A quantitative model of word order and movement in English, Dutch and German complement constructions

KARIN HARBUSCH
Computer Science Department
University of Koblenz-Landau
PB 201602, 56016 Koblenz/DE
harbusch@uni-koblenz.de

GERARD KEMPEN
Cognitive Psychology Unit
Leiden University, and
Max Planck Institute, Nijmegen/NL
kempen@fsw.leidenuniv.nl

Abstract *We present a quantitative model of word order and movement constraints that enables a simple and uniform treatment of a seemingly heterogeneous collection of linear order phenomena in English, Dutch and German complement constructions (Wh-extraction, clause union, extraposition, verb clustering, particle movement, etc.). Underlying the scheme are central assumptions of the psycholinguistically motivated Performance Grammar (PG). Here we describe this formalism in declarative terms based on typed feature unification. PG allows a homogenous treatment of both the within- and between-language variations of the ordering phenomena under discussion, which reduce to different settings of a small number of quantitative parameters.*

1. Introduction

We propose a quantitative model for expressing word order and movement constraints that enables a simple and uniform treatment of a heterogeneous collection of linear ordering phenomena in English, Dutch and German complement structures. Underlying the scheme are central tenets of the psycholinguistically motivated Performance Grammar (PG) formalism, in particular the assumption that linear order is realized at a late stage of the grammatical encoding process. The model is described here in declarative terms based on typed feature unification. We show that both the within- and between-language variations of the ordering phenomena under scrutiny reduce to differences between a few numerical parameters.

The paper is organized as follows. In Section 2, we sketch PG's hierarchical structures. Section 3, the kernel of the paper, describes the linearization and movement model. In Section 4, we turn to central word order phenomena in the three target languages. Section 5, finally, contains some conclusions.

2. Hierarchical structure in PG

PG's hierarchical structures consist of unordered trees ('mobiles') composed out of elementary building blocks called *lexical frames*. These are 3-tiered mobiles assembled from branches called *segments*. The top layer of a frame consists of a single *phrasal node* (the 'root'; e.g. Sentence, Noun Phrase, ADJectival Phrase, Prepositional Phrase), which is connected to one or more *functional nodes* in the second layer (e.g., SUBJect, HeaD). At most one exemplar of a functional node is allowed in the same frame. Every functional node dominates exactly one phrasal node ('foot') in the third layer, except for HD which immediately dominates a lexical (part-of-speech) node. Each lexical frame is 'anchored' to exactly one lexical item: a *lemma* (printed below the lexical node serving as the frame's HeaD). A lexical frame encodes the word category (part of speech), subcategorization features, and morphological diacritics (person, gender, case, etc.) of its lexical anchor (cf. the elementary trees of Tree Adjoining Grammar (TAG; e.g. Joshi & Schabes, 1997).

Associated with every categorial node (i.e., lexical or phrasal node) is a *feature matrix*, which includes two types of features: *agreement features* (not to be discussed here; see Kempen & Harbusch, forthcoming) and *topological features*. The latter play a central role in the linear ordering mechanism. *Typed feature unification* of topological features takes place whenever a phrasal foot node of a lexical frame is replaced (substituted for) by a lexical frame. *Substitution* is PG's sole composition operation. Substitution involves unification of the feature matrices that are associated with the substituted phrasal foot node and the root node of the substituting lexical frame. Substitution gives rise to the derivation tree of a well-formed syntactic structure iff the phrasal foot node of all obligatory segments of each lexical frame successfully

unifies with the root of another frame. The tree in Figure 1 is well-formed because the MODifier segments are not obligatory.

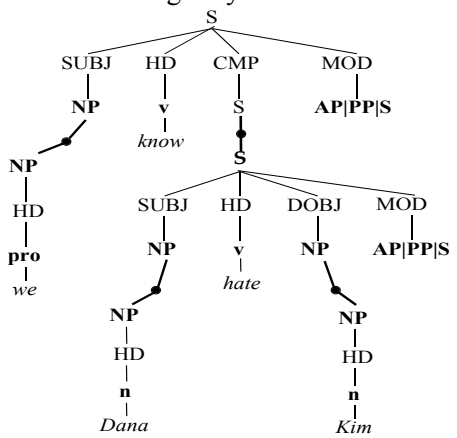


Figure 1. Simplified lexical frames underlying the sentences *We know Dana hates Kim* and *Kim we know Dana hates* (example from Sag & Wasow,1999). Order of branches is arbitrary. Filled circles denote substitution. (The feature matrices unified as part of the substitution operations are not shown.)

3. Linear structure in PG

The above-mentioned topological features are associated with the phrasal root nodes of lexical frames. Their value is a feature matrix specifying a 'topology', that is, a one-dimensional array of left-to-right *slots*. In this paper we will only be concerned with topological features associated with S-nodes. They serve to assign a left-to-right order to the segments (branches) of verb frames (i.e. lexical frames specifying the major constituents of clauses). On the basis of empirical-linguistic arguments (which we cannot discuss here), we propose that S-topologies of English, Dutch and German contain exactly nine slots:

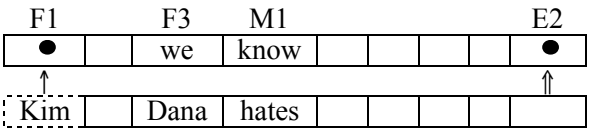
E	F1	F2	F3	M1	M2	M3	M4	E1	E2
D/G	F1	M1	M2	M3	M4	M5	M6	E1	E2

The slots labeled F*i* make up the Forefield (from Ger. *Vorfeld*); the M*j* slots belong to the Midfield (*Mittelfeld*); the E*k*'s define the Endfield (*Nachfeld;* terms adapted from traditional German grammar; cf. Kathol, 2000). Table 1 illustrates which clause constituents select which slot as their 'landing site'. Notice, in particular, that the placement conditions refer not only to the grammatical function fulfilled by a constituent but also to its shape. For instance, while the Direct Object takes M3 as its default landing site, it selects F1 if it is a Wh-phrase or carries focus, and M2 if it is a personal pronoun (*it*). In terms of Figure 1, if *Kim* carries focus, it may occupy slot F1 of the topology associated with the complement clause headed by *hate*.

Table 1. Examples of topology slot fillers (English). MODifier constituents are not shown. Precedence between constituents landing in the same slot is marked by "<".

Slot	Filler
F1	*Declarative main clause*: Topic, Focus (one constituent only) *Interrogative main clause*: Wh-constituent. *Complement clause*: Wh-constituent (including CoMPlementizeR *whether/if*)
F2	*Complement clause*: CoMPLementizeR *that*
F3	Subject (iff non-Wh)
M1	Pre-INFin. *to* < HeaD (oblig.) < PaRTicle
M2	Direct OBJect (iff personal pronoun) *Interrogative main clause*: SUBJect (iff non-Wh); SUBJ < DOBJ
M3	Indirect OBJect < Direct OBJect (non-Wh)
M4	PaRTicle
E1	Non-finite CoMPlement of 'Verb Raiser'
E2	Non-finite CoMP of 'VP Extraposition' verb Finite CoMPlement clause

How is the Direct Object NP *Kim* 'extracted' from the subordinate clause and 'moved' into the main clause? Movement of phrases between clauses is due to *lateral topology sharing*. If a sentence contains more than one verb, each of the verb frames concerned instantiates its own topology. This applies to verbs of any type, whether main, auxiliary or copula. In such cases, the topologies are allowed to *share* identically labeled lateral (i.e. left- and/or right-peripheral) slots, conditionally upon several restrictions to be explained shortly. *After two slots have been shared, they are no longer distinguishable; in fact, they are the same object.* In the example of Figure 1, the embedded topology shares its F1 slot with the F1 slot of the matrix clause. This is indicated by the dashed borders of the bottom F1 slot:



In sentence generation, the overt surface order of a sentence is determined by a *Read-out module* that traverses the hierarchy of topologies in left-to-right, depth-first manner. Any lexical item it 'sees' in a slot, is appended to the output string. E.g., *Kim* is seen while the Reader scans the matrix topology rather than during its traversal of the embedded to-

pology. See Figure 2 for the ordered tree corresponding to *Kim we know Dana hates*[1].

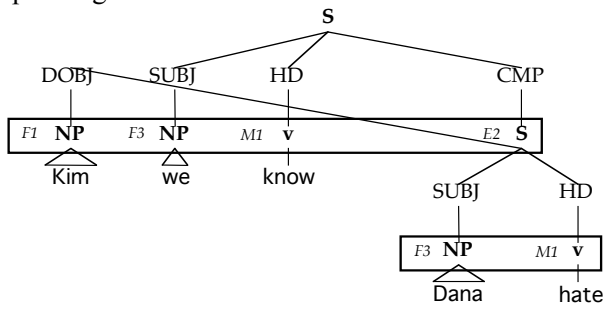


Figure 2. Fronting of Direct Object NP *Kim* due to promotion (cf. Figure 1). Rectangles represent (part of) the topologies associated with the verb frames.

The number of lateral slots an embedded topology shares with its upstairs neighbor is determined by the parameters *LS* (left-peripherally shared area) and *RS* (right-hand share). The two laterally shared areas are separated by a non-shared central area. The latter includes at least the slot occupied by the HeaD of the lexical frame (i.e., the verb) and usually additional slots. The language-specific parameters *LS* and *RS* are defined in the lexical entries of complement-taking verbs, and dictate how (part of) the feature structure associated with the foot of S-CMP-S segments gets instantiated. For instance, the lexical entry for *know* (Figure 1) states that *LS*=1 if the complement clause is finite and declarative. This causes the two S-nodes of the CoMPlement segment to share one left-peripheral slot, i.e. F1. If the complement happens to be interrogative (as in *We know who Dana hates*), *LS*=0, implying that the F1 slots do not share their contents and *who* cannot 'escape' from its clause.

In the remainder of this Section we present a rule system for lateral topology that is couched in a typed feature logic and uses HPSG terminology. The system deals with a broad variety of movement phenomena in English, Dutch and German.

We define a clausal topology as a list of slot types serving as the value of the topology ("TPL") feature associated with S-nodes:

S [TPL ⟨F1t,F2t,F3t,M1t,M2t,M3t,M4t,E1t,E2t⟩]

for English, and

S [TPL ⟨F1t,M1t,M2t,M3t,M4t,M5t,M6t,E1t,E2t⟩]

for Dutch and German. Slot types are defined as attributes that take as value a non-branching list of lemmas or phrases (e.g. SUBJect-NP, CoMPlement-S or HeaD-v). They are initialized with the value *empty list*, denoted by "⟨⟩" (e.g., [^{F1t}F1 ⟨⟩]).

Lists of segments can be combined by the *append* operation, represented by the symbol "⊕". The expression "L1⊕L2" represents the list composed of the members of L1 followed by the members of L2. We assume that L2 is non-empty. If L1 is the empty list, "L1⊕L2" evaluates to L2. Slot types may impose constraints on the *cardinality* (number of members) of the list serving as its value. Cardinality constraints are expressed as subscripts of the value list. E.g., the subscript "c=1" in [^{F1t}F1 ⟨⟩$_{c=1}$] states that the list serving as F1's value should contain exactly one member. Cardinality constraints are checked after all constituents that need a place have been appended.

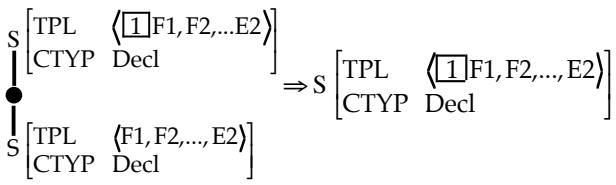
Depending on the values of sharing parameters *LS* and *RS*, the list can be divided into a left area (comprising zero or more slot types), the central area (which includes at least one slot for the HeaD verb), and the right area (possibly empty). Topology sharing is licensed exclusively to the lateral areas. *LS* and *RS* are set to zero by default; this applies to the topologies of main clauses and adverbial subclauses. The root S of a complement clause obtains its sharing parameter values from the foot of the S-CMP-S segment belonging to the lexical frame of its governing verb. For example, the lexical entry for *know* states that the complement of this verb should be instantiated with *LS*=1 if the clause type (CTYP) of the complement is declarative. This causes the first member of the topologies associated with the S-nodes to receive a coreference tag (indicated by boxed numbers):

$$\begin{array}{l} \text{S} \quad \left[\text{TPL} \ \left\langle \boxed{1}\text{F1, F2,..., E2} \right\rangle\right] \\ \text{CMP} \\ \quad\ \text{S} \quad \left[\begin{array}{ll} \text{TPL} & \left\langle \boxed{1}\text{F1,F2,...E2} \right\rangle \\ \text{CTYP} & \text{Decl} \end{array}\right] \end{array}$$

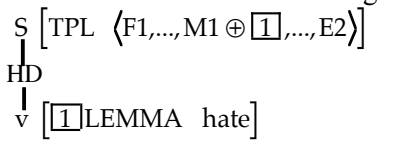
If, as in the example of Figure 1, *know*'s complement is indeed declarative, the foot of the complement segment can successfully unify with the root of the *hate* frame. As a consequence, the F1 slot of the complement clause is the same object as the F1 slot of the main clause, and any fillers will seem to have moved up one level in the clause hierarchy:

[1]The value of a TPL feature may be a disjunctive set of alternative topologies rather than a single topology. See the CMP-S node of Figure 3 for an example.

As for syntactic parsing, in Harbusch & Kempen (2000) we describe a modified ID/LP parser that can compute all alternative hierarchical PG structures licensed by an input string. We show that such a parser can fill the slots of the topologies associated with any such structure in polynomial time.



Filling a slot also involves coreference tags. For example, the HeaDs of English verb frames obtain their position in the local topology by looking up the slot associated with the coreference tag:



The information associated with the foot node of the HeaD segment will now be appended to the current content, if any, of slot M1. The same mechanism serves to allocate the finite complement clause (or rather its root S-node) to slot E2 of the matrix clause:

Other clause constituents receive their landing site (cf. Table 1) in a similar manner. Figure 2 depicts the configuration after Fronting of NP *Kim*.

Figure 3 below includes a paraphrase where the focus on *Kim* is stressed prosodically rather than by Fronting. This is indicated by the disjunctive set carrying the tag [4]. In sentence generation, the Read-out module selects one alternative, presumably in response to pragmatic an other context factors. In parsing mode, one or the alternatives is ruled out because it does not match word order in the input string.

The formalism defined so far yields unordered hierarchical structures. However, the values of the TPL features enable the derivation of ordered output strings of lexical items. As indicated above in connection with Figure 2, we assume that this task can be delegated to a simple *Read-out* module that traverses the clause hierarchy in a depth-first manner and processes the topologies from left to right[2]. If a slot is empty, the Reader jumps to the next slot. If a slot contains a lexical item, it is appended to the current output string and tagged as already processed. It follows that, if a slot happens to be shared with a lower topology, its contents are only processed at the higher clause level, i.e., undergo promotion.

4. Linearization of complement clauses in English, Dutch and German

The PG formalism developed above provides a simple quantitative linearization method capturing both within-clause and between-clause phenomena. The assignment of constituents to topology slots (including, e.g., scrambling in Dutch and German) has been dealt with in Kempen & Harbusch (in press; forthcoming). In the present paper we focus on promotion in complement constructions — a domain where the three target languages exhibit rather dissimilar ordering patterns. We highlight the fact that PG enables highly similar treatments of them, differing only with respect to the settings of some quantitative parameters.

The movement (promotion) phenomena at issue here depend primarily on the values assigned to sharing parameters *LS* and *RS* in five different types of complement clauses. These settings are shown Table 2. They are imported from the lexicon and control the instantiation of the TPL feature of the root S-node of the complement. We begin with some illustrations from English.

Table 2. Size of the left- and right-peripheral shared topology areas (*LS* and *RS*) in diverse complement constructions.

Clause type	English	Dutch	German
Interrogative	*LS*=0 *RS*=0	*LS*=0 *RS*=1	*LS*=0 *RS*=1
Declarative & Finite	*LS*=1 *RS*=0	*LS*=1 *RS*=1	*LS*=1 *RS*=1
Decl. & Non-Finite, VP Extraposition	*LS*=3 *RS*=0	*LS*=1 *RS*=1	*LS*=1 *RS*=1
Decl. & Non-Finite, Verb Raising	*LS*=3 *RS*=0	*LS*=4:6 *RS*=1	*LS*=5 *RS*=1
Decl. & Non-Finite, Third Construction	n.a.	*LS*=1:6 *RS*=1	*LS*=1:6 *RS*=1

The non-finite complements of *do* and *have* in sentence (1) below are both declarative. (Cf. the paraphrase "For which person *x* is it the case that I have to call *x*", which highlights the scope of *who*.) It follows that *LS*=3 in both complements. Notice that *do* is treated as a 'Verb Raiser', *have* (in *have to*) as a VP Extraposition verb.

[2] A slot may contain more than one phrase (e.g., Direct and Indirect OBJect in slot M3; cf. Table 1). We assume they have been ordered as part of the *append* operation, according to the sorting rule associated with the slot.

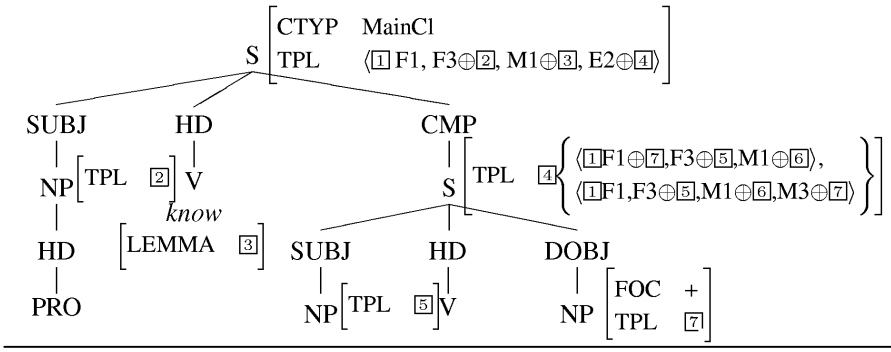
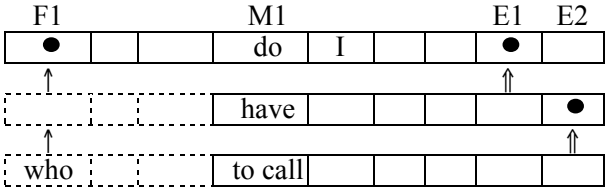


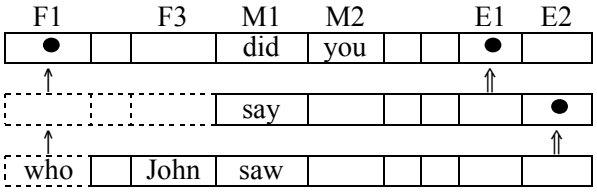
Figure 3. Analysis of *Kim we know Dana hates* (cf. Figure 1) and *We know Dana hates Kim*. The versions correspond to different options of the topology value associated with the CoMPlement (indicated within curly brackets). Empty slots are not shown in the TPL features.

(1) *Who do I have to call?*

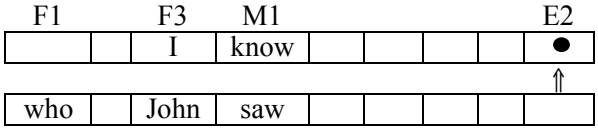


In example (2), the lower clause is finite and declarative —cf. the paraphrase "For which person *x* is it the case that you said that John saw *x*". The scope of *who* exceeds its 'own' clause and includes the matrix clause. In (3), on the other hand, the scope of the interrogative pronoun does not include the main clause ("I know for which person *x* it is the case that John saw *x*"). Therefore, the complement is interrogative and cannot share its F1 slot with that of the main clause.

(2) *Who did you say John saw?*



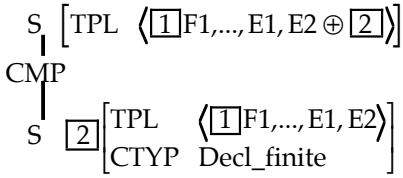
(3*) I know who John saw*



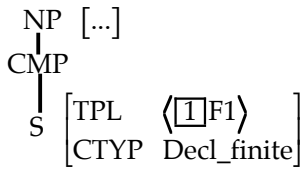
The system predicts 'island effects' as in (4).

(4) a. *Who did you claim that you saw?*
b. **Who did you make the claim that you saw?*

The lexical frame of the verb *claim* includes an S-CMP-S segment identical to that of *know* above (repeated here for convenience):



The feature matrices of root and foot nodes of this segment both specify a TPL feature referencing the slot F1. This enables insertion of coreference tag 2 and thus promotion of the filler of slot F1. However, the complement segment of the noun *claim* is rooting in an NP node, which cannot have a TPL feature with type F1t.



So, tag 1 is meaningless here, ruling out promotion in (4b).

Turning now to Dutch, we first refer to Table 3, which specifies some important landing sites for major clause constituents. Because of the similarity with German, we combine the two languages. First, we illustrate question formation.

Dutch interrogative main clauses feature Subject-Verb inversion without the equivalent of *do*-insertion (cf. 5).

(5) a. M1 *Zag* M2 *je* M3 *dat?*
saw you that
'Did you see that?'
b. F1 *Wie* M1 *zag* M3 *dat?*
who saw that
'Who saw that?'
c. F1 *Wat* M1 *zagen* M2 *ze?*
'What did they see?'

Because the complement in (6) is interrogative, the sharing rule in Table 2 prohibits left-peripheral sharing *(LS*=0).

(6) *Zij vroeg of ik hem kende*
She asked whether I him knew
'She asked whether I knew him'

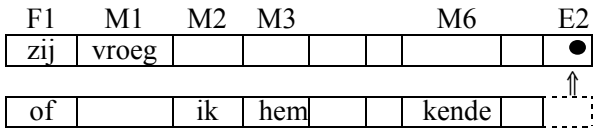
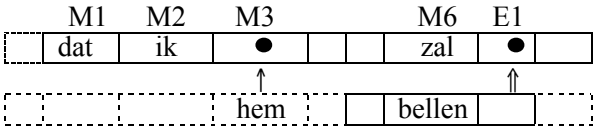


Table 3. Examples of topology slot fillers (Dutch and German). Precedence between constituents landing in the same slot is marked by "<".

Slot	Filler
F1	*Declarative main clause*: SUBJect, Topic or Focus (one constituent only) *Interrogative main clause*: Wh-constituent, including Du. *of* and Ger. *ob* 'whether' *Complement clause*: Wh-constituent
M1	*Main clause:* HeaD verb *Complement clause*: CoMPLementizeR *dat/om* (Du.)*, dass* (Ger.)
M2	Subject NP (iff non-Wh), Direct Object (iff personal pronoun)
M3	Indirect < Direct OBJect (iff non-Wh)
M4	PaRTicle (Du. only)
M5	Non-finite CoMPlement of Verb Raiser
M6	*Subordinate clause:* Du.: Pre-INFinitive *te* < HeaD verb Ger.: PaRTicle < Pre-INFinitive *zu* < HeaD
E1	Non-finite Complement of Verb Raiser (Du. only)
E2	Non-finite CoMP of VP Extraposition verb Finite Complement

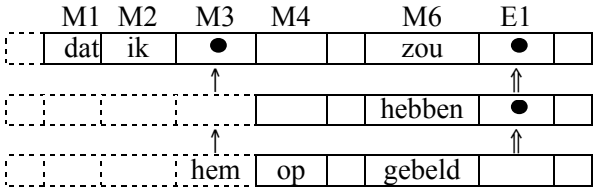
The subordinate clause in (7) features clause union, causing the auxiliary *zal* to intervene between the Direct *hem* the latter's governor *bellen*. The left-peripheral sharing area may vary between 4 and 6 slots (*LS*=4:6). Because *hem* lands in M3, i.e. in the shared area, it is promoted. The remainder of the lower topology, including the HeaD *bellen* itself, occupies E1 — one of the options of the complement of a Verb Raiser. The other option, with the complement in M5 (giving *bellen zal*) is also allowed.

(7) *...dat ik hem zal bellen*
that I him will phone
'...that I will phone him'



Sentence (8) illustrates the treatment of 'particle hopping'. The positions marked by "^" are grammatical alternatives to the particle (*op*) position mentioned in the example; no other positions are allowed. Given *LS*=4:6 for complements of Verb Raisers, it follows that *hem* is obligatorily promoted into the higher topology:

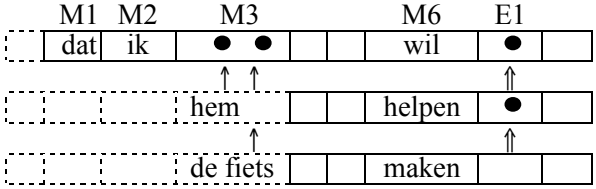
(8) *...dat ik hem ^ zou ^ hebben op gebeld*
that I him would have up called
'...that I would have called him up'



However, sharing of the fifth slot (M4) is optional. If this option is realized in the middle topology, the order *zou op hebben gebeld* ensues. If, in addition, the middle topology shares M4 with its governor, the string comes out as *op zou hebben gebeld*.

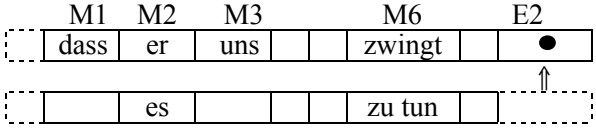
The treatment of cross-serial dependencies is exemplified in (9). In order to deal with this construction, we need to make an additional assumption about the order of constituents that land in the same slot but originate from different levels in the clause hierarchy. We stipulate that constituents from more deeply embedded clauses trail behind constituents belonging to higher clauses. This ordering can be determined locally within a slot if we equip every constituents in the hierarchy with a numerical 'clause depth' index (for instance, a Gorn number; Gorn, 1967). Given this index (not shown in the topology diagram accompanying (9)), the order *hem de fiets* results.

(9) *... dat ik hem de fiets wil helpen maken*
that I him the bike want-to help repair
'... that I want to help him to repair the bike'



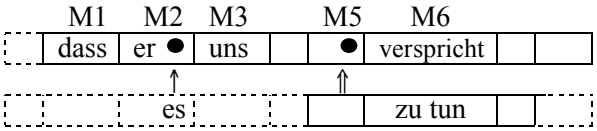
We now turn to German, concentrating on structures usually labeled "VP Extraposition" (10) and "Third Construction" (11).

(10) *... dass er uns zwingt es zu tun*
that he us forces it to do
'... that he forces us to do it'



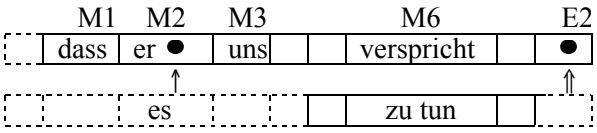
(11) a. *... dass er uns verspricht es zu tun*
that he us promises it to do
'... that he promises us to do it'

b. *... dass er es uns zu tun verspricht*

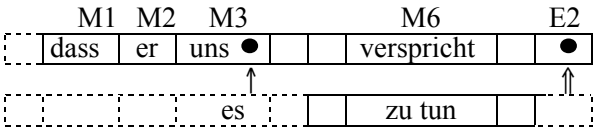


c. *...dass er uns es zu tun verspricht*

d. *...dass er es uns verspricht zu tun*



e. ? *...dass er uns es verspricht zu tun*



The verb *zwingen* allows its complement to share slot F1 only (*LS*=1). This prevents promotion of the Direct OBJect *es*. Third Construction verbs like *versprechen* allow a great deal of variation in the size of the left-peripherally shared topology area (*LS*=1:6), thereby licensing optional promotion of *es*. However, since *es* is a personal pronoun, it only takes M2 as its landing site (see Table 3). The latter constraint is violated in (11e).

5. Discussion

We have shown that the introduction of topologies with a fixed number of slots, in conjunction with cross-clause lateral topology sharing, enables a simple treatment of word order and movement (promotion) in complement structures of the three target languages. The considerable within- and between-language variation typical of these constructions could be analyzed as resulting from different settings of a small number of quantitative parameters, in particular the size of shared areas. We claim that our approach is conducive to theoretical parsimony (and, presumably, computational efficiency). For instance, HPSG-style treatments of Wh-movement and Clause Union typically invoke very different types of mechanisms (e.g., the SLASH or GAP feature for WH-movement, and argument composition for Clause Union; cf. Sag & Wasow, *o.c.,* and Kathol *o.c.*).

Elsewhere we have provided a more fine-grained discussion of our approach and its psycholinguistic motivation (Kempen & Harbusch, in press; forthcoming). Future study is needed to find out whether the PG approach generalizes to other languages.

Finally, we refer to the PG sentence generator for Dutch which was implemented by Camiel van Breugel. It covers the ordering phenomena described here and in Kempen & Harbusch (forthcoming) and runs under Java-enabled Internet browsers (www.liacs.nl/~cvbreuge/pgw). Vosse & Kempen (2000) describe a computational model of human syntactic parsing based on a PG-like formalism.

References

GORN, S. (1967). *Explicit Definition and Linguistic Dominoes. Systems and Computer Science.* Toronto: University of Toronto Press.

HARBUSCH, K. & KEMPEN, G. (2000). Complexity of linear order computation in Performance Grammar, TAG and HPSG. In: *Proceedings of Fifth International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5),* University of Paris 7, May 2000.

JOSHI, A.K. & SCHABES, Y. (1997). Tree Adjoining Grammars. In: Rozenberg, G. & Salomaa, A. (Eds.), *Handbook of formal languages (Vol. 3)*. Berlin: Springer.

KATHOL, A. (2000). *Linear Syntax.* New York: Oxford University Press.

KEMPEN, G. & HARBUSCH, K. (in press). Word order scrambling as a consequence of incremental sentence production. In: Haertl, H., Olsen, S. & Tappe, H. (Eds.), *The syntax-semantics interface: Linguistic structures and processes*. Berlin: De Gruyter.

KEMPEN, G. & HARBUSCH, K. (forthcoming). Dutch and German verb clusters in Performance Grammar. In: Seuren, P. & Kempen, G. (Eds.), *Verb clusters in Dutch and German.* Amsterdam: Benjamins.

SAG, I.A. & WASOW, T. (1999). *Syntactic theory: a formal introduction.* Stanford CA: CSLI Publications.

VOSSE, T. & KEMPEN, G. (2000). Syntactic structure assembly in human parsing: A computational model based on competitive inhibition and a lexicalist grammar. *Cognition*, *75*, 105-143.