TAG and Topology

Problems and Proposals for German

1. Introduction

This paper proposes yet another formalism in the TAG family with the purpose of capturing German word order phenomena and yet another linearization system of dependency grammars based on the topological model of the German sentence.

The originality may be seen in the fact that the proposed formalism accomplishes both these tasks at the same time.

2. Phrase structure and its shortcomings

Classical phrase structure tries to collapse syntactic and ordering information. However, this conception of the syntax of language is erroneous because it supposes that word order is always an immediate reflection of the syntactic hierarchy and that any deviation from this constitutes a problem, denoted by terms with negative undertones like *scrambling*[1].

Modern linguistic frameworks propose a double structure consisting at least of the basic syntactic structure (valency, functor-argument structure, f-structure, deep structure – we will use the term syntactic dependency) and a linear structure (surface structure, c-structure, phrase structure, precedence rules, …); some frameworks like LFG put this duality at the basis of their system, some others, like HPSG unify the different structures (e.g. SYNSEM and DTRS) in one sign.

While the functional subcategorization give rise to few discussions, the different phrase structures proposed for linearization constitute the bone of syntactic contention. One reason seems to be that surface phrase structure attempts to represent as much dependency information as possible, even though there is a large consensus on the impossibility of phrase structure to represent dependency in a satisfying manner. All these approaches appear to be caught in the transformational thinking which supposes two closely related structures: an already ordered[2] deep structure transforms after some movements into and a surface structure which still carries functional information. For English this attempt can go quite far, for languages with case however, the word order serves mainly to represent other communicative goals than functional recovery. Since Evers (1975) proposed a quite flat phrase structure for German, the follow-ups didn't look any better from a functional point of view, because they all fail to include full functional information in the phrase structure tree.

3. German sentence topology

For German, an equally important point against using phrase structure is that it fails to include the topology of the sentence.

The classical analysis of German sentence structure (Bech 1955) divides the sentence in a fixed sequence of fields, in which the syntactic elements are placed. We denote by *domain* a sequence of fields. The *main domain* of a declarative sentence consists of [Vorfeld(VF), left bracket('['), Mittelfeld(MF), right bracket(']') Nachfeld(NF)]. We call the fields VF, MF, and NF *major fields*.

Kathol 1995 proposes a formalization of the topological structure in HPSG (refining work of Reape 1994). He shows that this structure is independent of phrase structure and essential for linearizing German. However, based on the HPSG framework, he still needs to keep phrase structure for combining signs. In a sense, he keeps three levels of description: The domain structure (DOM) giving the linearization, the phrase structure tree (DTRS), representing how the structure has been build, and the dependency graph (encoded in SYNSEM), corresponding to the subcategorization.

Recent works in dependency grammar have tried to link directly the dependency structure to the placement of the words in different fields[3]. The topological structure then controls word order directly, skipping the constituent structure underlying HPSG. See Bröker 1998 for a lexicalized description of very simple phenomena based on modal logic, Duchier and Debusman 2001 for a description in constraint programming, and Gerdes and Kahane 2001 for a description of a topological hierarchy seen as a sophisticated syntactic module of Meaning Text Theory[4] (Mel'cuk 1987).

This paper is an attempt to lexicalize this latter approach, although we cannot describe it here in greater detail.

Essential to this analysis is the choice a verbal complement has when it is placed into the topological structure: It always goes into the right bracket of a domain, but this domain can be the existing domain of its verbal governor or, under some restriction, it can be a new domain it creates in a major field of its governor or in a higher domain containing the domain of its governor. If it creates a new domain, the verb behaves just like any non-verbal complement of the verb.

We take as an example the sentence (1), a variation of Rambow 1994's main example[5].

(1) Dem Lehrer zu lesen versprochen hat das Buch niemand.
The teacher to read promised has the book nobody.
Nobody has promised to the teacher to read the book.

A corresponding dependency tree is shown in Figure 1:[6]

[1] First used by Ross (1967).

[2] Although this ordering doesn't seem to be crucial for many of the proposed transformational analyses.

[3] The word order problem of many dependency approaches is somehow orthogonal to the problems of phrase structure grammars stated above: Lots of research in dependency grammars goes into the question of how to encode the linear ordering of the words into the dependency graph, as purely local rules (on the ordering of the dependents of a lexeme) are evidently not sufficient. See Lombardo and Lesmo 2000 for a short summary of the different attempts to define the 'right' degree of projectivity.

[4] MTT, as the name suggests, describes correspondences between different levels of linguistic description from a meaning representation to the surface string in the direction of text generation.

[5] *Reading the book* is a better example than *repairing the fridge*, because we avoid confusion with the benefactive dative (*ihm den Kühlschrank reparieren* vs. **ihm das Buch lesen)*.

[6] This approach needs a very 'surfacic' version of dependency: The controlled verb *zu lesen* does not control its subject *niemand*/*nobody*, and the subject belongs to the auxiliary or the past particple. As subject placement in German is identicical for auxiliaries, raising and control verbs, this approach needs a very 'surfacic' dependency and hence places the subject under the auxiliary.

We start from the root of the tree and place the finite verb in the left bracket. Its subject has to go in one of the major fields, and it goes into the Mittelfeld.
The past participle could join the right bracket of the existing domain, but, for the sake of this example, it decides differently and creates in the Vorfeld[7] a subordinate domain consisting only of Mittelfeld, right bracket, and Nachfeld. It takes the right bracket of this new domain. Its non-verbal dependent can go into one of the major fields of its governor's domain or of a domain containing its governor. Here, it will plump for the Mittelfeld of its governor's domain.
Its verbal dependent, the infinitive, could create a new domain in one of the major fields of its governor's domain or of a domain containing its governor. The other choice is to join the right bracket of its governor's domain. It decides to do the latter, and in this case, it has to go directly to the left of its governor.
Now it remains only to place the last complement, the book. It can again go in one of the major fields of its governor's domain, or of a domain containing its governor. It decides to join the higher Mittelfeld. We obtain the following final topological structure.

Vorfeld			*(*	*Mittelfeld*	*)*	*NF*
MF	*)*	*NF*				
Dem Lehrer	zu lesen versprochen		hat	das Buch niemand		
the teacher (dative)	to read promised		has	the book nobody (acc.) (nom.)		

The other possible surface orders of German can be obtained by making different choices[8]. These rules constitute the backbone of a description of dependency linearization in a topological model. For lack of space, we cannot give all the finer grained rules for licensing the creation of new domains and for a penalty system to prefer some choices of fields and some orders inside of one field (Oberfeldumstellung, order in the Mittelfeld). Equally, we cannot show that this approach gives elegant descriptions of complex phenomena like pied piping as well.

4. TAGs and their shortcomings

Even though we cannot justify the topological approach further, it is clear that TAG and its close relatives lack the generative power to engender the desired topological structure and the derivation tree in parallel.
To create a Tree Adjoining Grammar, one cuts the phrase structure tree of a sentence into lexicalized chunks that can be recombined by simple rules to the desired phrase structure, while noting down the steps taken yields a derivation tree, interpretable as a dependency. An analysis in Lexicalized TAGs (LTAG) consists of the string, the attached phrase structure (called derived tree), and the derivation tree. Becker et alii 91 respectively called giving the correct objects *weak*, *strong*, and *derivational generative power*. This notation disregards the fact that the derivation is part of the analysis that TAG attaches to a sentence, and should thus be considered as a part of the *strong generative power*. The above topological analysis makes us want to go a step further and include the linking of the topological structure and the dependency structure in the *strong generative power*. We could call this capacity of a lexicalized grammar to generate both structures compositionally (with corresponding substructures) the *descriptive strong generative power*.[9]

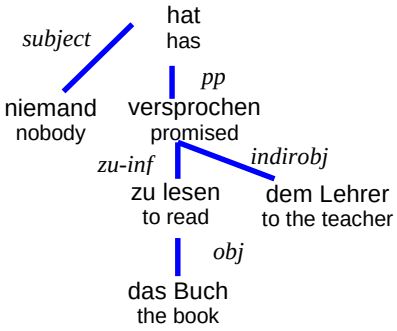


Figure 1 dependency tree

Our goal is thus to define a lexicalized tree grammar with the descriptive strong generative power for the relation between dependency and topology presented above. We will see that this grammar should also remedy some other flaws of TAG that we recall briefly:
- Becker et alii, 91, 92 show that TAG can not describe scrambling in German in a derivationally satisfying manner
- The derivation trees are lacking some of the desired dependencies (control verbs, relatives, etc.). The desired graph structure cannot be obtained (Candito and Kahane 1998a).
- Each adjunction adds an additional node level into the derived structure, often resulting in unmotivated derived trees. Different levels allow controlling adjunction between specific elements and they play in fact the role of linearization rules[10].
- . As elementary trees are ordered, we obtain a combinatorial explosion of elementary trees undistinguishable from lexical ambiguity and thus a high redundancy of information, in particular for freer word order languages like German.[11]

5. Giving German a TUG

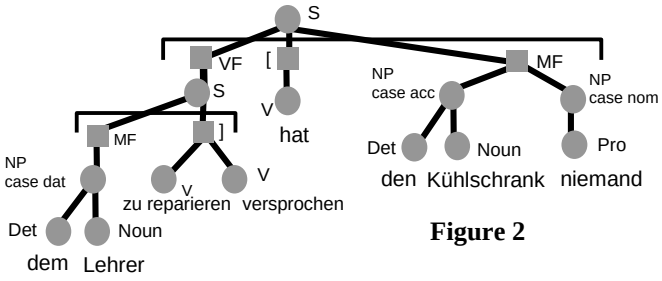


Figure 2

The hierarchy of domain boxes of Gerdes and Kahane 2001 can equally be represented as a tree (Figure 2): The lying square brackets represent domains, round atoms (nodes) represent regular phrase structure symbols, and square atoms field names.
This tree can also be seen as an ordered feature structure (Figure 3), which makes the following definition of tree

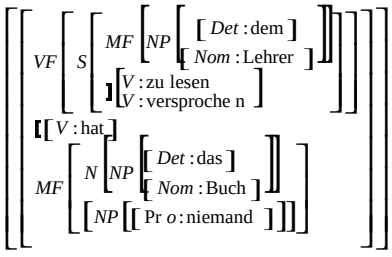


Figure 3

[7] This restriction applies only to past participles (pps), *Zu*-infinitives can be positioned in any of the major fields.
[8] The creation of an embedded domain has a prosodic and communicative value. Different topological structures can correspond to the same dependency structure and surface string, i.e. a sentence can be topological ambiguous. See Gerdes and Kahane 2001 for a more precise description of this phenomenon.
[9] This can be seen in the MTT framework as the complete correspondence between a syntactic level (surface syntactic dependency) and the topological hierarchy (used for linearizing the morphological level, i.e. the string of lexemes).
[10] This is true for DTGs (Rambow et alii 95) and GAGs (Candito and Kahane 1998b) as well
[11] One proposed solution, the metagrammar (Candito 1999), allows a solution of the practical aspects of grammar generation, but moves linguistic description out of the tree sets into the meta-

unification more intuitive.

Plucking the topological tree apart while keeping in mind the predicate-argument cooccurrence of the corresponding dependency structure, we introduce a new lexicalized tree grammar in the TAG family based on superposing and unifying tree structures. We call the formalism TUG (tree unification grammar):

Let *V* be an alphabet, let S∈ *V* be a distinguished letter.

An elementary tree is a tree with atoms as its nodes. Atoms can be full ■ ● (i.e. gray, the distinction of square and round atoms is only for better readability), empty ○, or optional (dotted lines). Atoms have a name out of *V* and a simple (non-embedded) feature structure. Features of empty nodes can also be *functional features*, to be defined below.

Leave atoms can have a lexicalization. The edges can be full or dotted and supplied with right or left sister adjunction exclusion. [12]

Additionally, an atom can control a domain bracket, specifying an ordered list of atoms it can exclusively dominate.[13]

Domain brackets unify in a specific way: Let $D=[D_1,D_2]$, D_1 and D_2 being sublists of D, $E=[E_1, E_2]$, E_1 and E_2 being sublists of E, and $D_2=E_1\neq\varnothing$. Then the domains D and E unify to $D\cup E=[D_1,D_2,E_2]$.[14]

The superposition procedure implied that the connection between the topological level, constructed out of the elementary trees, and the dependency has to be encoded specifically.[15] For this reason, we introduce a special feature, called *functional feature*, marked with an arrow and a name of the dependence link to construct: Let ↓*f* be a *simple functional feature* of atom X of the elementary tree E_1 with lexeme L_1. Then, an atom Y of elementary tree E_2 with corresponding lexeme L_2 unifies with X iff the lexeme L_1 is joined to a set containing L_2 by a dependence link with name *f*. Let ↓*g*↓*h* be a c*omplex functional feature* of atom X of the elementary tree E_1 with lexeme L_1. Then, an atom Y of elementary tree E_2 with corresponding lexeme L_2 unifies with α iff the lexeme L_1 is joined to a set containing L_2 by a dependence link with name *g* followed by a dependence with name *h*.

Unification of elementary trees is possible iff node names, node features, and domain brackets unify. Examples will help to clarify the definitions:

The trees α and the tree β (= α) unify, for instance, in three ways: The result can be identical to α or give the tree γ (in both orders of unification). α and δ can unify in only two ways, giving ε and ζ, as their bottom nodes have different names and can not be superposed.

The tree η, having a domain restriction, allows unification only with α (yielding η), as the domain restriction says that X has a unique daughter of name Y. Unification with γ, ε and ζ, fails. However, unification with θ is possible, as the two domains can unify, yielding θ ([Y] ∪[Y,Z]=[Y,Z]).[16]

Another simpler restriction consists of right and left sister exclusion: the node Y of ι cannot have a left sister, and ι and δ unify only to give κ. λ does not allow sisters in any direction, and it differs from η in the sense that it does not describe a domain and can never unify to form a larger domain.

Say the lexeme corresponding to the tree μ is also μ, and the lexeme of ν is ν. Then, the superposition of atom Y of the tree ν with the atom Y of the tree μ implies the existence of a link named *f* between μ and ν in the dependency graph of the construction, as in graph A.

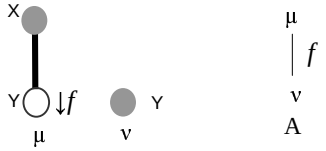


Figure 4: functional features

A lexical entry of a TUG consists of a lexeme and a finite set of elementary (topological field) trees, with at least one of them lexicalized. A domain bracket stores a list of the lexemes corresponding to the lexical entries that constructed it.

A topological derivation of sentence *P* is complete iff[17]

- a topological field structure under a node *S* can be created over *P*, using up all involved elementary trees
- all empty atoms are filled
- all solid (full or empty) atoms are connected by solid lines.
- the functional features relate the structure to a connected dependency tree *D*.[18]
- all lexemes of domain members have to be direct or indirect dependents of one of the lexemes that constructed the domain.

6. A toy TUG

We'll present a simple TUG with the lexical entries of sentence (1). The formalism allows more freedom in the construction of the dependency graph, and we can suppose as corresponding dependency the graph of Figure 8, which does not give a subject link to the auxiliary and constructs the correct 'double' control of the subject by the matrix verb and its infinitive. Moreover, the combination of noun and determiner in 'bubbles' goes back to Tesnière's *nuclei*. It avoids many practical and theoretical problems of a hierarchical representation (DP vs. NP, head ellipsis, …)[19]

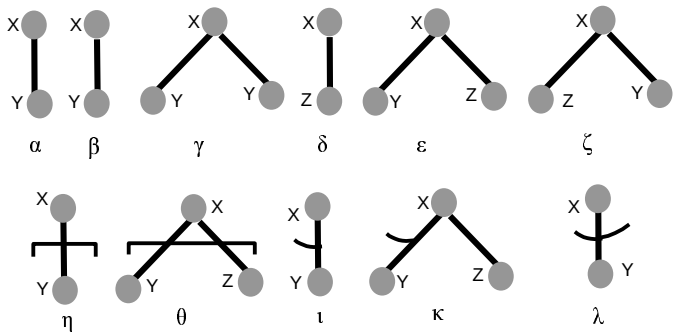


Figure 5

grammar, reducing the elementary trees to some algorithmic side product.

[12] This is a slightly simplified version for sentences without complementizers. Edges could also have barriers with precise permissivity rules, if we wanted to add CPs, relatives, and wh-movement.

[13] All adjunctions to such a node have to pass through one of the existing field atoms.

[14] This definition is less ad hoc if one thinks of it graphically as the superposition of two lists that have to have a part in common to be glued together.

[15] The link has to be more sophisticated than in TAG, where we just note at which node one tree was adjoined or substituted into the other.

[16] Note that domain union is not monotonic, as θ now unifies with ε. This could be a problem for the computational complexity.

[17] or equivalently: *a topological linearization of the dependency tree D is complete iff…*

[18] The correspondence between topological field structure and dependency is somewhat asymmetrical as we only have a well-formedness condition on the topological side. It is easy to make both sides of the correspondence generative by adding to each lexical entry its subcategorization frame and require the frame to be satisfied.

[19] A dependency structure augmented by bubbles can solve problems of ambiguity representation (Gerdes and Lopez 2000) and of word order in case of extraction (Kahane 2000)

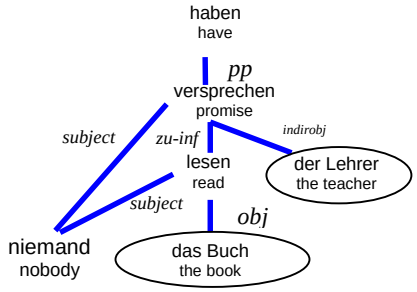
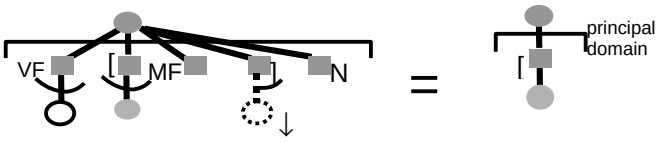
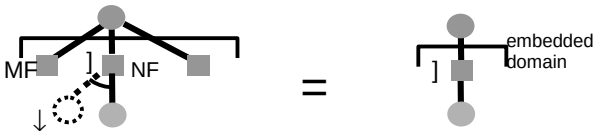


Figure 8: A better dependency graph

First, we simplify the notation for the appearing domains. The principal domain, introduced by a finite V2 verb, has the following form: The Vorfeld is obligatory (it has to be filled at the end of the analysis), and it has exactly one daughter. The left bracket, which will introduce this domain, will have one daughter, too. The Mittelfeld and the Nachfeld have no restrictions, and the right bracket can have none (optional node) or many daughters, but the rightmost daughter has to be any kind of dependent of the node that introduces this domain (denoted by a functional feature without a specified name). For this complete principal domain, we just write a simple domain bracket as at the right hand side of the equation.



We do the same for the embedded domain: It consists only of a Mittelfeld, a right bracket, and a Nachfeld. If there is left neighbor of the lexicalized node, it has to be its dependent[20]. We do not write the type of domain if confusion is not possible.



With the definition of domain unification above, a principal domain and an embedded domain can unify (to a principal domain).

Now we can show the first lexical entry: The entry for *hat* consists of 3 trees.

The auxiliary verb plans already the placement of its dependents, the past participle (pp) and the subject. The pp has to go into the right bracket, the NP in any principal field (xF=VF, MF, or NF). If the elementary tree for the pp creates its own domain, it can only do it in the VF (indicated by the optional node[21]). A *zu*-infinitive, does not have this restriction and it can create its domain in any principal field.

In the corresponding dependency structure, the lexeme of *haben* has a link labeled *pp* with the pp that will fill the empty V-atom (simple functional feature). The lexeme of this participle will have a link labeled *subj* with the set of all NP nodes that will fill the empty NP-atom (complex functional feature), i.e. the NP will be the subject of the pp. The subject has to stay in the principal domain.

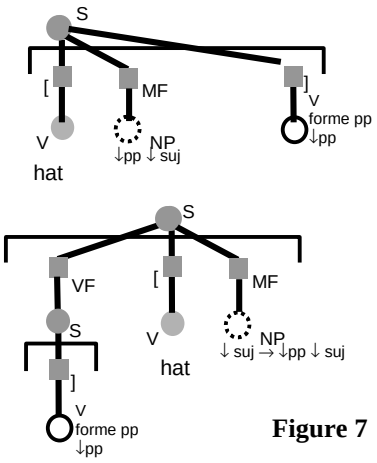


Figure 7

The entry for *versprochen*: Note that the subject of this control verb will also create a subject-link for its infinitive. The dotted line of the subject indicates that this edge has to be created by another elementary tree (the one that introduces the subject, i.e. the entry of *hat*).

The entry for *zu lesen*: The infinitive just takes its object and creates a dependency link *obj* with it.

The (strong) determiner can be a noun phrase by itself, contrarily to the noun, which needs a determiner. In a more complete version of the grammar, the internal structure of the NP is described by a domain structure as well, but this is beyond the scope of this article.

Let us now see how these lexical entries combine exactly to all possible surface structures while giving the same dependency:

In the entry of the auxiliary *hat*, the lexicalized tree and the tree for the pp can unify in two ways, one being the domain union, the other the creation of the new domain (1st and 2nd tree of Figure 7). The subject can go into any principal field of the principal domain.

Only the last choice of Figure 7 will lead us to the analysis of sentence 1. Now the lexicalized branch of *versprochen* has to join the open atom waiting for a pp, creating a *pp* link between the lexemes *haben* and *versprechen*. The branch for the infinitive in its entry can unify with this last tree in three basically different manners: it can join its governors domain (first tree), it can open a new domain in its governor's domain (second tree) and it can open a new domain in a domain containing its governor's domain (third tree of Figure 9).

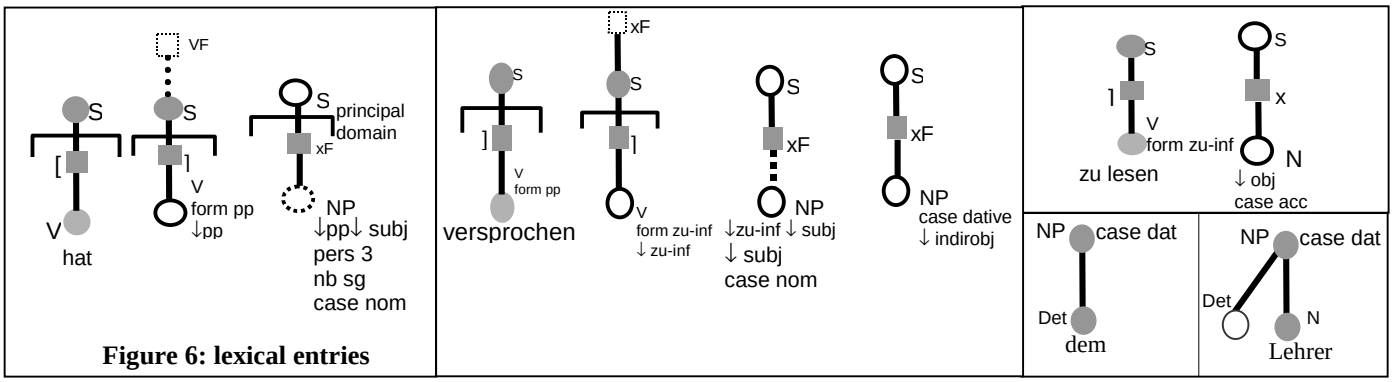


Figure 6: lexical entries

[20] For this presentation we simplify the grammar and cannot treat the Oberfeldumstellung in the right bracket.

[21] It is important to note that the optionality does not mean that the elementary tree can be used with or without the optional node, but that the optional node is not a part of the final structure if some solid node has not superposed it.

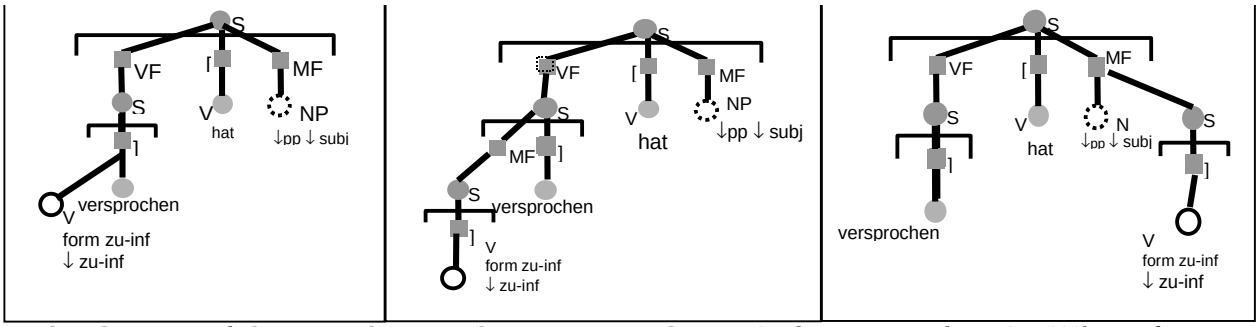


Figure 9: Placing the infinitive

The placement of the noun phrase in the open principal domains is controlled by the restriction stated above that a domain bracket can only host its direct or indirect dependents. If for instance *zu lesen* opens its own domain, the dative object *dem Lehrer* of *versprochen* cannot join this domain, as it is not a dependent of *zu lesen*.

When plugging in the elementary trees of the NPs, the determiner and the noun both fill in the empty atom, and the dependency graph will treat them as one set.

7. Conclusion and outlook

We see how a unique lexical entry allows for all the topological structures and thus describes very elegantly all the possible word orders for a given dependency.

This formalism allows us to derive in parallel the topological structure and a correct dependency graph. We have not shown that this is equally possible for object control and subjectless verbs. We also left aside field internal ordering rules (Mittelfeld, Oberfeldumstellung, …). These additional constraints on the final order are directly related to a topological structure, as they affect the fields as a whole, nearly independently of subcategorization (see for example Lenerz 1977).

An important question remaining concerns the computational aspect of the new formalism. For understanding its complexity, it may be useful to comprehend the procedure of superposing different elementary trees as a simple unification at any level of depth in an ordered feature structure. The additional domain union procedure, however, makes it difficult to determine an upper limit for the algorithm. This is ongoing work.

It is nonetheless important for our understanding of natural language to start with the pure description of the observed phenomena and to start only then with the formalization. Many formalisms, like TAGs, came the other way: They were simple and nice mathematical descriptions that were thought of as useful for natural language. The result is often a system functioning well for basic phenomena, but difficult to improve substantially as it is lacking descriptive linguistic power.

Some References

Abeillé Anne, 1991: *Une LTAG pour le français*. Ph.D. thesis. Univ. Paris 7.

Bech Gunnar, 1955, *Studien über das deutsche Verbum infinitum*, 2nd edition 1983, Linguistische Arbeiten, Nr. 139, Niemeyer, Tübingen.

Becker, T., A.K. Joshi, and O. Rambow 1991: "Long distance scrambling and tree adjoining grammars" in *Proceedings of ACL-Europe.*

Becker T., Niv M., and O. Rambow, 1992: *The Derivational Generative Power of Formal Systems or Scrambling is Beyond LCFRS.* IRCS Report 92-38, IRCS, University of Pennsylvania.

Bröker, N., 1998: "Separating Surface Order and Syntactic Relations in a Dependency Grammar". In *COLING-ACL 98*.

Candito M.-H., Kahane S. 1998a. `Can the TAG derivation tree represent a semantic graph? An answer in the light of Meaning-Text Theory.' *Proc. Fourth Workshop on Tree-Adjoining Grammars and Related Frameworks (TAG+4).*

Candito M.-H., Kahane S. 1998b. "Defining DTG derivations to get semantic graphs", *Proc. Fourth Workshop on Tree-Adjoining Grammars and Related Frameworks (TAG+4).*

Marie-Hélène Candito. 1999. *Structuration d'une grammaire LTAG : Application au français et à l'italien*. Ph.D. thesis, University of Paris 7.

Duchier Denys, Debusmann Ralph 2001: *Topological Dependency Trees: A Constraint-based Account of Linear Precedence*. Submitted, http://www.ps.uni-sb.de/~duchier/papers.html

Evers Arnoldus, 1975, *The transformational cycle in Dutch and German*. PhD thesis, University of Utrecht.

Gerdes K., Lopez P., 2000: "Shared Semantic Representations for LTAG", *Proceedings of the ATALA Workshop on Representation and Treatment of Syntactic Ambiguity in Natural Language Processing*. Paris, January 2000

Gerdes Kim, Kahane Sylvain, 2001: "Description of German syntax based on a topological hierarchy", *Proceedings of Seventh Germanic Linguistics Annual Conference*, GLAC7, Banff.

Kathol Andreas, 1995, *Linearization-based German Syntax*, PhD thesis, Ohio State University.

Lenerz J. 1977: *Zur Abfolge nominaler Satzglieder im Deutschen*, Tübingen, Narr.

Lombardo V. and Lesmo L. 2000: "A formal theory of dependency syntax with empty units", in Kahane S.(ed.): *Traitement Automatique des Langues.: Les grammaires de dépendance*, T.A.L., 41.1.

Mel'cuk Igor, Nicolas Pertsov, 1987: Surface syntax of English – A Formal Model within the Meaning-Text Framework, Amsterdam: Benjamins.

Netter, Klaus, 1996: *Functional Categories in an HPSG for German*, PhD thesis, Saarland University.

Pollard C.J., Sag I., 1994: Head driven phrase structure grammar, Studies in Contemporary Linguistics, Chicago, London: Unversity of Chicago Press.

Rambow Owen, 1994: *Formal and Computational Aspects of Natural Language Syntax*, Institute For Research in Cognitive Science, PhD thesis, University of Pennsylvania, Philadelphia,.

Rambow O., Vijay-Shanker, K. and Weir D., 1995. "D-Tree Grammars", Proceedings ACL 1995.

Rambow, Owen and Aravind Joshi, 1997. "A Formal Look at Dependency Grammars and Phrase-Structure Grammars, with Special Consideration of Word-Order Phenomena". In Wanner, Leo, editor, *Current Issues in Meaning-Text Theory*. Pinter, London.

Reape, M. 1994: « Domain Union and Word Order Variation in German », dans J. Nerbonne et al. (éds.), *German in Head-Driven Phrase Structure Grammar*, CSLI Lecture Notes, N° 46, Stanford,.

Ross John Robert. 1967: *Constraints on Variables in Syntax*, MIT, Cambridge, Mass.: Ph.D. Dissertation.

Shieber S. and Schabes Y. 1994, "An alternative conception of tree-adjoining derivation", in Computational Linguistics, vol. 20,n° 1, pp. 91-124.

Tesnière Lucien, 1959: Eléments de syntaxe structurale, Paris, Kliencksieck.