG signs with sequences of constituents, also called word order domains. Surface ordering is determined by the sequence of constituents associated with the root node. The order domain of a mother node is the sequence union of the order domains of the daughter nodes, which means that the relative order of elements in an order domain is retained, but material from several domains may be interleaved, resulting in discontinuities. Whether an order domain allows interleaving with other domains is a parameter of the constituent. This approach is very similar to ours in that order domains separate word order from the syntactic tree, but there is one important difference: Word order domains in HPSG do not completely free the hierarchical structure from ordering considerations, because discontinuity is specified per phrase, not per modifier. For example, two projections are required for an NP, the lower one for the continuous material (determiner, adjective, noun, genitival and prepositional attributes) and the higher one for the possibly discontinuous relative clause. This dependence of hierarchical structure on ordering is absent from our proposal.

We may also compare our approach with the projection architecture of LFG (Kaplan & Bresnan, 1982; Kaplan, 1995). There is a close similarity of the LFG projections (c-structure and f-structure) to the dimensions used here (order domain structure and dependency tree, respectively). C-structure and order domains represent surface ordering, whereas f-structure and dependency tree show the subcategorization or valence requirements. What is more, these projections or dimensions are linked in both accounts by an element-wise mapping. The difference between the two architectures lies in the linkage of the projections or dimensions: LFG maps f-structure off c-structure. In contrast, the dependency relation is taken to be primitive here, and ordering restrictions are taken to be indicators or consequences of dependency relations (see also Bröker (1998b, 1998a)).

7 Conclusion

We have presented an approach to word order for DG which combines traditional notions (semantically motivated dependencies, topological fields) with contemporary techniques (logical description language, model-theoretic semantics). Word order domains are sets of partially ordered words associated with words. A word is contained in an order domain of its head, or may float into an order domain of a transitive head, resulting in a discontinuous dependency tree while retaining a projective order domain structure. Restrictions on the floating are expressed in a lexicalized fashion in terms of dependency relations. An important benefit is that the proposal is lexicalized without reverting to lexical ambiguity to represent order variation, thus profiting even more from the efficiency considerations discussed by Schabes et al. (1988).

It is not yet clear what the generative capacity of such lexicalized discontinuous DGs is, but at least some index languages (such as $a^n b^n c^n$) can be characterized. Neuhaus & Bröker (1997) have shown that recognition and parsing of such grammars is $\mathcal{NP}$-complete. A parser operating on the model structures is described in (Hahn et al., 1997).

References

Becker, T., A. Joshi & O. Rambow (1991). Long-Distance scrambling and tree-adjoining grammar. In *Proc. 5th Conf. of the European Chapter of the ACL*, pp. 21–26.

Bhatt, C. (1990). *Die syntaktische Struktur der Nominalphrase im Deutschen.* Studien zur deutschen Grammatik 38. Tübingen: Narr.

Blackburn, P. (1994). Structures, Languages and Translations: The Structural Approach to Feature Logic. In C. Rupp, M. Rosner & R. Johnson (Eds.), *Constraints, Language and Computation*, pp. 1–27. London: Academic Press.

Bröker, N. (1997). *Eine Dependenzgrammatik zur Kopplung heterogener Wissenssysteme auf modallogischer Basis.* Dissertation, Deutsches Seminar, Universität Freiburg.

Bröker, N. (1998a). How to define a context-free backbone for DGs: An experiment in grammar conversion. In *Proc. of the COLING-ACL'98 workshop "Processing of Dependency-based Grammars".* Montreal/CAN, Aug 15, 1998.

Bröker, N. (1998b). A Projection Architecture for Dependency Grammar and How it Compares to LFG. In *Proc. 1998 Int'l Lexical-Functional Grammar Conference.* (accepted as alternate paper) Brisbane/AUS: Jun 30-Jul 2, 1998.

Eisner, J. (1997). Bilexical Grammars and a Cubic-Time Probabilistic Parser. In *Proc. of Int'l Workshop on Parsing Technologies*, pp. 54–65. Boston/MA: MIT.

Gaifman, H. (1965). Dependency Systems and Phrase Structure Systems. *Information and Control*, 8:304–337.

Haegeman, L. (1994). *Introduction to Government and Binding.* Oxford/UK: Basil Blackwell.

Hahn, U., P. Neuhaus & N. Bröker (1997). Message-Passing Protocols for Real-World Parsing – An Object-Oriented Model and its Preliminary Evaluation. In *Proc. Int'l Workshop on Parsing Technology*, pp. 101–112. Boston/MA: MIT, Sep 17-21, 1997.

Hellwig, P. (1986). Dependency Unification Grammar. In *Proc. 11th Int'l Conf. on Computational Linguistics*, pp. 195–198.

Hepple, M. (1994). Discontinuity and the Lambek Calculus. In *Proc. 15th Int'l Conf. on Computational Linguistics*, pp. 1235–1239. Kyoto/JP.

Hudson, R. (1990). *English Word Grammar.* Oxford/UK: Basil Blackwell.

Hudson, R. (1993). Recent developments in dependency theory. In J. Jacobs, A. v. Stechow, W. Sternefeld & T. Vennemann (Eds.), *Syntax. Ein internationales Handbuch zeitgenössischer Forschung*, pp. 329–338. Berlin: Walter de Gruyter.

Kaplan, R. (1995). The formal architecture of Lexical-Functional Grammar. In M. Dalrymple, R. Kaplan, J. I. Maxwell & A. Zaenen (Eds.), *Formal Issues in Lexical-Functional Grammar*, pp. 7–27. Stanford University.

Kaplan, R. & J. Bresnan (1982). Lexical-Functional Grammar: A Formal System for Grammatical Representation. In J. Bresnan & R. Kaplan (Eds.), *The Mental Representation of Grammatical Relations*, pp. 173–281. Cambridge, MA: MIT Press.

Kruijff, G.-J. v. (1997). *A Basic Dependency-Based Logical Grammar.* Draft Manuscript. Prague: Charles University.

Maruyama, H. (1990). Structural Disambiguation with Constraint Propagation. In *Proc. 28th Annual Meeting of the ACL*, pp. 31–38. Pittsburgh/PA.

Matthews, P. (1981). *Syntax.* Cambridge Textbooks in Linguistics, Cambridge/UK: Cambridge Univ. Press.

McCord, M. (1990). Slot Grammar: A System for Simpler Construction of Practical Natural Language Grammars. In R. Studer (Ed.), *Natural Language and Logic*, pp. 118–145. Berlin, Heidelberg: Springer.

Melc'ŭk, I. (1988). *Dependency Syntax: Theory and Practice.* Albany/NY: State Univ. Press of New York.

Melc'ŭk, I. & N. Pertsov (1987). *Surface Syntax of English: A Formal Model within the MTT Framework.* Philadelphia/PA: John Benjamins.

Neuhaus, P. & N. Bröker (1997). The Complexity of Recognition of Linguistically Adequate Dependency Grammars. In *Proc. 35th Annual Meeting of the ACL and 8th Conf. of the EACL*, pp. 337–343. Madrid, July 7-12, 1997.

Reape, M. (1993). *A Formal Theory of Word Order: A Case Study in West Germanic.* Doctoral Dissertation. Univ. of Edinburg.

Schabes, Y., A. Abeille & A. Joshi (1988). Parsing Strategies with 'Lexicalized' Grammars: Application to TAGs. In *Proc. 12th Int'l Conf. on Computational Linguistics*, pp. 578–583.

Sgall, P., E. Hajicova & J. Panevova (1986). *The Meaning of the Sentence in its Semantic and Pragmatic Aspects.* Dordrecht/NL: D.Reidel.

Tesnière, L. (1959). *Eleménts de syntaxe structurale.* Paris: Klincksiek.