Joan Bresnan,
Ronald M. Kaplan,
Stanley Peters,
Annie Zaenen

Cross-serial Dependencies in Dutch

1. Are Natural Languages Context Free?

Chomsky's argument that natural languages are not finite state languages puts a lower bound on the weak generative capacity of grammars for natural languages (Chomsky (1956)). Arguments based on weak generative capacity are useful in excluding classes of formal devices as characterizations of natural language, but they are not the only formal considerations by which this can be done. Generative grammars may also be excluded because they cannot assign the correct structural descriptions to the terminal strings of a language; in this case, the grammars are excluded on grounds of strong generative capacity. Thus, the deterministic subclasses of context-free grammars (Knuth (1965)) can be rejected because they cannot assign alternative phrase structures to represent natural language ambiguities.

A question of some interest is whether natural languages can be characterized by utilizing the full class of context-free grammars. Despite the early rejection of such grammars by transformational grammarians (Chomsky (1957), Postal (1964a)), recent work has shown that context-free grammars are powerful devices that can describe many complex properties of natural languages in a formally restricted but linguistically general way (Gazdar (1981; in press)). A convincing demonstration that natural language string sets are not context-free languages would indicate that these grammars are too restrictive to be capable in principle of even weakly characterizing natural language. Several attempts to establish this result have been offered in the literature (see Postal (1964b), Langendoen (1977), Huybregts (1976), and other references cited in Pullum and Gazdar (in press)). However, Pullum and Gazdar (in press) argue that all of these attempts suffer from either formal or empirical deficiencies. Thus, it remains possible that natural languages considered as string sets are in fact weakly generable by context-free grammars.

We are indebted to Oliver Gajek, Geoff Pullum, and Koos van der Wilt for commenting on earlier drafts of this article. This study is based on work supported in part by the National Science Foundation under Grant Nos. BNS-80-14730 to the Massachusetts Institute of Technology and BNS-76-20307 to the University of Texas at Austin, in part by the Cognitive and Instructional Sciences Group of the Xerox Palo Alto Research Center, and in part by the Center for Advanced Study in the Behavioral Sciences.

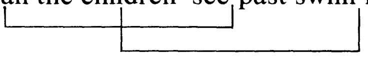
Linguistic Inquiry, Volume 13, Number 4, Fall 1982
0024-3892/82/040613-23 $02.50/0
©1982 by The Massachusetts Institute of Technology

There is another, arguably more interesting sense in which a natural language can be context free—namely, if there is a context-free grammar that assigns syntactically and semantically motivated structural descriptions to the strings of the language. If this is the case, we will say that the language is *strongly context free*. A language which is weakly context free need not be strongly context free. Even if the string set of the language is weakly generable by some context-free grammar, there may be no context-free grammar which assigns the correct set of structural descriptions to the language. We will show in this article that Dutch is just such a language, and thus, that natural languages in general are not strongly context free. This does not imply, however, that adequate natural language descriptions require the full power of transformational grammar: we also show that the troublesome Dutch constructions are strongly generated by a lexical-functional grammar (Kaplan and Bresnan (1982)).

2. An Invalid Lower Bound Argument Based on Dutch

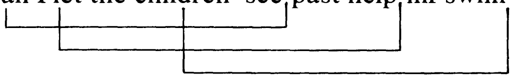
Huybregts (1976) has argued that Dutch cannot be a (weakly) context-free language because it contains an infinite set of grammatical sentences which have cross-serial dependencies of the form given in (1)–(3).

(1) . . . dat Jan de kinderen zag zwemmen
that Jan the children see-past swim-inf



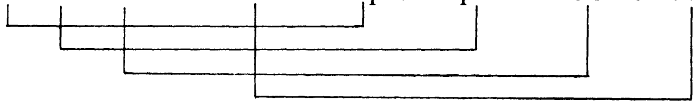
'. . . that Jan saw the children swim'

(2) . . . dat Jan Piet de kinderen zag helpen zwemmen
that Jan Piet the children see-past help-inf swim-inf



'. . . that Jan saw Piet help the children swim'

(3) . . . dat Jan Piet Marie de kinderen zag helpen laten zwemmen
that Jan Piet Marie the children see-past help-inf make-inf swim-inf



'. . . that Jan saw Piet help Marie make the children swim'

Arbitrarily many of these sentences can be formed simply by inserting into the string a noun phrase and a verb that is subcategorized for both a noun phrase and an infinitival complement without the complementizer *te*. The verb in first position is formally distinguished by its marking for tense and its person and number agreement with the first NP. The verb in last position is distinguished from the others by its subcategorization restrictions. Although there are only a finite number of insertable verbs, they can be repeated, as in (4).

(4) . . . dat de leraar Jan Marie de kinderen leerde laten leren
that the teacher Jan Marie the children teach-past make-inf teach-inf
zwemmen
swim-inf
'. . . that the teacher taught Jan to make Marie teach the children to swim'

While it is true that the formal language {ωω | ω ∈ V*}, whose strings exhibit arbitrarily deep cross-serial dependencies, is not context free if V contains at least two elements, the set of Dutch examples differs crucially from this language. For provided that the number of verbs matches the number of noun phrases, and provided that the agreement constraint between the first NP and the first verb is respected and the subcategorization restrictions between the final NPs and the final verb are satisfied, all permutations of the NPs within the NP sequence and all permutations of the verbs within the verb sequence produce grammatical sentences. These restrictions can be expressed by a context-free grammar, because even though the restrictions impose cross-serial dependencies, there are only finitely many of them (namely, two) to be encoded in the grammar. Thus, examples like the following are all grammatical.

(5) . . . dat Jan Marie Piet de kinderen zag helpen laten zwemmen
'. . . that Jan saw Marie help Piet make the children swim'
(6) . . . dat Jan Marie de kinderen Piet zag helpen laten zwemmen
'. . . that Jan saw Marie help the children make Piet swim'
(7) . . . dat Jan Marie de kinderen Piet zag laten helpen zwemmen
'. . . that Jan saw Marie make the children help Piet swim'

There are indeed infinitely many cross-serial associations between the NP arguments and their corresponding predicates, but these are not formally encoded in the string set of Dutch in any way.

As a result of these considerations, we can see that the following context-free grammar suffices to generate the string set of this class of Dutch examples.

(8) a.
$$\text{S} \rightarrow \underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\end{bmatrix}}{\text{NP}} \quad \underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\\ +n\end{bmatrix}}{\text{S}'} \quad \underset{[+n]}{\text{V}}$$

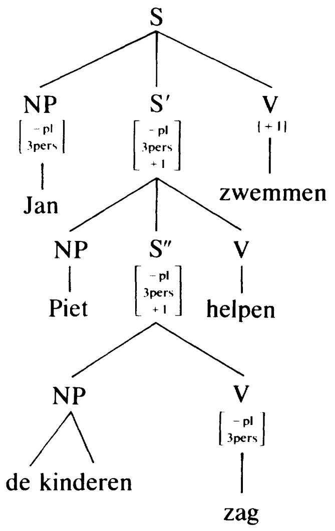
b.
$$\underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\\ +n\end{bmatrix}}{\text{S}'} \rightarrow \text{NP} \quad \underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\\ +n\end{bmatrix}}{\text{S}'} \quad \text{V}$$

c.
$$\underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\\ +n\end{bmatrix}}{\text{S}'} \rightarrow \text{NP} \quad \underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\\ +n\end{bmatrix}}{\text{S}''} \quad \text{V}$$

d.
$$\underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\\ +n\end{bmatrix}}{\text{S}''} \rightarrow \text{NP}^{n} \quad \underset{\begin{bmatrix}\alpha\text{pl}\\ \beta\text{pers}\end{bmatrix}}{\text{V}}$$

In (8), α, β, and n are abbreviatory devices that provide schemata for a finite set of context-free rules. α ranges over + and −; β ranges over 1, 2, 3; and $1 \leq n \leq u$, where u is the (finite) upper bound on the number of NPs that any Dutch verb can be subcategorized for.[1] For example, (8) generates example (2) in the way shown in (9). This grammar generates an artificially restricted (but infinite) proper subset of the relevant Dutch examples; see Pullum and Gazdar (in press) for discussion of how wider classes of examples can be described.

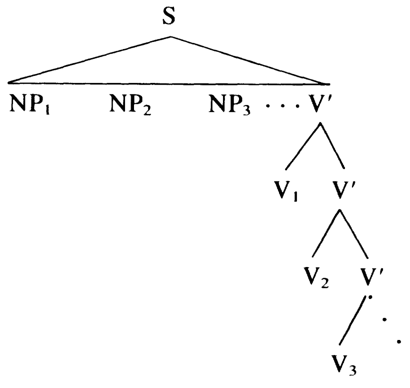
(9)



3. Evidence for the Correct Tree Structures

While the grammar of (8) weakly generates the cross-serial examples of Dutch, the constituent structures that it assigns are linguistically incorrect. Linguists have argued that the cross-serial constructions have the surface phrasal structure shown in (10) (Evers (1975)).

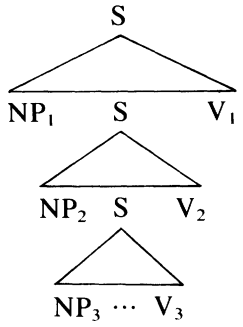
(10)



[1] For an exposition of such abbreviatory notations for context-free grammars, see Gazdar (in press).

Working within a transformational framework, Evers (1975) proposed that structures like (10) are derived from structures like (11) by verb-raising and tree-pruning operations.

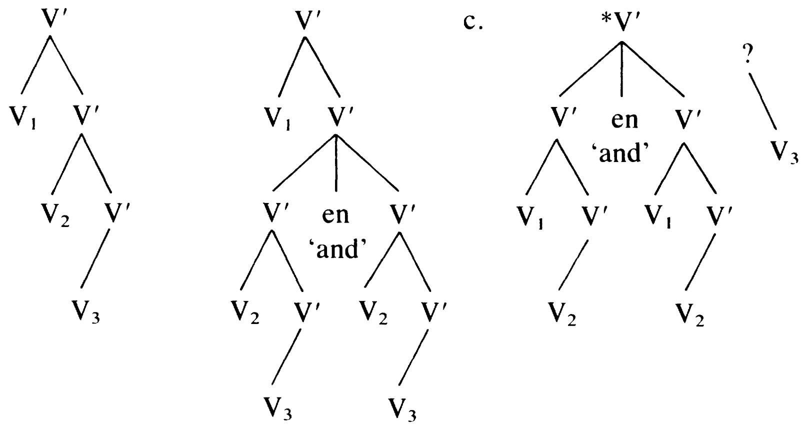
(11)



Specifically, the right-branching verbal group in (10) is produced by cyclically adjoining each verb or verb group to the higher verb on its right and then extraposing the former around the latter. The flat NP structure in (10) is produced by pruning the S nodes of the embedded clauses as their verbs are raised out of them.

There is good evidence for the right-branching verbal structure shown in (10). It is possible to conjoin single constituents in Dutch, but not in general nonconstituent sequences of categories. Hence, if the verbal group in cross-serial constructions has the constituent structure shown in (12a), the conjunction shown in (12b) should be well formed, whereas the one in (12c) should not.

(12) a.



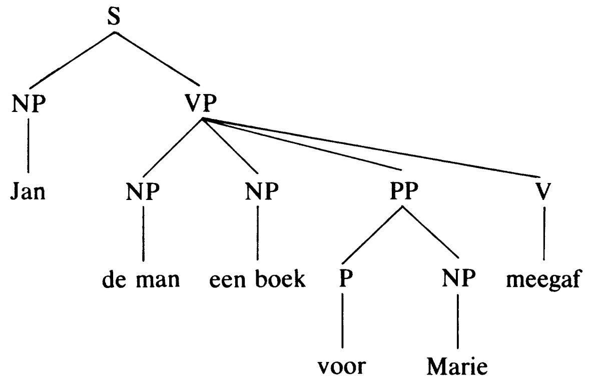
This accounts for the contrast between (13) and (14).

(13) . . . dat Jan Marie de kinderen zag leren zwemmen en helpen
that Jan Marie the children see-past teach-inf swim-inf and help-inf
lopen
run-inf
'. . . that Jan saw Marie teach the children to swim and help the children to run'

(14) *?. . . dat Jan Marie de kinderen zag leren en liet helpen
that Jan Marie the children see-past teach-inf and make-past help-inf
zwemmen
swim-inf
'. . . that Jan saw Marie teach the children to swim and made her help the children to swim'

Example (14) is marginally acceptable with comma intonation setting off *en liet helpen*; this probably arises from the marginal applicability of the Right Node Raising rule, which differs from ordinary conjunction in requiring special intonation and in allowing only a single node to the right of the conjoined elements. Further evidence for the right-branching verb cluster is given by Evers (1975).

In contrast, the flat NP structure proposed by Evers and illustrated in (10) does not seem to be correct. There is evidence that the sequence of NPs has more constituent structure than the diagram in (10) shows. In general, PPs can occur in any order with respect to their sister constituents in Dutch. Accordingly, given the structure in (15), the examples in (16) are predicted.

(15)


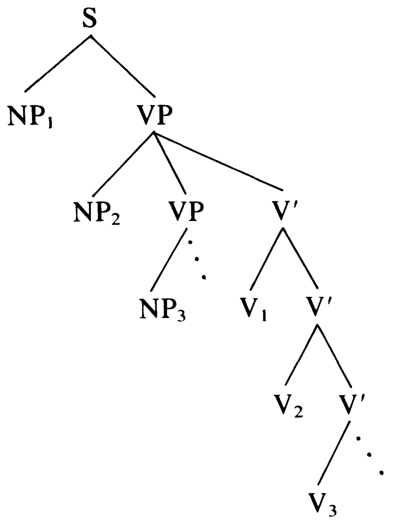
(16) a. . . . dat Jan de man een boek voor Marie meegaf
that Jan the man a book for Marie give-with-past
'. . . that Jan gave a book to the man (to take with him) for Marie'
b. . . . dat Jan de man voor Marie een boek meegaf
c. . . . dat Jan voor Marie de man een boek meegaf
d. *. . . dat voor Marie Jan de man een boek meegaf

The following examples indicate that in sentences exhibiting cross-serial dependencies only the last NP is a sister of the PP.

(17) a. . . . dat Jan Piet een boek op de tafel zag neerleggen
that Jan Piet a book on the table see-past put-down
'. . . that Jan saw Piet put a book down on the table'
b. . . . dat Jan Piet op de tafel een boek zag neerleggen
c. *. . . dat Jan op de tafel Piet een boek zag neerleggen
d. *. . . dat op de tafel Jan Piet een boek zag neerleggen

This can be explained under the assumption that the tree in (18), not the one in (10), is the correct form of constituent structure for cross-serial sentences.[2]

(18)



The type of constituent structure shown in (18) also explains a contrast in the conjoinability of material before the verb sequence:

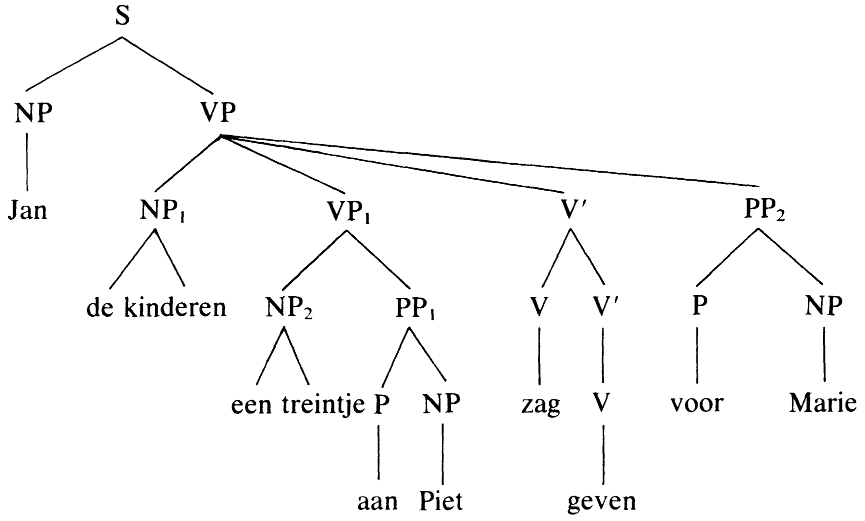
(19) . . . dat Jan de kinderen een treintje aan Piet en een pop aan Henk
that Jan the children a toy train to Piet and a doll to Henk
zag geven voor Marie
see-past give-inf for Marie
'. . . that Jan saw the children give a toy train to Piet and a doll to Henk for Marie'

(20) ??. . . dat Jan de meisjes een treintje aan Piet en de jongens een pop
that Jan the girls a toy train to Piet and the boys a doll
aan Henk zag geven voor Marie
to Henk see-past give-inf for Marie
'. . . that Jan saw the boys give a toy train to Piet and the girls give a doll to Henk for Marie'

As shown in (21), the sequence NP_2 PP_1 forms a constituent (VP_1), while the sequence NP_1 NP_2 PP_1 = NP_1 VP_1 does not form a constituent.

[2] This explanation was suggested by Ewan Klein (personal communication). Evers (1975) gives several arguments for flat NP structure, based on extraposition, clitic placement, quantifier hopping, and clause negation in Dutch, but all of his evidence is consistent with the weaker hypothesis that the NP sequence in cross-serial examples lacks S structure, as in (18).

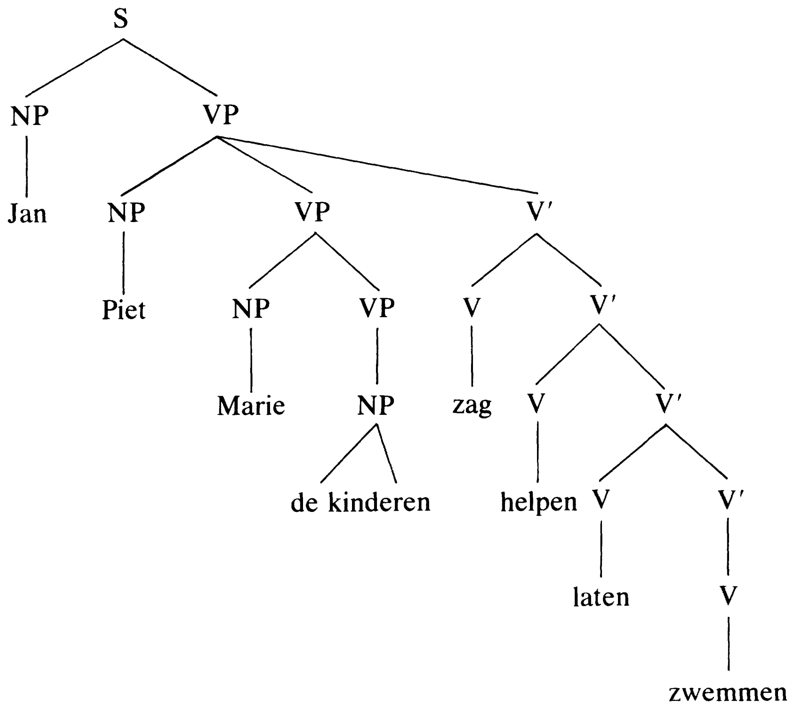
(21)



Hence, it should be possible to conjoin two NP_2 PP_1 sequences, as in (19), but not two NP_1 NP_2 PP_1 sequences, as in (20). The PP_2 has been included in these examples to exclude the possibility of deriving (20) by Right Node Raising of the V sequence, which must be final in the VP for that rule to apply.

In summary, the correct structural descriptions of these Dutch sentences can be characterized as follows. ((22) provides an example for reference.)

(22)



There is a right-branching complement VP structure which contains the objects and complements of the verbs but not the verbs themselves, and a sister right-branching verbal group that contains the verbs without their objects and complements. The subcategorization requirements of a particular verb on the right must be satisfied by the phrases at the corresponding level of embedding in the structure on the left. Failure to observe this restriction leads to ungrammatical examples like the following:

(23) *. . . dat Jan de leraar de kinderen zag helpen laten leren
that Jan the teacher the children see-past help-inf make-inf teach-inf
zwemmen
swim-inf

(24) *. . . dat Jan Piet Marie de leraar de kinderen zag leren zwemmen
that Jan Piet Marie the teacher the children see-past teach-inf swim-inf

These restrictions hold for Dutch examples of arbitrary depth.

Given this characterization of the correct tree structures and given the uncontroversial assumption that subcategorization restrictions are syntactic (on which see Grimshaw (1982a)), the question arises of how the subcategorization restrictions between the verbs and their complements are to be stated within a context-free grammar. One might think that some set of context-free feature propagation devices could do the job, but it turns out that this is not possible: as we show in the next section, there is no context-free grammar that can generate all and only the syntactically well-formed trees for Dutch sentences of this type.

4. Dutch Is Not Strongly Context Free

If Dutch cross-serial constructions are correctly described by trees of the form characterized in the preceding section, then there is no context-free grammar that can assign the correct structural descriptions to Dutch sentences. To establish this result, we will argue for a slightly stronger conclusion, namely, that the structural descriptions of Dutch do not constitute a set of trees recognizable by a finite state tree automaton. The fact that Dutch is not strongly generable by any context-free grammar then immediately follows by virtue of the theorem that the derivation trees of any context-free grammar constitute a recognizable set (Thatcher (1967)).

We use a pumping lemma on recognizable sets of trees to demonstrate that no such set can have the formal property isolated at the end of the last section, namely, that the trees contain two right-branching subtrees of matching heights. For every recognizable set of trees, there is a constant n such that any tree in the set having height greater than n can be partitioned into three parts t_1, t_2, t_3 where the height of the subtree t_2t_3 is less than n (see figure 1) and any tree formed by iterating the middle part t_2 as in figure 2 also belongs to the recognizable set (Thatcher (1973)).

Now let us assume that the trees of Dutch constitute a recognizable set, and let n be the appropriate constant for this set. Consider a tree of the form in (18) whose height

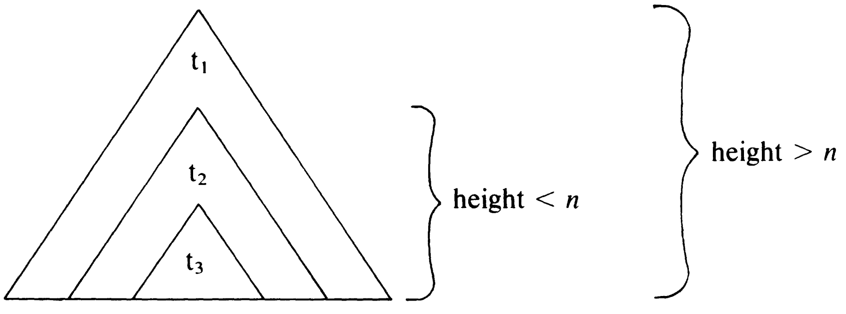


Figure 1

is greater than n. If we partition it into the parts t_1, t_2, t_3, part t_2 must either be in the VP branch of the tree, as shown in figure 3, or else it must be in the V′ branch, as shown in figure 4. In either case, iterating t_2 produces a tree in which the VP and V′ subtrees are not of corresponding heights. In general, such trees are not well-formed structural descriptions of Dutch, as the ungrammaticality of examples (23) and (24) illustrates. This contradicts the assumption that Dutch structural descriptions form a recognizable set of trees.

We have shown that there is no context-free grammar that can generate all and only the correct structural descriptions for Dutch. Thus, this natural language lies beyond

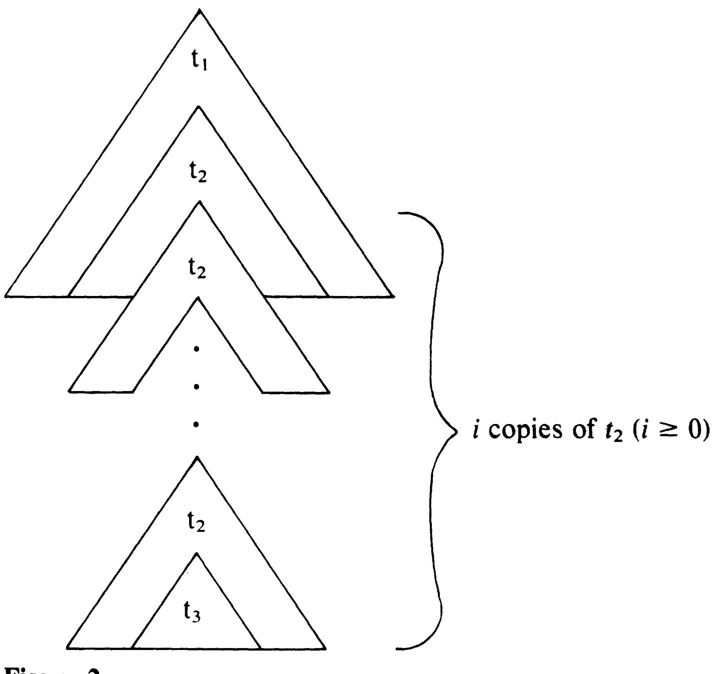


Figure 2

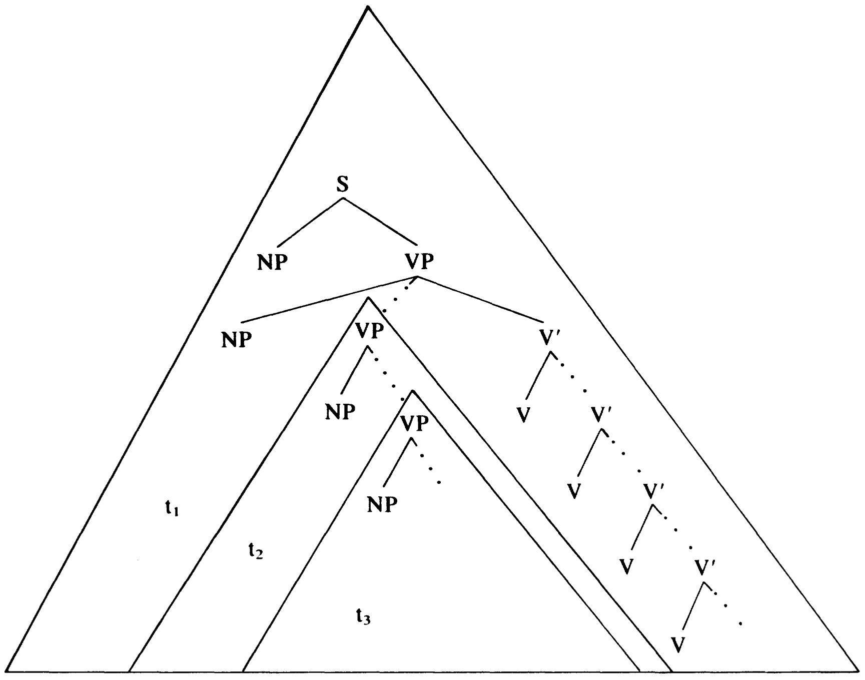


Figure 3

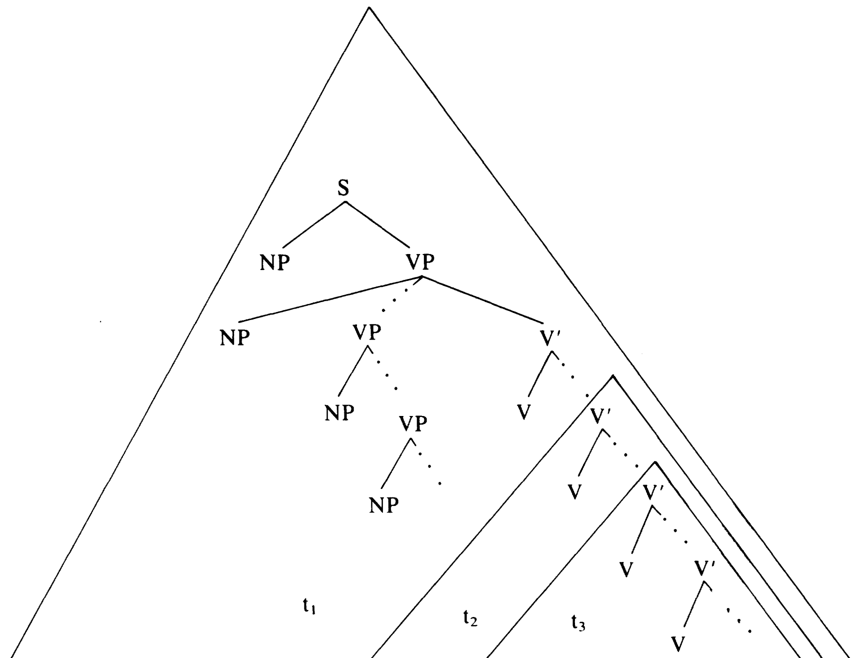


Figure 4

the strong generative capacity of context-free grammars. From this we can also conclude that Dutch cannot be strongly generated by a categorial grammar, because the structures that such grammars generate are included in the structures generated by context-free grammars (Bar-Hillel, Gaifman, and Shamir (1960)). This result also extends to Bach's (1979; 1980; 1981) generalization of categorial grammars if his right-wrap operation is sufficiently constrained.[3]

5. A Lexical-Functional Grammar Generates the Set of Correct Tree Structures

Whether or not context-free grammars can weakly generate the string sets of natural languages, they are not in general sufficient for generating the correct structural descriptions of natural language. This, however, is not an argument that transformational devices are necessary. In fact, we can show that the correct descriptions can be assigned to the class of Dutch cross-serial constructions by a more restrictive system than transformational grammars, the lexical-functional grammars (LFGs) of Kaplan and Bresnan (1982). We will limit our attention to the class of examples discussed in section 2, for these are sufficient to illustrate the essential formal properties of the LFG solution.

A lexical-functional grammar includes a set of context-free rules for generating the constituent structures ("c-structures") of sentences. These rules are annotated with functional schemata that combine with similar lexical schemata to determine the functional structures ("f-structures") corresponding to those c-structures. F-structures are hierarchical structures that formally represent the grammatical relations of sentences in terms of such universal functions as SUBJ(ect), OBJ(ect), and COMP(lement), abstracting away from language-particular differences in surface form. For a string to be grammatical, it must be assigned not only a well-formed c-structure according to the standard interpretation of context-free rules, but also an f-structure that satisfies the general well-formedness conditions of Uniqueness, Completeness, and Coherence (Kaplan and Bresnan (1982)). The Uniqueness Condition asserts that every grammatical function or feature must be assigned a single value. The Completeness and Coherence Conditions require that all and only the grammatical functions mentioned by a lexical predicate are local to that predicate in the f-structure. Together they guarantee that the subcategorization requirements of lexical entries are satisfied. Because of these three functional well-formedness conditions, the functional component of a lexical-functional grammar serves as a filter on the output of the c-structure rules, marking as ungrammatical strings that have otherwise valid c-structures.

The linguistically motivated c-structures for sentences with cross-serial dependencies have two parallel right-branching structures, as illustrated in (22). Trees of this sort can be generated by the following simple context-free grammar:

(25)
$$\begin{array}{lll} S & \rightarrow & NP\ \ VP \\ VP & \rightarrow & (NP)\ \ (VP)\ \ (V') \\ V' & \rightarrow & V\ \ (V') \end{array}$$

[3] One constraint that suffices, for example, is that whenever A is right-wrapped around B, the position where B is inserted in A's leftmost branch is a bounded distance from the bottom or top of that branch.

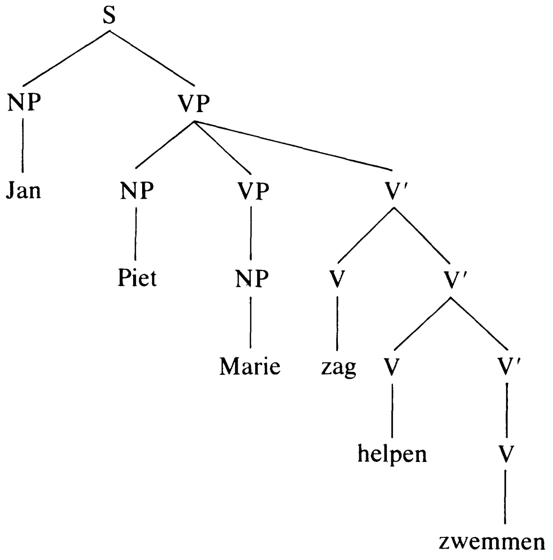
It is obvious that this grammar generates far more than just the set of correct Dutch trees, since the grammar does not express the dependency between the depth of the branching structures on the left and right. However, when the appropriate functional schemata are added to these rules, they determine for each c-structure a corresponding f-structure which does represent the dependency. The general well-formedness conditions on f-structures will eliminate those trees in which the depth of branching on the left and right is mismatched.

The f-structure for sentence (26), whose c-structure is shown in (27), is given in (28).

(26)	. . . dat	Jan	Piet	Marie	zag	helpen	zwemmen
	that	Jan	Piet	Marie	see-past	help-inf	swim-inf

'. . . that Jan saw Piet help Marie swim'

(27)



(28)
$$\left[\begin{array}{ll}
\text{SUBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`JAN'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{PRED} & \text{`SEE}\langle(\uparrow \text{SUBJ})(\uparrow \text{OBJ})(\uparrow \text{VCOMP})\rangle\text{'}\\
\text{TENSE} & \text{PAST}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`PIET'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{VCOMP} & \left[\begin{array}{ll}
\text{SUBJ} & \\
\text{PRED} & \text{`HELP}\langle(\uparrow \text{SUBJ})(\uparrow \text{OBJ})(\uparrow \text{VCOMP})\rangle\text{'}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`MARIE'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{VCOMP} & \left[\begin{array}{ll}\text{SUBJ} & \\ \text{PRED} & \text{`SWIM}\langle(\uparrow \text{SUBJ})\rangle\text{'}\end{array}\right]
\end{array}\right]
\end{array}\right]$$

As (28) shows, an f-structure is a set of ordered pairs, each consisting of the name of a grammatical function or feature and a *value* for that function or feature. For a grammatical feature such as TENSE, the values are drawn from finite sets of symbols over which that feature ranges (e.g. the symbols PAST and PRESENT for the TENSE feature, SG and PL for the number feature NUM, etc.). The value for a grammatical function like SUBJ or VCOMP is an embedded f-structure, a subsidiary set of functions and features. Thus, the value of the VCOMP (for verb complement) in the outermost brackets in (28) is itself an f-structure with internal functions SUBJ, OBJ, and VCOMP. PRED features have a third type of value, called a *semantic form*. This is a quoted expression containing the name of a semantic predicate and, in the case of a relational predicate, a specification of how the grammatical functions in the local f-structure are to be assigned to the predicate's logical arguments. The list in angle brackets after SEE, for example, indicates that SEE is a three-place semantic predicate whose first argument is filled by the SUBJ ('JAN'), whose second argument is the OBJ ('PIET'), and whose third argument is the hierarchical VCOMP. Thus, the predicate argument relations for the outermost clause of sentence (26) may be read directly from the outermost PRED and functions in (28):

(29) SEE(JAN, PIET, *Piet help Marie swim*)

In (28) the values of the embedded SUBJ functions are not fully spelled out; instead, there is a line linking each of those SUBJs to the value of the OBJ in the immediately enclosing f-structure. This linkage represents the fact that a *functional control* relation holds between the linked SUBJs and OBJs. In functional control, the linked functions have exactly the same value. Thus, the linkage between the OBJ of SEE and the SUBJ of HELP in this example indicates that PIET is understood as both the OBJ of SEE and the SUBJ of HELP, so that (30) is a more complete rendition of the predicate argument relations of this sentence:

(30) SEE(JAN, PIET, HELP(PIET, MARIE, SWIM(MARIE)))

Because of the identities represented by control linkages such as these, f-structures technically are acyclic directed graphs, not just simple hierarchies.[4]

F-structures are assigned to subconstituents of the c-structure by virtue of functional annotations associated with the context-free rules and lexical entries of a lexical-functional grammar; the f-structure that the grammar assigns to a sentence as a whole is taken to be the one assigned to its root S node. These annotations specify a node's f-structure in terms of its own lexical or grammatical features and its daughter's f-structures.[5] An illustration of the notation in which these functional specifications are expressed is given in (31), a partial lexical entry for the proper noun *Jan*:

[4] Halvorsen (1981) shows that this representation supports a model-theoretic semantic interpretation for control and quantification phenomena.

[5] As described by Kaplan and Bresnan (1982), f-structure specifications also come from more remote nodes in cases of long-distance dependencies (called *constituent control*). Specifications from these remote sources are not relevant to the current discussion.

(31) *Jan*: N (↑ PRED) = 'JAN'
(↑ NUM) = SG

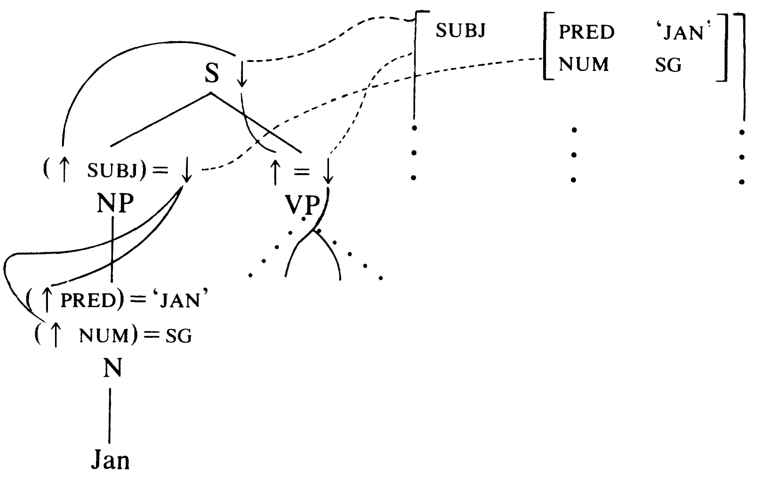
This entry lists N as the c-structure category of *Jan* and provides a set of equations that define feature values of the f-structures corresponding to any nodes headed by this word. A parenthetic expression of the form (f α) refers to the value of the α function or feature in the f-structure designated by f. Thus, the first equation in (31) asserts that the value of the PRED feature in the f-structure designated by ↑ is the semantic form 'JAN', and the second equation defines the value of the NUM feature of that f-structure to be SG. Note that these equations would both be true if the f-structure designated by ↑ were the value of the outermost SUBJ in (28), and thus that f-structure is a "solution" to this pair of equations. In general, the LFG machinery produces such equations from functional annotations throughout the c-structure, and the f-structure for a sentence is the solution to that set of simultaneous equations.

The rule in (32) is an annotated version of the context-free S rule in (25).

$$\text{(32)}\quad S \rightarrow \underset{(\uparrow \text{ SUBJ}) = \downarrow}{\text{NP}} \quad \underset{\uparrow = \downarrow}{\text{VP}}$$

The equation under the NP category, (↑ SUBJ) = ↓, asserts that the value of the SUBJ function of the f-structure designated by ↑ is the f-structure designated by the symbol ↓. When this rule is used to expand a node, ↑ and ↓ are taken to designate the f-structures associated with the S and NP nodes, respectively. Thus, in equations that are produced in expanding the NP, ↑ must also refer to the same f-structure that is referred to by ↓ in the SUBJ equation in the S rule. (33) associates these functional annotations with the nodes of the tree and shows an assignment of f-structures to the ↑ and ↓ symbols such that all of the equations are simultaneously satisfied.

(33)

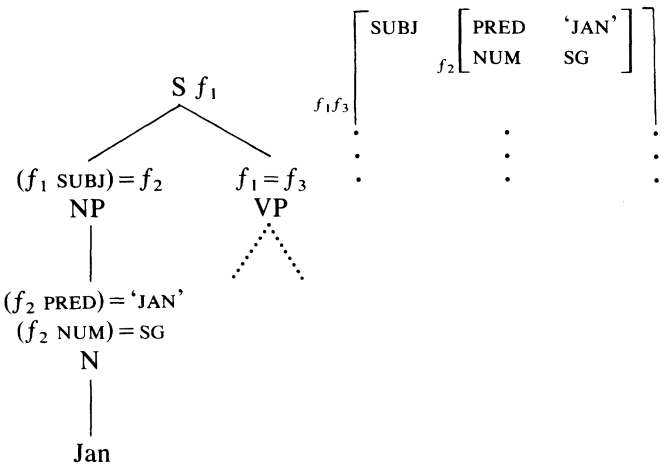


In the example, the solid curves connect symbols which must designate the same f-structure by virtue of the tree relations of the nodes with which they are associated.

The broken curves indicate the f-structure assigned to those related sets of $\downarrow$ and $\uparrow$ symbols. An additional $\downarrow$ appears at the root of the tree to stand for the f-structure assigned to the sentence as a whole. The equation $\uparrow = \downarrow$ indicates that the S and VP nodes have the same f-structure. Because of this, the equations will be satisfied only by an f-structure in which the functions and features of the VP's f-structure are merged with the functions and features of the S's, thus expressing the fact that the verb phrase is the head of the sentence.

The information represented graphically by the curves in (33) is expressed symbolically by means of the *instantiation procedure* described in Kaplan and Bresnan (1982). The equations on the tree in (34) are derived from those in (33) by replacing codesignating $\uparrow$ s and $\downarrow$ s with common indices f_1, f_2, etc. Kaplan and Bresnan (1982) show that it is decidable whether or not there exists an f-structure satisfying an instantiated set of equations (called a *functional description* or "f-description") and present an algorithm for actually synthesizing the f-structure that an f-description describes. The present discussion, however, depends only on the procedure for verifying that a given candidate f-structure is in fact a solution for a particular f-description.

(34)


S f_1
$(f_1 \text{ SUBJ}) = f_2$
NP
$f_1 = f_3$
VP
$(f_2 \text{ PRED}) = \text{'JAN'}$
$(f_2 \text{ NUM}) = \text{SG}$
N
Jan
$f_1 f_3$
SUBJ
f_2
PRED 'JAN'
NUM SG

The additional rules and lexical entries necessary to generate sentence (26) are given in (35) and (36).

(35)
$$\text{VP} \rightarrow \begin{pmatrix} \text{NP} \\ (\uparrow \text{OBJ}) = \downarrow \end{pmatrix} \begin{pmatrix} \text{VP} \\ (\uparrow \text{VCOMP}) = \downarrow \end{pmatrix} \begin{pmatrix} \text{V}' \\ \uparrow = \downarrow \end{pmatrix}$$

$$\text{V}' \rightarrow \text{V} \begin{pmatrix} \text{V}' \\ (\uparrow \text{VCOMP}) = \downarrow \end{pmatrix}$$

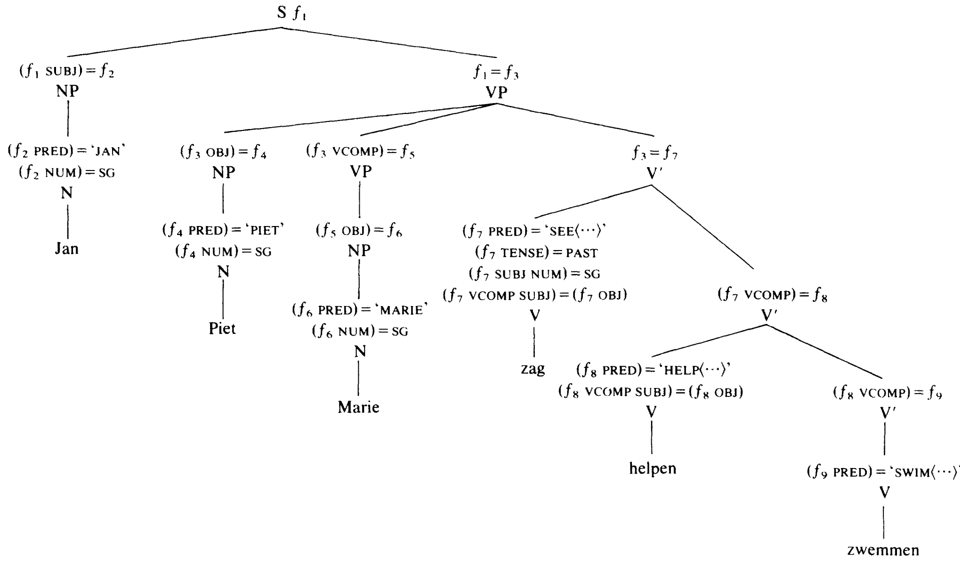
$$\text{NP} \rightarrow \text{N}$$

(36) *zag:* V $(\uparrow \text{PRED}) = \text{'SEE}\langle(\uparrow \text{SUBJ})\ (\uparrow \text{OBJ})\ (\uparrow \text{VCOMP})\rangle\text{'}$
$(\uparrow \text{TENSE}) = \text{PAST}$
$(\uparrow \text{SUBJ NUM}) = \text{SG}$
$(\uparrow \text{VCOMP SUBJ}) = (\uparrow \text{OBJ})$

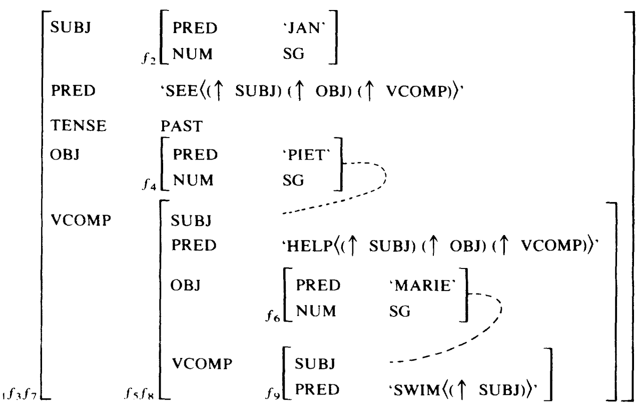
helpen:	V	(↑ PRED) = 'HELP⟨(↑ SUBJ) (↑ OBJ) (↑ VCOMP)⟩'
		(↑ VCOMP SUBJ) = (↑ OBJ)
zwemmen:	V	(↑ PRED) = 'SWIM⟨(↑ SUBJ)⟩'
Piet:	N	(↑ PRED) = 'PIET'
		(↑ NUM) = SG
Marie:	N	(↑ PRED) = 'MARIE'
		(↑ NUM) = SG

The lexical entries for *zag* and *helpen* include the functional control equations (↑ VCOMP SUBJ) = (↑ OBJ). These equations assert that the object of the verb is identified with its complement's subject. (37) shows the complete set of instantiated equations for sentence (26). The reader may verify that the f-structure satisfies all of the equations in this figure under the assignment of indices indicated in (38).

(37)



(38)



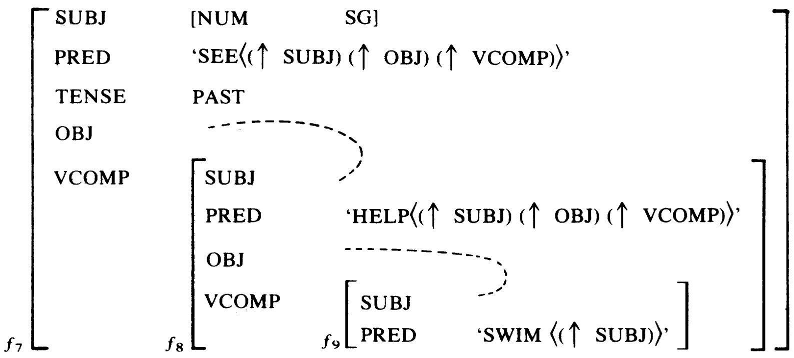
In (37) the specifications just on the left branches of the VP subtree characterize an f-structure containing only one embedded VCOMP with an OBJ but no PRED, as shown in (39).

(39)

$$f_3\left[\begin{array}{ll} \text{OBJ} & f_4\left[\begin{array}{ll} \text{PRED} & \text{`PIET'} \\ \text{NUM} & \text{SG} \end{array}\right] \\ \text{VCOMP} & f_5\left[\begin{array}{ll} \text{OBJ} & f_6\left[\begin{array}{ll} \text{PRED} & \text{`MARIE'} \\ \text{NUM} & \text{SG} \end{array}\right] \end{array}\right] \end{array}\right]$$

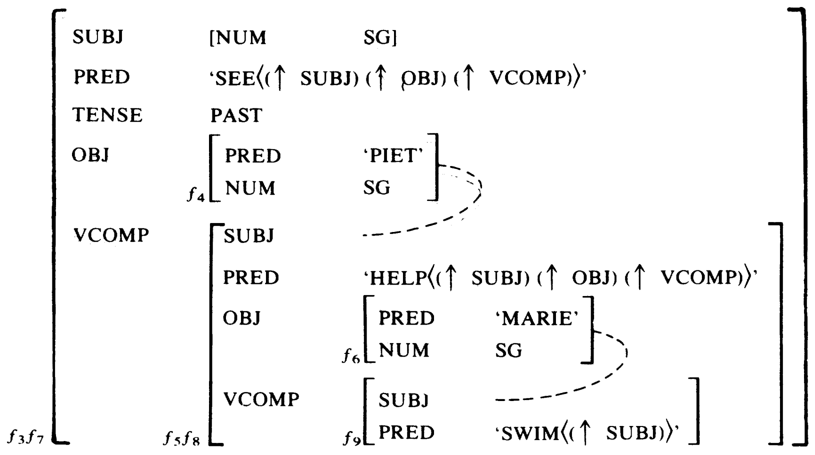
In contrast, specifications on the V′ subtree characterize a VCOMP hierarchy with PREDS and functional control relations between SUBJS and OBJS (40), but the internal features of these functions are not specified on that branch.

(40)



However, because the identity ↑ = ↓ on the topmost V′ node is instantiated as $f_3 = f_7$, the only f-structure that satisfies together all of the equations under the highest VP is one in which the information specified on the two branches is hierarchically merged, as shown in (41). The "merger" of the discontinuous functional specifications of (39) and (40) is a formal consequence of the Uniqueness Condition.

(41)



The requirement that subject–verb agreement hold between the first NP and the finite verb also follows from the Uniqueness Condition. A finite form of the verb such as *zag* specifies a value for the NUM feature of its SUBJ. This feature of the verb is propagated to the f-structure of the sentence by virtue of the fact that the verb is the head of the VP and the VP is the head of the sentence, as indicated by the $\uparrow = \downarrow$ identity annotated to the VP in rule (32). The Uniqueness Condition holds if the specifications defined on the verb and the specification derived from lexical material within the SUBJ NP both assign the same value, as is the case for sentence (26). However, example (42) is ungrammatical because the lexical entry for *zagen* specifies a plural number for its SUBJ, by means of the alternative equation ($\uparrow$ SUBJ NUM) = PL.

(42) *. . . dat Jan Piet Marie zagen helpen zwemmen

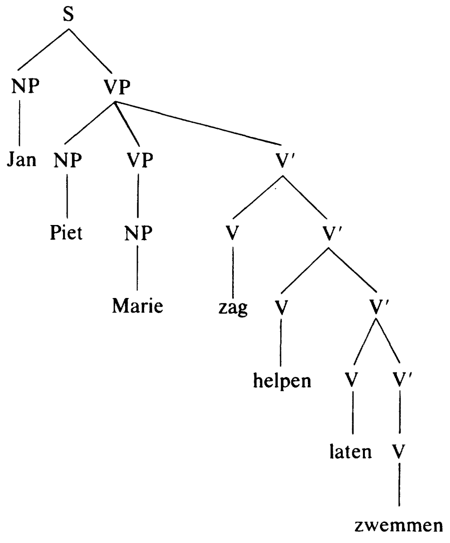
This is inconsistent with the SG specification contributed by the lexical entry for *Jan*.

We have seen how the grammar fragment and lexical entries above do in fact assign the f-structure (28) to sentence (26). This f-structure also satisfies the Completeness and Coherence Conditions: the functions subcategorized by each verb are in one-to-one correspondence with the functions found in its local f-structure. Now consider how these rules and annotations would apply for the string (43), which has an additional verb but not an additional NP.

(43) *. . . dat	Jan	Piet	Marie	zag	helpen	laten	zwemmen
that	Jan	Piet	Marie	see-past	help-inf	make-inf	swim-inf

As (44) shows, the V′ branch in the c-structure for this string contains an extra level, and the f-description associated with that branch specifies an extra VCOMP level with the PRED feature for *laten*. A corresponding level does not exist in the VP structure, so there are no specifications for the OBJ of that VCOMP.

(44)



In the resulting f-structure (45), the MAKE and SWIM PREDS refer to grammatical functions (OBJ and SUBJ) for which no values are specified; the string is ungrammatical because its f-structure violates the Completeness Condition.

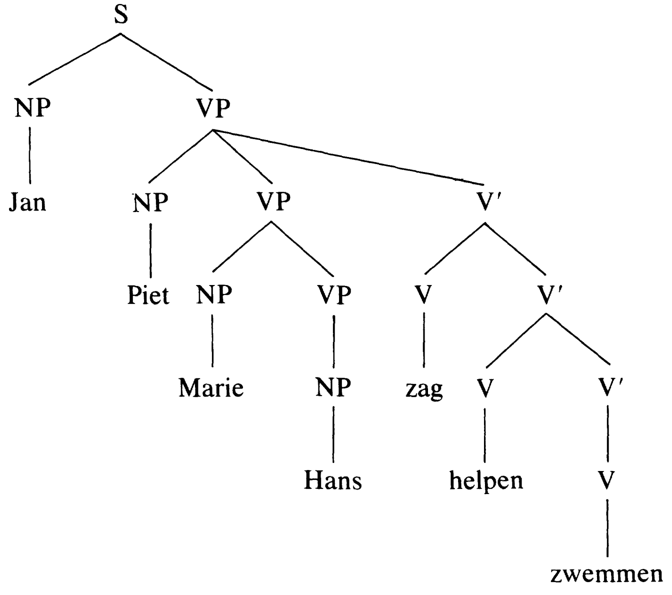
(45)

$$\left[\begin{array}{ll}
\text{SUBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`JAN'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{PRED} & \text{`SEE}\langle(\uparrow\ \text{SUBJ})\,(\uparrow\ \text{OBJ})\,(\uparrow\ \text{VCOMP})\rangle\text{'}\\
\text{TENSE} & \text{PAST}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`PIET'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{VCOMP} & \left[\begin{array}{ll}
\text{SUBJ} & \\
\text{PRED} & \text{`HELP}\langle(\uparrow\ \text{SUBJ})\,(\uparrow\ \text{OBJ})\,(\uparrow\ \text{VCOMP})\rangle\text{'}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{`MARIE'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{VCOMP} & \left[\begin{array}{ll}
\text{SUBJ} & \\
\text{PRED} & \text{`MAKE}\langle(\uparrow\ \text{SUBJ})\,(\uparrow\ \text{OBJ})\,(\uparrow\ \text{VCOMP})\rangle\text{'}\\
\text{OBJ} & \\
\text{VCOMP} & \left[\begin{array}{ll}\text{SUBJ} & \\ \text{PRED} & \text{`SWIM}\ \langle(\uparrow\ \text{SUBJ})\rangle\text{'}\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]$$

The tree in (47) is the c-structure for the string (46), which has an additional NP but not an additional verb.

(46)	*. . . dat Jan Piet Marie Hans zag	helpen zwemmen
	that Jan Piet Marie Hans see-past	help-inf swim-inf

(47)



An extra VCOMP level containing an OBJ function is specified on the VP branch of this c-structure. As shown in (48), this becomes the OBJ for the SWIM PRED. Because the SWIM semantic form does not subcategorize for OBJ, this f-structure violates the Coherence Condition. Again the sentence is rejected.

(48)
$$\left[\begin{array}{ll}
\text{SUBJ} & \left[\begin{array}{ll}\text{PRED} & \text{'JAN'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{PRED} & \text{'SEE}\langle(\uparrow \text{SUBJ})(\uparrow \text{OBJ})(\uparrow \text{VCOMP})\rangle\text{'}\\
\text{TENSE} & \text{PAST}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{'PIET'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{VCOMP} & \left[\begin{array}{ll}
\text{SUBJ} & \\
\text{PRED} & \text{'HELP}\langle(\uparrow \text{SUBJ})(\uparrow \text{OBJ})(\uparrow \text{VCOMP})\rangle\text{'}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{'MARIE'}\\ \text{NUM} & \text{SG}\end{array}\right]\\
\text{VCOMP} & \left[\begin{array}{ll}
\text{SUBJ} & \\
\text{PRED} & \text{'SWIM}\langle(\uparrow \text{SUBJ})\rangle\text{'}\\
\text{OBJ} & \left[\begin{array}{ll}\text{PRED} & \text{'HANS'}\\ \text{NUM} & \text{SG}\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right]$$

Note that because of the optional expansions of V′ under both VP and V′, these rules generate grammatical sentences with nested dependencies (illustrated in (11)) as well as grammatical sentences with mixed nested and crossed dependencies. Generation of the same verb in both positions, however, is ruled out by the Uniqueness Condition, because of the unique instantiation of semantic forms (Kaplan and Bresnan (1982), Grimshaw (1982b)); and omission of a verb in both positions is ruled out by the Coherence Condition. Thus, the general well-formedness conditions on f-structures explain why these two structures appear to be related by a movement of the verb.

In summary, we have presented a lexical-functional grammar fragment that assigns syntactically motivated c-structures to an infinite set of Dutch sentences with cross-serial argument–predicate associations. Given the general conditions on f-structure well-formedness and the functional annotations that are needed independently to assign grammatical relations appropriate for subcategorization (Grimshaw (1982a)) and semantic interpretation (Halvorsen (1981)), this grammar generates no examples where the numbers of subcategorized objects and predicates are not properly matched.

6. Conclusion

While Dutch may or may not be context free in the weak sense, it is not strongly context free: there is no context-free grammar that can assign the correct structural descriptions to Dutch cross-serial dependency constructions. In these constructions the verbs are discontinuous from the verb phrases that contain their arguments.

The phenomenon of "discontinuous constituents"—that is, noncontiguous constituents defining single functional units—is pervasive in natural language. It occurs in its most extreme forms in Australian aboriginal languages such as Warlpiri (Hale (1979), Nash (1981), Simpson (in preparation), Simpson and Bresnan (1982)). It is found in much less extreme forms in the clitic doubling phenomena of Romance (Montalbetti (1981)) and in the verb-agreement phenomena of Athapaskan (Roberts (1981)). The transformational solution to the Dutch case does not generalize to these kinds of cases, but the LFG solution does. This in itself is remarkable in view of the greater restrictiveness of lexical-functional grammars.

References

Bach, E. (1979) "Control in Montague Grammar," *Linguistic Inquiry* 10, 515–531.

Bach, E. (1980) "In Defense of Passive," *Linguistics and Philosophy* 3, 297–341.

Bach, E. (1981) "Discontinuous Constituents in Generalized Categorial Grammars," in V. Burke and J. Pustejovsky, eds., *Proceedings of the Eleventh Annual Meeting of the Northeastern Linguistic Society*, Graduate Linguistics Student Association, University of Massachusetts at Amherst.

Bar-Hillel, Y., C. Gaifman, and E. Shamir (1960) "On Categorial and Phrase-Structure Grammars," *Bulletin of the Research Council of Israel*, 9F.

Chomsky, N. (1956) "Three Models for the Description of Language," *IRE Transactions on Information Theory IT-2*, 113–134.

Chomsky, N. (1957) *Syntactic Structures*, Mouton, The Hague.

Evers, A. (1975) *The Transformational Cycle in Dutch and German*, Doctoral dissertation, Rijksuniversiteit, Utrecht, Holland.

Gazdar, G. (1981) "Unbounded Dependencies and Coordinate Structure," *Linguistic Inquiry* 12, 155–184.

Gazdar, G. (in press) "Phrase Structure Grammar," in P. Jacobson and G. K. Pullum, eds., *The Nature of Syntactic Representation*, Reidel, Dordrecht.

Grimshaw, J. (1982a) "Theories of Subcategorization," paper presented at the First West Coast Conference on Formal Linguistics, Stanford University, January 1982.

Grimshaw, J. (1982b) "On the Lexical Representation of Romance Reflexive Clitics," in J. Bresnan, ed., *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, Massachusetts.

Hale, K. (1979) "On the Position of Walbiri in a Typology of the Base," available from the Indiana University Linguistics Club, Bloomington, Indiana.

Halvorsen, P.-K. (1981) "An Interpretation Procedure for Functional Structures," unpublished manuscript, MIT, Cambridge, Massachusetts.

Huybregts, M. A. C. (1976) "Overlapping Dependencies in Dutch," *Utrecht Working Papers in Linguistics* 1, 24–65.

Kaplan, R. M. and J. Bresnan (1982) "Lexical-Functional Grammar: A Formal System for Grammatical Representation," in J. Bresnan, ed., *The Mental Representation of Grammatical Relations*, MIT Press, Cambridge, Massachusetts.

Knuth, D. (1965) "On the Translation of Languages from Left to Right," *Information and Control* 8, 607–639.

Langendoen, D. T. (1977) "On the Inadequacy of Type-3 and Type-2 Grammars for Human Languages," in P. J. Hopper, ed., *Studies in Descriptive and Historical Linguistics: Festschrift for Winfred P. Lehmann*, John Benjamin, Amsterdam, 159–171.

Montalbetti, M. (1981) "Consistency and Clitics," unpublished manuscript, Department of Linguistics and Philosophy, MIT, Cambridge, Massachusetts.

Nash, D. (1981) *Topics in Walbiri Grammar*, Doctoral dissertation, MIT, Cambridge, Massachusetts.

Postal, P. (1964a) *Constituent Structure*, Indiana University Research Center in Anthropology, Folklore, and Linguistics, Publication 30, Indiana University, Bloomington, Indiana.

Postal, P. (1964b) "Limitations of Phrase Structure Grammars," in J. Fodor and J. Katz, eds., *The Structure of Language*, Prentice-Hall, Englewood Cliffs, New Jersey, 137–151.

Pullum, G. K. and G. Gazdar (in press) "Natural Languages and Context-free Languages," *Linguistics and Philosophy*.

Roberts, J. (1981) "Toward a Unified Analysis of the Passive in Navajo," unpublished manuscript, Department of Linguistics and Philosophy, MIT, Cambridge, Massachusetts.

Simpson, J. (in preparation) *Control and Predication in Warlpiri*, Doctoral dissertation, MIT, Cambridge, Massachusetts.

Simpson, J. and J. Bresnan (1982) "Control and Obviation in Warlpiri," paper presented at the First West Coast Conference on Formal Linguistics, Stanford University, January 1982.

Thatcher, J. W. (1967) "Characterizing Derivation Trees of Context-free Grammars through a Generalization of Finite Automata Theory," *Journal of Computer and System Science* 1, 317–322.

Thatcher, J. W. (1973) "Tree Automata: An Informal Survey," in A. V. Aho, ed., *Currents in the Theory of Computing*, Prentice-Hall, Englewood Cliffs, New Jersey, 143–172.

(Bresnan)
20D–219
MIT
Cambridge, Massachusetts 02139

(Kaplan)
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304

(Peters)
Center for Advanced Study
in the Behavioral Sciences
and
Department of Linguistics
University of Texas
Austin, Texas 78712

(Zaenen)
Department of Linguistics
Science Center 223
Harvard University
Cambridge, Massachusetts 02138