Movement as well-formedness conditions

Ralph Debusmann
Computational Linguistics
Saarland University
rade@coli.uni-sb.de

Abstract

We introduce a new dependency-based grammar formalism recently suggested in (Duchier & Debusmann 2001*b*), (Duchier 2001) and (Debusmann 2001). The grammar formalism is called Topological Dependency Grammar (TDG). TDG shares with GB (Chomsky 1986) a notion of *movement*. In GB, movement is carried out by tree transformations. In TDG, it is the effect of well-formedness conditions on dependency trees and does not involve transformations. We illustrate both kinds of movement by showing how the two theories account for the three German clause types. Then, we point out the similarities between GB and TDG, and raise the question whether GB's transformational notion of movement could be replaced by TDG's constraint-based account of movement.

1 Introduction

In this article, we introduce a new dependency-based grammar formalism recently suggested in (Duchier & Debusmann 2001*b*), (Duchier 2001) and (Debusmann 2001). The grammar formalism is called Topological Dependency Grammar (TDG). TDG not only allows to specify elegant analyses for languages with freer word order but also to process them, and efficiently so: (Duchier & Debusmann 2001*a*) is a first example of an efficient constraint-based TDG parser implementation.

A TDG analysis consists of two tree structures: a *syntactic dependency tree* (*ID tree*, where *ID* stands for *immediate dominance*) and a *topological dependency tree* (*LP tree*, *linear precedence*). The ID tree is a dependency tree in the spirit of (Tesnière 1959) whose edges are labeled by grammatical roles. ID trees are unordered (and hence in a sense non-projective), as opposed to LP trees which are ordered and projective. The LP tree is inspired by topological fields theory (Herling 1821), (Erdmann 1886), (Höhle 1986). Its shape is essentially a flattening of the ID tree, and its edges are labeled by topological fields.

The approach taken in TDG is very similar to other recent dependency-based approaches tackling discontinuity, such as (Bröker 1998), (Kahane, Nasr & Rambow 1998) and (Gerdes & Kahane 2001). It is also reminiscent of HPSG-based theories by (Reape 1994), (Müller 1999) and (Kathol 2000). TDG's strong resemblance with Government-Binding (GB) theory (Chomsky 1986) is probably less obvious. We will show that, above all, TDG shares with GB a notion of *movement*. In GB, movement is carried out by tree transformations, which have properties undesirable for parsing, besides other problems. In TDG, movement is the effect of well-formedness conditions on finite labeled trees and does not involve transformations.

The outline of this article is as follows. We start out by presenting the essentials of topological fields theory in section 2. In section 3, we show how the three German clause types are analyzed within the GB paradigm. In section 4, we introduce the new TDG grammar formalism and give TDG-analyses of the German clause types. Finally, in section 5, we compare GB and TDG, and hint at the possibility to reformulate parts of GB, movement in particular, in a constraint-based way.

2 Topological Fields Theory

Both GB and TDG borrow ideas from *Topological Fields Theory* (TFT). TFT is a descriptive theory of German word order and has a long tradition in German linguistics reaching back to (Herling 1821) and (Erdmann 1886), as (Höhle 1986) shows.

2.1 German clause types

There are three clause types in German: *verb-first*, *verb-second* and *verb-final*. We give examples of the clause types in (1), (2) and (3) below:

Hat	Maria	einen	Mann	geliebt?	
Has	Maria	a	man(acc)	loved?	(1)

"Has Maria loved a man?"

Einen	Mann	hat	Maria	geliebt.	
A	man(acc)	has	Maria	loved.	(2)

"Maria has loved a man."

dass	Maria	einen	Mann	geliebt	hat.	
that	Maria	a	man(acc)	loved	has.	(3)

"that Maria has loved a man."

The clause types' names reflect the respective position of the finite verb: in a verb-first clause, the finite verb comes first, as *hat* in (1). In a verb-second clause, the finite verb is in second position, preceded by a constituent of arbitrary type (2). In a verb-final clause, the finite verb comes last (3).

2.2 Topological Fields Theory analysis

A TFT analysis divides a German clause into five continuous substrings and assigns to these substrings one of the five *topological fields*: *Vorfeld*, *left sentence bracket* ("("), *Mittelfeld*, *right sentence bracket* (")") and *Nachfeld*. Here are analyses of examples (1–3) in terms of TFT:

Vorfeld	(	Mittelfeld	)	Nachfeld
	Hat	*Maria einen Mann*	*geliebt?*	
Einen Mann	*hat*	*Maria*	*geliebt.*	
	dass	*Maria einen Mann*	*geliebt hat.*	

(4)

The Vorfeld is occupied only in verb-second clauses, but only by at most one constituent. In verb-first and verb-final clauses, the Vorfeld is empty. The left sentence bracket is occupied by the finite verb in verb-first and verb-second clauses, and by the complementizer (here: *dass*) in verb-final clauses. The right sentence bracket is occupied by the remaining verbal material (excluding the finite verb in verb-first and verb-second clauses) and is often called *verb cluster*. The Mittelfeld, surrounded by the left and right sentence brackets, is occupied by all of the remaining non-verbal material (excluding the fronted material in verb-second clauses). The Nachfeld to the very right is occupied by extraposed material such as relative clauses and subordinate clauses.[1]

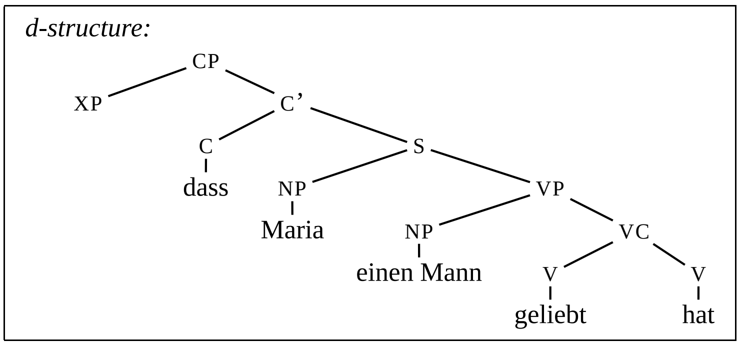
3 The German clause: a GB analysis

3.1 Verb-final clauses

In this section, we outline the GB approach to German syntax elaborated in (Grewendorf 1988). Grewendorf takes the popular view that verb-first and verb-second clauses are derived from the basic verb-final clause structure. The d-structure analysis of the example

[1]In the examples in this paper, the Nachfeld is always empty for simplicity.

verb-final clause (3) is depicted below:[2][3]

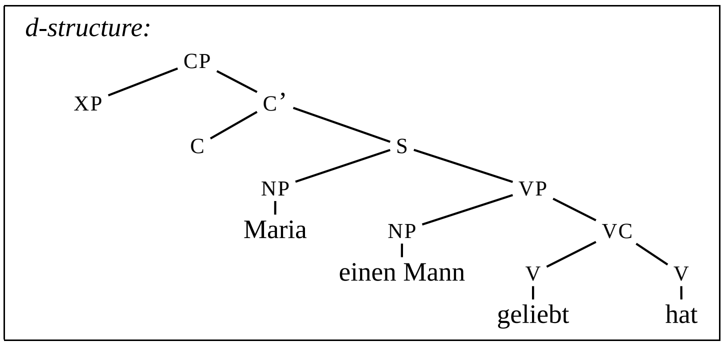


(5)

The s-structure analysis of (3) is the same as the d-structure. The topological fields of TFT can be mapped to GB-positions as follows: [CP,XP] corresponds to the Vorfeld, [C',C] to the left sentence bracket, [S,NP] and [VP,NP] to the Mittelfeld and [VP,VC] to the verb cluster.

3.2 Verb-first clauses

The d-structure analysis of verb-first clause (1) is the same as (5) above except that the complementizer *dass* is not included:



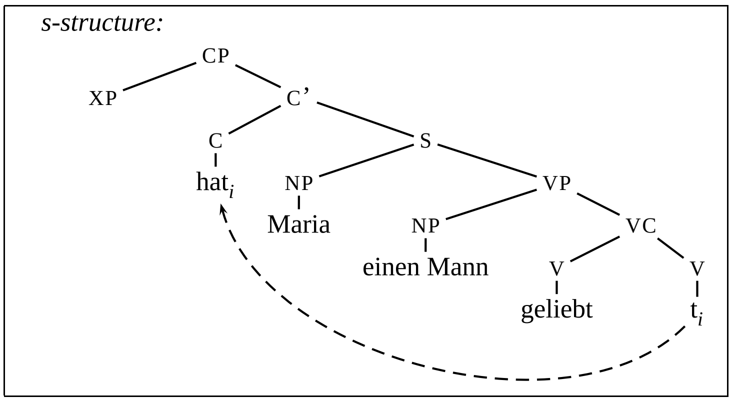
(6)

The s-structure analysis of the verb-first clause (1) is different from the d-structure analysis: here, movement takes place. More specifically, GB's the move-α-operation is used to move nodes into *landing sites*. The number of landing sites is restricted by constraints such as the θ-criterion and the Case-Filter. In the example, [CP,XP] and [C',C] are landing sites. The XP-position [CP,XP] is a landing site only for maximal projections, whereas [C',C] is a head position and therefore only available to heads. In the s-structure analysis

[2]For simplicity, we do not show the IP- and I'-nodes of Grewendorf's basic clause structure tree in (Grewendorf 1988), p. 49.

[3]VC stands for *verb cluster*.

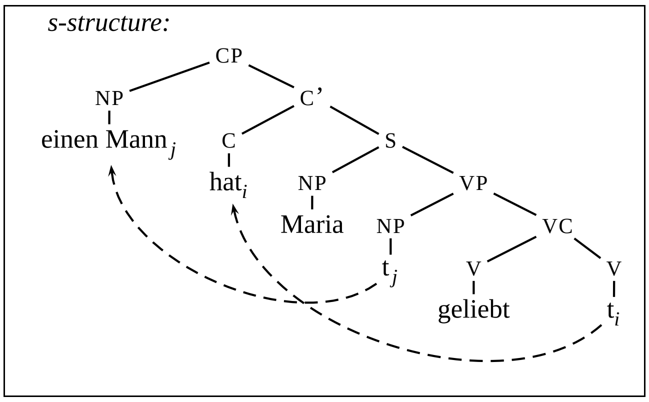
of the verb-first clause (1), move-α moves the finite verb *hat* to the landing site [C',C], as indicated by the dashed arrow:



(7)

3.3 Verb-second clauses

The d-structure analysis of the verb-second clause (2) is the same as of the verb-first clause (1). However, the s-structure analyses are different. In the s-structure analysis of the verb-second clause (2), not only has *hat* moved to [C',C], but also the NP *einen Mann* has moved to the landing site [CP,XP]:

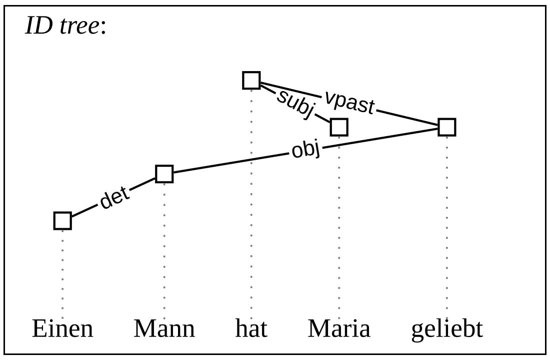


(8)

4 The German clause: a TDG analysis

4.1 ID and LP trees

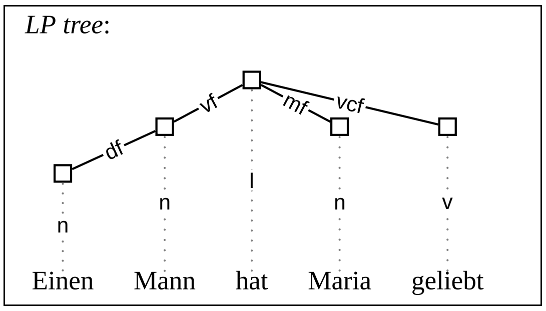
A TDG analysis is a pair of an unordered ID tree and an ordered and projective LP tree. ID and LP trees share the same set of nodes, which correspond one-to-one with words, but have different edges. Below, we give an ID tree analysis of (2):[4]



(9)

ID tree edges are labelled by grammatical functions like subj (for a nominative subject), obj (for an accusative object) vpast (for a past participle complement) and det (for a determiner). The mother of a node in the ID tree is called *syntactic head* and its daughters *syntactic dependents*.

Here is the corresponding topological dependency (LP tree) analysis:



(10)

The mother of a node in the LP tree is called *topological head* and its daughters *topological dependents*.

4.2 Ordering words in the LP tree

TDG employs a set $\mathcal{F}$ of *fields* to determine the licensed linearizations. $\mathcal{F} = \mathcal{F}_\mathsf{E} \uplus \mathcal{F}_\mathsf{N}$, where $\mathcal{F}_\mathsf{E} = \{\mathsf{df}, \mathsf{vf}, \mathsf{mf}, \mathsf{vcf}, \mathsf{nf}\}$ is the set of *edge fields* and $\mathcal{F}_\mathsf{N} = \{\mathsf{n}, \mathsf{l}, \mathsf{v}\}$ is the set of *node fields*.[5] $\mathcal{F}$ is totally ordered, which induces a partial order on LP trees:

1. The topological dependents of each node are ordered by their edge fields.

2. Each node is positioned with respect to its topological dependents by its node field.

[4]Since ID trees are unordered, we can pick an arbitrary linear arrangement for display purposes. In the picture below, we stick to the word order given in example (2).

[5]For simplicity, $\mathcal{F}$ only includes the fields needed to account for our examples. More fields are required for a more complete account of German, as for instance in (Debusmann 2001).

The induced order is partial and not total because if two words land[6] in the same edge field, they remain unordered with respect to each other. We use this partial order to account for the fact that in German, the order of constituents in the Mittelfeld is rather arbitrary.[7]

The set $\mathcal{F}$ of fields is essentially motivated by Topological Fields Theory. For example vf models the Vorfeld, l the left sentence bracket, mf the Mittelfeld, vcf the right sentence bracket and nf the Nachfeld. df and n are used to determine word order within noun phrases: e.g. df stands for *determiner field* and n for *noun field*. The total order on $\mathcal{F}$ is given below:

$$\mathsf{df} \prec \mathsf{n} \prec \mathsf{vf} \prec \mathsf{l} \prec \mathsf{mf} \prec \mathsf{vcf} \prec \mathsf{v} \prec \mathsf{nf} \tag{11}$$

The global total order on $\mathcal{F}$ can be decomposed into local orders. For instance the local order $\mathsf{df} \prec \mathsf{n}$ requires determiners to precede their corresponding nouns. The sequence $\mathsf{vf} \prec \mathsf{l} \prec \mathsf{mf} \prec \mathsf{vcf} \prec \mathsf{nf}$ requires the Vorfeld (vf) to precede the left sentence bracket (l) to precede the Mittelfeld (mf) to precede the right sentence bracket (vcf) to precede the Nachfeld (nf).

In our example LP tree (10), the desired word order is induced as follows:

1. *Mann* lands in the vf, *Maria* in the mf and *geliebt* in the vcf of *hat*. Since $\mathsf{vf} \prec \mathsf{mf} \prec \mathsf{vcf}$ in (11), *Mann* must precede *Maria* and *Maria* must precede *geliebt*.

2. The node field of *hat* is l. Because $\mathsf{vf} \prec \mathsf{l} \prec \mathsf{mf}$ in (11), it must be placed between the *Mann* in the vf and *Maria* in the mf.

4.3 Example lexicon

TDG states well-formedness conditions for LP trees in a lexicalized fashion: a lexical entry stipulates which edge fields are *offered* for topological dependents to land in and which are *accepted*. A node w' can land in edge field f of topological head w iff w *offers* f and w' *accepts* f. A lexical entry also assigns a set of possible node fields to each word. Here are

[6]A node is said to *land* in edge field f iff its incoming edge is labeled with f.

[7]Of course, there *are* constraints or at least preferences on the order of elements in the Mittelfeld in German. However, TDG is not yet equipped with a declarative means to order elements within the same field.

the lexical entries for our example:[8]

	offers	accepts	node fields
dass	{nf}	{vf, nf}	{v}
einen	{}	{df}	{n}
Mann	{df}	{vf, mf}	{n}
Maria	{}	{vf, mf}	{n}
hat	{vf?, mf∗, vcf?, nf?}	{}	{l}
hat	{mf∗, vcf?, nf?}	{nf}	{v}
geliebt	{mf∗, nf?}	{vf}	{v}
geliebt	{}	{vcf}	{v}

(12)

The set of offered edge fields is written in *wildcard notation*: e.g. vf? indicates that there can be at most one topological dependent in the vf (as stated by TFT), and mf∗ that any number of daughters can land in the mf.

Notice that *hat* and *geliebt* are assigned two lexical entries. The upper lexical entry for *hat* is appropriate if *hat* is in the left sentence bracket (node field l). If *hat* is in the left sentence bracket, it is the head of a verb-first or a verb-second clause. The clause is verb-first if the vf of *hat* is empty and verb-second if it is occupied. The lower lexical entry for *hat* applies if *hat* is the head of a subordinate clause. In this position, *hat* does not offer vf.

The upper lexical entry for *geliebt* is appropriate if *geliebt* is fronted into the Vorfeld: it accepts only edge field vf. In this position, *geliebt* offers the fields mf and nf. The lower lexical entry for *geliebt* applies if *geliebt* is in the verb cluster: it accepts only edge field vcf. Here, *geliebt* does not offer any field.

4.4 Climbing

The well-formedness conditions for LP trees are not only constrained by lexicalized constraints but also by a *grammatical principle*[9]:

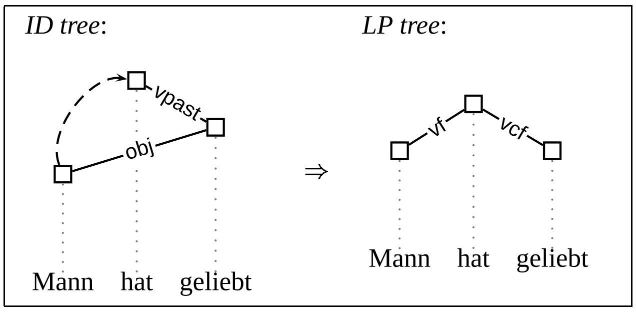
Principle 1 *A node must land on a transitive head*

The principle states that the topological head w of a node w' in the LP tree must be above w' in the ID tree. If a node lands above of its syntactic head, it is said to have *climbed*.

[8]We display only the LP tree part of the lexical entries. Full lexical entries also comprise ID tree information such as subcategorization and agreement.

[9]We only mention the first of the three principles given in (Duchier & Debusmann 2001*b*), (Duchier 2001) and (Debusmann 2001).

Below, we illustrate how *Mann* climbs into the vf of *hat* (as indicated by the dashed arrow):

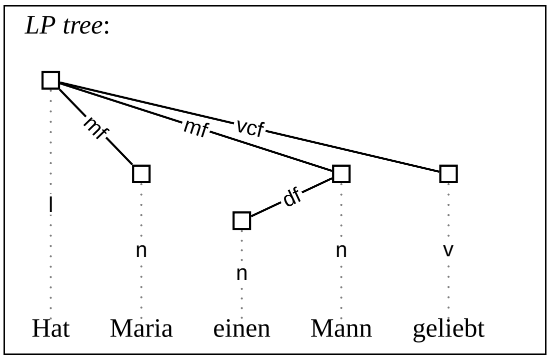


(13)

Note that *Mann* is forced to climb by the lower lexical entry for *geliebt*: it cannot land on *geliebt* because *geliebt* (in verb cluster-position, i.e. if landing in the vcf) offers no field.

4.5 Verb-first clauses

The ID tree analysis of verb-first clause (1) is the same as the one of the verb-second clause (2), given in (9). The LP tree analysis of (1) is however different:



(14)

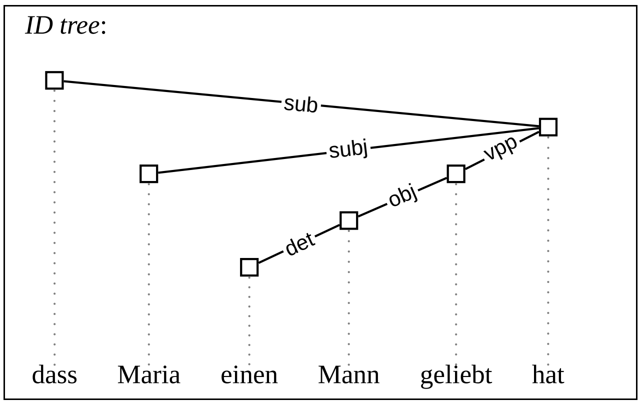
Here, the NP *einen Mann* climbs into the Mittelfeld (mf) instead of the Vorfeld (vf).

4.6 Verb-second clauses

The ID and LP tree analyses of the verb-second clause (2) were given in (9) and (10), respectively.

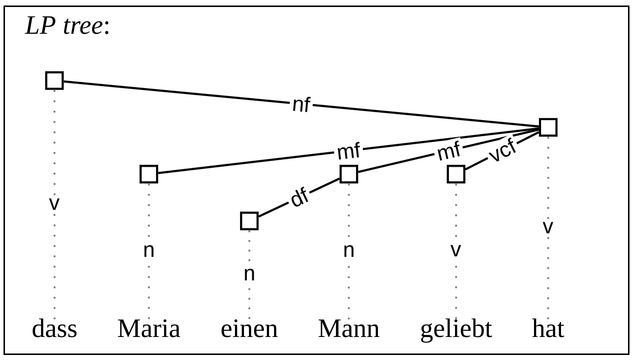
4.7 Verb-final clauses

Below, we show the ID tree analysis of the verb-final clause (3):



(15)

Here, the complementizer *dass* is the root. The corresponding LP tree analysis is displayed below:



(16)

In the LP tree, the finite verb *hat* lands in the nf of *dass*. The NP *einen Mann* climbs into the mf of *hat*.

Notice that this TDG analysis distinguishes verb-first and verb-second clauses from verb-final clauses in a different way as the GB analysis does. For one, the GB analysis of verb-final clauses given in (Grewendorf 1988) correctly predicts that the finite verb *hat* cannot be moved into the left sentence bracket-position [C',C] because this position is already occupied by the complementizer *dass*. However, Grewendorf runs into complications when he explains why the Vorfeld-position [CP,XP] cannot be occupied in verb-final (and verb-first) clauses (Grewendorf 1988, Chapter 11).

In the TDG analysis, verb-first and verb-second clauses are distinguished in a lexical fashion: there are two lexical entries for *hat*: one applies if *hat* is the head of a verb-first or verb-second clause (in the left sentence bracket) and one if *hat* is the head of a verb-final clause (in the right sentence bracket). If *hat* is in the left sentence bracket, it offers an optional vf-position and is positioned to the left of the Mittelfeld (mf) by virtue of having node field l. If *hat* is the head of a verb-final clause (in the right sentence bracket), it does not offer a vf-position and is positioned to the right of the Mittelfeld (in the node field v).

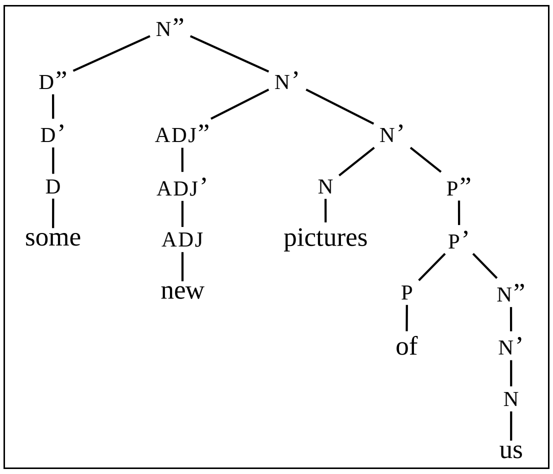
5 GB and TDG: a comparison

After illustrating how GB and TDG analyze the three German clause types, we turn now to a comparison of the two theories.

5.1 Dependency

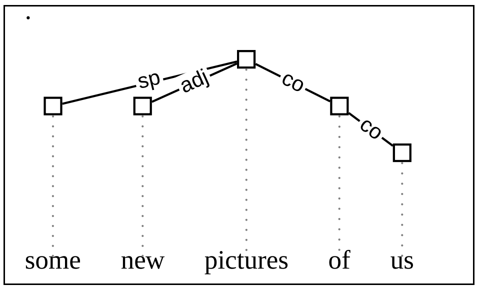
An obvious difference between GB and TDG is that GB is a phrase structure-based theory and TDG dependency-based. But this is no crucial difference: since GB is based on X-bar theory (Jackendoff 1977), it also incorporates the notion of a *head*: X-bar theory requires that every phrase has a head which is a single word. The only exception to this rule is GB's I-node which does not correspond to a word. (Covington 1990) argues that if a phrase structure analysis (1) picks out one node as the head of each phrase and (2) has no labels or features on non-terminal nodes (unless of course copied unchanged from terminal nodes), it can be regarded as being equivalent to a dependency analysis that specifies word order.

(Covington 1992) even goes one step further by attempting to simplify GB theory by recasting it into a dependency formalism. He shows how to convert GB's X-bar-based phrase structure trees into equivalent dependency trees and then redefines government in terms of dependency. As an example, consider the GB phrase structure tree below:



(17)

The head of the phrase is *pictures*, which has a specifier (*some*), an adjunct (the adjective *new*) and a complement (*of*). The complement of *of* is *us*. Here is an equivalent dependency tree:



(18)

where *sp* stands for *specifier*, *adj* for *adjunct* and *co* for *complement.* With respect to a dependency tree, GB's notion of government is now much easier to define:

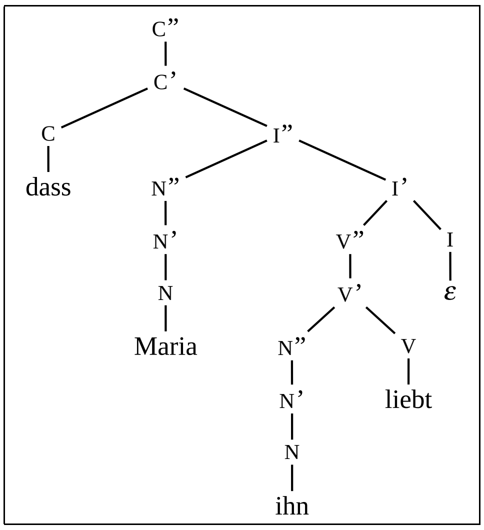
Definition 1 *A governs B iff B is an immediate dependent of A.*

Let us go through another example of recasting an X-bar-based GB phrase structure tree into a much simpler dependency tree. We consider the German subordinate clause below:

dass	Maria	ihn	liebt.
that	Maria	him	loves.

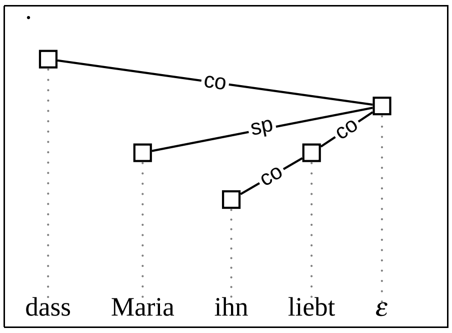
"that Maria loves him." (19)

Here is a GB analysis of the clause:



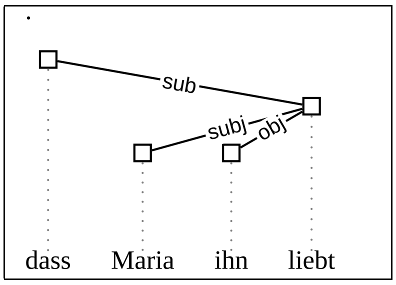
(20)

And here is the corresponding dependency tree, recasted using the algorithm sketched in (Covington 1992):



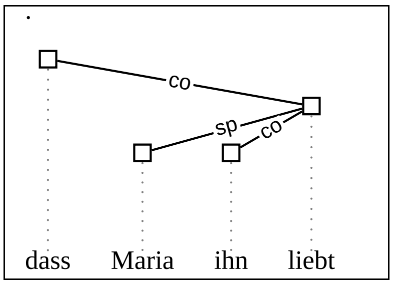
(21)

A typical "real" dependency tree analysis of example (19) is displayed below:



(22)

(21) is already very similar to the "real" dependency tree. If one now removes the empty word ϵ from the recasted dependency tree (21) and assumes that *Maria* is the specifier of *liebt*, the resulting dependency tree becomes equivalent to the "real" dependency analysis (22), except that the edge labels are different:



(23)

5.2 Valency

GB and TDG and in fact most linguistic theories to date share a notion of *valency*. Both GB and TDG state valency requirements in the lexicon: GB uses subcategorization frames to specify the required θ-roles, and a TDG lexicon includes ID tree and LP tree valency. ID tree valency is encoded by stating which syntactic roles a word offers and is very similar to GB's subcategorization frames. For example, a finite transitive verb offers subj and obj. LP tree valency specifies which fields a word offers.

5.3 Constituency

While GB includes the notion of dependency as a derived notion only, constituency is incorporated as a first class citizen. Constituents or phrases in GB are contiguous substrings of the analyzed sentence. TDG includes constituency as a derived notion in both the ID and LP tree. In the ID tree, the set of nodes equal or below a syntactic head can be viewed as a constituent, but one which is not required to be contiguous (since the ID tree is unordered). In the LP tree, the set of nodes equal or below a topological head forms a contiguous substring constituent (because LP trees are ordered and projective).

5.4 Movement

Both theories use a notion of *movement* to relate a deep or syntactic structure (d-structure in GB, ID tree in TDG) to a surface or topological structure (s-structure, LP tree). But while in GB, move-α is a primitive operation modelled by tree transformations, climbing is a derived notion in TDG: it describes the effect of well-formedness conditions. These well-formedness conditions are axiomatized in a constraint-based fashion, as outlined in (Duchier 1999) and (Duchier 2000), and can be easily turned into a parser.

GB restricts the applicability of move-α by providing a fixed set of kinds of movement, including *wh*- and NP-movement. Movement is further constrained by general principles such as the Case Filter and the θ-criterion. For instance, only $\overline{\theta}$-positions[10] may function as *landing sites* for movement in GB. XP-positions are landing sites for maximal projections only (e.g. [CP,XP]) and head-positions for heads.

TDG constraints movement in a lexicalized way. Only a small number of grammatical principles are postulated and the remaining work is done in the lexicon: a lexical entry stipulates which fields are *offered* and which are *accepted*. Making the connection to GB again, the notion of offered fields is very similar to GB's *landing sites*.

6 Conclusion

The new dependency grammar-based framework described in TDG employs concepts which are very similar to concepts in GB theory. Above all, both GB and TDG use a notion of *movement* to mediate between levels of syntax and linear precedence. But while GB models movement as tree transformations, movement in TDG is the consequence of well-formedness conditions.

We demonstrated with analyses of the three German clause types that on a descriptive level, the notions of movement in GB and TDG are yet very similar. This suggests that GB's approach to movement could be reformulated in a way similar to TDG's constraint-based approach. A non-transformational account of movement based on well-formedness

[10] A $\overline{\theta}$-position is a position which is not assigned a θ-role.

conditions would make GB much more attractive from a computational point of view, and could make use of techniques developed for TDG, including an efficient treatment of ambiguity using finite-set constraints.

References

Bröker, N. (1998), Separating surface order and syntactic relations in a dependency grammar, *in* 'COLING-ACL 98 - Proc. of the 17th Intl. Conf. on Computational Linguistics and 36th Annual Meeting of the ACL.', Montreal/CAN.

Chomsky, N. (1986), *Barriers*, Linguistic Inquiry Monograph 13, MIT Press, Cambridge/MA.

Covington, M. A. (1990), A dependency parser for variable-word-order languages, Research Report AI-1990-01, Artificial Intelligence Programs, University of Georgia, Athens/GA.

Covington, M. A. (1992), GB theory as dependency grammar, Research Report AI-1992-03, Artificial Intelligence Program, University of Georgia, Athens/GA.

Debusmann, R. (2001), A declarative grammar formalism for dependency grammar, Master's thesis, University of Saarland.

Duchier, D. (1999), Axiomatizing dependency parsing using set constraints, *in* 'Sixth Meeting on the Mathematics of Language', Orlando/FL.

Duchier, D. (2000), 'Configuration of labeled trees under lexicalized constraints and principles', To appear in the Journal of Language and Computation.

Duchier, D. (2001), Lexicalized syntax and topology for non-projective dependency grammar, *in* 'Eighth Meeting on Mathematics of Language', Helsinki/FIN.

Duchier, D. & Debusmann, R. (2001*a*), 'Topological dependency grammar 1.0'. http://www.ps.uni-sb.de/∼duchier/mogul/info/duchier/coli/dg.html.

Duchier, D. & Debusmann, R. (2001*b*), Topological dependency trees: A constraint-based account of linear precedence, *in* '39th Annual Meeting of the Association for Computational Linguistics (ACL 2001)', Toulouse, France.

Erdmann, O. (1886), *Grundzüge der deutschen Syntax nach ihrer geschichtlichen Entwicklung dargestellt*, Erste Abteilung, Stuttgart/FRG.

Gerdes, K. & Kahane, S. (2001), Word order in german: A formal dependency grammar using a topological hierarchy, *in* '39th Annual Meeting of the Association for Computational Linguistics (ACL 2001)', Toulouse/FRA. To appear.

Grewendorf, G. (1988), *Aspekte der deutschen Syntax. Eine Rektions-Bindungs-Analyse*, Studien zur deutschen Grammatik 33, Gunter Narr, Tübingen/FRG.

Herling, S. (1821), 'Über die Topik der deutschen Sprache'.

Höhle, T. (1986), Der Begriff "Mittelfeld", Anmerkungen über die Theorie der topologischen Felder, *in* W. Weiss, H. E. Wiegand & M. Reis, eds, 'Akten des 7. Internationalen Germanisten-Kongresses, Göttingen 1985', Vol. 3, Max Niemeyer Verlag, Tübingen/FRG, pp. 329–340.

Jackendoff, R. (1977), *$\bar{X}$ Syntax: A Study of Phrase Structure*, number 2 *in* 'Linguistic Inquiry Monographs', MIT Press, Cambridge/MA.

Kahane, S., Nasr, A. & Rambow, O. (1998), Pseudo-projectivity: a polynomially parsable non-projective dependency grammar, *in* '36th Annual Meeting of the Association for Computational Linguistics (ACL 1998)', Montreal/CAN.

Kathol, A. (2000), *Linear Syntax*, Oxford University Press.

Müller, S. (1999), *Deutsche Syntax deklarativ. Head-Driven Phrase Structure Grammar für das Deutsche*, Linguistische Arbeiten 394, Max Niemeyer Verlag, Tübingen/FRG.

Reape, M. (1994), Domain union and word order variation in german, *in* J. Nerbonne, K. Netter & C. Pollard, eds, 'German in Head-Driven Phrase Structure Grammar', CSLI, Stanford/CA, pp. 151–197.

Tesnière, L. (1959), *Eléments de Syntaxe Structurale*, Klincksiek, Paris/FRA.