[*International Conference on the State of Machine Translation in America, Asia and Europe*. Proceedings of IAI-MT86, 20-22 August 1986, Bürgerhaus, Dudweiler]

An LFG-based Approach To Machine Translation

Klaus Netter / Juergen Wedekind
Department of Linguistics
University of Stuttgart
West Germany

I.1. PRELIMINARY REMARKS*

The project's goal is to integrate results of recent research in the fields of theoretical linguistics and computer science. In this way we aim to develop a system for machine translation by a methodologically sound procedure and to fulfill two conditions of adequacy:

(i) The system should be based on a well-established linguistic theory that includes a clearly defined grammar formalism. (The format of the grammatical theory is of overall importance insofar as all components of the system intrinsically depend on the type of grammar chosen.)

(ii) With the exception of language particular descriptive components (grammars, transfer rules), the components of the system should be suitable for universal application, i.e., usable by any grammar of the chosen type. Conversely the (language-dependent) components may serve as input to any component compatible with the given formalism.

The decision to design our system on the basis of these conditions was motivated by several theoretical as well as practical considerations:

1) The obvious advantages of 'buying' a well-established (and well-specified) grammatical theory 'off the shelf': It can be assumed to have been tested sufficiently for practical application, and to have been examined with respect to its most fundamental formal properties (e.g. decidability). This saves on the preliminary basic research, always necessary to establish the fundamental practicability and feasibility of a new formalism.

2) Not only can one exploit results of previous research, but also the accessibility of project work makes possible innovational feedback from other scientists working in the same paradigm. Indirectly this increases the number of collaborators.

3) Since the development of algorithms for the various components is meant to yield general theoretical insights, the system's components should be independent of the actual application domain. Consequently the various components can be used where ever grammars of the chosen type are applied. This guarantees a certain projectability of research results.

4) A clearly defined linguistic formalism permits one to develop the grammatical and the computational components independently and therefore in parallel, which facilitates inter-disciplinary cooperation of theoretical linguists and computer scientists. Neither linguists nor computer scientists have to wait for each other's research results. Nor do linguists and computer scientists have to re-discover results well-known to the experts in the alien discipline.

The choice of LFG as grammatical theory was motivated by the following considerations:

a) LFGs are decidable and generate a superset of the class of context-free languages. This guarantees practical applicability and certainly provides enough expressive power for describing natural languages.

b) The theory was developed by linguists. Consequently one can expect, that it provides enough heuristics for the empirical application by theoretical linguists. The formalism is sufficiently explicit to be applied by computer scientists.

c) The control-structure (functional- or f-structure) built up by an LFG for each sentence of a language seems to contain enough syntactic as well as semantic information about the sentence to translate it adequately on a context independent, sentence-by-sentence basis using systematic rules. Such a translation system will inherently have an upper limit on the quality of translation.

The working hypothesis in c) receives support from the fact, that a purely semantic representation, reducing the sentence to its truth conditions, is too weak on the one hand -- as one might like to draw on information about the syntactic form of the source expression (e.g. passive, relative clauses, topic/ focus structure, etc.) -- and too strong on the other hand as ambiguities of scope that are not reflected in the surface string will normally not have to be taken into consideration. [fn.l]

I.2. STRUCTURE AND STATE OF DEVELOPMENT OF THE SYSTEM

In the following we give a short survey on the structure and components of the system as well as the time schedule for its development. (Two of these components will be introduced in more detail in the later chapters.)

A. Linguistic Input

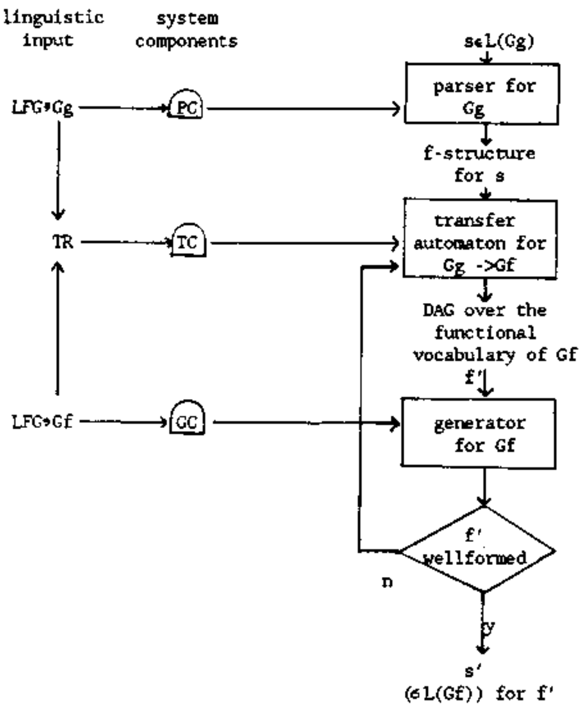
- Gg (German grammar), under development since April 1985, (cf. III.)
- Fg (French grammar), under development since April 1985
- TR (Transfer rules):
  - lexical TR , under development since April 1985,
  - structural TR, second phase from 1987 onwards

B. System Components
- PC (parser compiler), existing for top-down parsing, under development for bottom-up parsing. [fn.2]
- GC (generator compiler), existing algorithm (cf. IV.), implementation under development.
- TC (transfer automaton compiler) second phase of project

Adequacy condition I.(ii) above can be elaborated for the language independent components as follows:

It should be provable for the construction algorithms on which PC and GC are based that for any LFG the constructed
- (PC) parser accepts an input string s by building up an f-structure f iff
- (GC) generator accepts an input f-structure f by building up a terminal string s iff

s is element of the language and has the f-structure f.



II. THE FRAGMENT OF GERMAN GRAMMAR

This section will give a short survey of the fragment currently covered by the German grammar and will describe some parts of it in more detail.
The German grammar contains descriptions for the following structures and phenomena:

1) Sentence type and sentence mood
2) Position of the finite verb and separable verb prefixes
3) Infinitival constructions
4) Periphrastic tense forms and passives
5) Extraposition phenomena
6) Free word order of NP- and PP-complements

In section II.1. we give a rough sketch of some differences between the version of LFG we use and the one given in [9] . Since much of the grammar has or will be documented elsewhere ([11], [12]) we will concentrate in the sections II.2. and II.3. on the points mentioned under 1) and 2).

II.1. DIFFERENCES TO THE STANDARD VERSION OF LFG

In the following we will sketch some of the relevant differences between the traditional KB version of LFG and the one used in our system. This version of LFG differs from the one given in [9] with respect to the treatment of 1) unbounded dependencies and 2) non-configurational encoding.

1) In [9] unbounded dependencies were captured by a mechanism called constituent control. The long distance identification of the controlling and the controlled f-structure was established by bounded domination meta-variables properly annotated to the righthand sides of the rules. Thus it was possible to express an infinite number of different controller-controllee linkages, if a recursive category from which the controlee was derivable was reachable from the domain root.

The new treatment of unbounded dependencies expresses these possible linkages by means of regular expressions over the functional vocabulary. An annotated regular expression denotes all f-structure paths which can link the controlling and controlled f-structure. Both mechanisms are closely related since one can indicate for a given controller-controllee pair a regular expression that denotes all f-structure paths which can link the controller and controllee in derivations of c-structures containing the controllee, from the root node of the controller. As a regular expression denotes paths that are generable using a regular grammar with annotated unary rules, an additional condition has to ensure that the set of paths consistent with the f-structure is not infinite, if the functions are not governable functions (ADJ.etc.). (In the old version this is guaranteed, as unary recursive categories are excluded such that a recursively derived iteration of ungoverned functions has to become apparent in the surface string.)

In [10] the coherence condition of [9] is modified to an 'extended coherence condition' which covers the set of nuclear functions. The nuclear functions are the governable functions plus ADJ and XADJ, which are all together the functions the regular expressions are formulated over. For the ungovernable functions among the nuclear functions the extension of the coherence condition says the following: each of these functions has to be element of an f-structure with a PRED attribute. Thus the length of a path which is element of the set denoted by a regular expression is bounded by the length of the string.

2) We have introduced variables to dispense with disjunctions for the non-configurational encoding of grammatical functions (cf. [16]).

The principle of non-configurational encoding is to annotate disjunctions of pairs of function assigning and feature assigning equations. The disjuncts have the form < (↑G) = ↓, (↓ F) = v > (cf [1]). Which function of the disjunction is selected, depends on the value of the feature F. Each disjunction can be assigned bijectively a variable ranging over the set of functions of the disjunction. Each function in the range is assigned the feature-assigning equation (in a more general version a partial f-structure) which selects the function the variable is instantiated by.

Thus it is possible to store the range and the feature assigning equations for the variables independent from the lexicon and the rule system and to instantiate the variables as soon as the f-structure information necessary for the instantiation is available. This reduces the amount of backtracking and makes the rule system a little bit clearer.

II.2. SENTENCE MOOD AND SENTENCE TYPE

The underlying idea in our treatment of sentence mood and sentence type follows the assumption that (in German) these two features are basically determined by a sentence initial structural position.

We will distinguish between sentence S-MOOD (ranging over feature values like interrogative, assertive etc.) and sentence S-TYPE (with values like subordinate, relative etc.). We hypothesize that in a theory like LFG, where no use is made of information about categorial distinctions in the subcategorization frames of lexical items, both features will be necessary to characterize a clause sufficiently. Take for example verbs like 'behaupten' vs. 'fragen': Both may take finite structures as complements,

however the first will accept any sentence that may be characterized as assertive, whereas the latter is sensitive to the distinction between (possible) main clause vs. subordinate clause (see below).
To cover at least some of the relevant data of subordination and sentence mood one has to account for the fact that
- main clauses in German may be introduced by structures like:

(a) XP + Vfin (assertive / declaratives)
(b) XP + Vfin (Wh-interrogatives, the XP has to contain a interrogative pronoun)
(c) Vfin (Yes/No interrogatives)

- subordinate clauses follow the patterns:

(d) CONJ (conjunctive clauses)
(e) XP (relative clauses, the relative pronoun has to agree with the noun the relative clause refers to and subordinate interrogatives)
(f) XP + Vfin (assertive subordinate clauses)

These structures are illustrated by the following sentences and clauses (C and S are two categories into which we divide these sentences, see below):

	C	S (-VP)
(1)	Der Mann hat	das Buch gelesen. (a)
(2)	Das Buch hat	der Mann gelesen.
(3)	Wer hat	das Buch gelesen? (b)
(4)	Hat	der Mann das Buch gelesen? (c)
(5)	daß	der Mann das Buch gelesen hat. (d)
(6)	der	das Buch gelesen hat. (e)
(7)	wer	das Buch gelesen hat. (e)

		C	S
(8)	die Behauptung,	daß/*ob	er das Buch gelesen habe
(9)	die Behauptung,	er habe	das Buch gelesen
(10)	die Frage,	*daß / ob	er das Buch gelesen hat,
(11)	die Frage,	wer/*der	das Buch gelesen hat,
(12)	die Frage,	*hat	er das Buch gelesen, ...
(13)	der Mann,	*wer / der	das Buch gelesen hat.

The sentences (8)-(13) exemplify that in German certain lexical items are compatible only with specific conjunctions; also, (similar to English) there are lexical items that permit assertive main clause structures as subordinate clauses but not interrogative structures like (12). As an additional complication German distinguishes between d_relative pronouns and w_pronouns, the latter being used in free relative clauses and subordinate interrogative clauses. [fn.3]

To capture these regularities we make the following assumptions:
- All finite clauses are categorized as S'
- S' dominates a category which we call C [fn.4] and a category S
- The category S in German is equivalent to VP. It may or nay not contain the SUBJect NP. Also it may or may not contain the finite verb.
- C may be expanded further or it may immediately dominate a terminal symbol (conjunction)

Irrespective of the theoretical status or value of these assumptions is, it should be obvious that this subdivision of sentences permits to isolate a structural position C which is subject to much variation from a structure S which can be shown to be independent of this variation.

The start rule for the grammar is then (Rl) [fn.5]

```
(Rl)  S' ->  C    ↑ = ↓
                (↑ TENSE)
              [S ↑ = ↓]
```

Since we assume only finite sentences to have a C constituent, this symbol is assigned an existential constraint requiring the sentence to have a verbal head that contains a value for TENSE. The trivial equation assigned to C may be justified for two reasons: On the one hand C may contain the functional head of the sentence, the finite verb, on the other hand (as long as one doesn't assume that conjunctions are subcategorized for sentential complements) this rule permits to isolate a structural position without affecting the corresponding f-structures.

The category C could then be expanded as follows: We do not make a strict distinction between lexical categories and syntactic categories, in the sense of deriving lexical items only from lexical categories. In theory it thus becomes possible to access the lexicon from any non-terminal symbol. Our LFG system, which is built along this line, thus provides a very simple way of making sure that conjunctions like 'daß' (that), 'ob' (whether) will always appear in a complementary distribution with the finite verb by categorizing them as C right away:

```
(L1) daß:    C,    (↑ S-TYPE) - subordinate
                   (↑ S-MOOD) - assertive
(L2) ob:     C,    (↑ S-TYPE) - subordinate
                   (↑ S-MOOD) - interrogative
```

For relative clauses and subordinate clauses introduced by a w_constituent, we have to expand C to an NP. Suppose we have NP-rules like (R2)/(R3) which will derive expressions like 'der Mann' or 'Peters Vaters Hut': [fn.6]

```
(R2)   NP   ->   DET  ↑ = ↓
                 N'   ↑ = ↓
(R3)   NP   ->   NP  (↑ POSS) = ↓
                     (↓ CASE) = genitive
                 N'   ↑ = ↓
```

Relative and interrogative pronouns like 'der', 'wer' (who), 'dessen', 'wessen' (whose) in these constructions always have to be exhaustively dominated by an NP-node. (This node can be the highest NP-node itself or a possessive NP at the lowest level of embedding.) Assigning to these pronouns the category NP will consequently yield the desired results in connection with rule (R4):

```
(L3) wer:      NP,   (↑ PRED) = "pro"
                     (↑ WH) = qu
                     (↑ AGR NUMBER) = sing
                     (↑ CASE) = nom.
(L4) der:      NP,   (↑ PRED) = "pro"
                     (↑ WH) = rel
                     (↑ AGR NUMBER) = sing
                     (↑ AGR GENDER) = masc
                     (↑ CASE) = nom.
(L5) dessen:   NP,   (↑ PRED) = "pro"
                     …
                     (↑ CASE) = genitive.
```

(The feature AGReement seems to be useful to subsume a bundle of features like Number, Gender and Person; although complex-valued,AGR is not to be interpreted as a function.)

(R4) C -> NP (↑ XCOMP* G) = ↓
(/(↑ QU-POCUS) = (↓ POSS*)
(↑ QU-POCUS WH) =c qu
(↑ S-MOOD) = interrogative
(↑ S-TYPE) = subordinate
/(↑ TOPIC) = (↓ POSS*)
(↑ TOPIC WH) =c rel)
(↑ S-TYPE) = relative /).

The first functional schema annotated at the NP in (R4) will have the effect that this constituent is analyzed as a governable function contained in the range of the variable G, and as representing the argument of a verb, which may be arbitrarily deep embedded in an f-structure that can be reached along a path containing any number n of XCOMPs. The relative pronoun 'dem' in (14) is the OBJ2 of the embedded and extraposed non-finite verb 'schenken':

(14) der Mann, den sie versucht hat, ein Buch zu schenken.

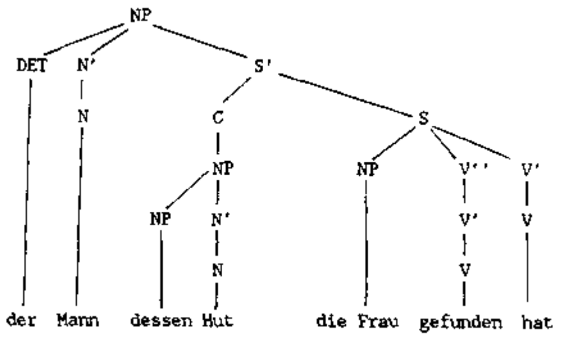
The functional control schema referring to QU-FOCUS and TOPIC equates the partial f-structure which represents either the f-structure corresponding to the NP itself or to a POSS-function within the NP with the value of these attributes. This attribute will be accessible at the top level of the sentence's f-structure, which will be necessary and useful for two reasons: In interrogatives one would obviously like to know, what the focus of the question is; in relative clauses one has to guarantee that the respective relative pronoun agrees with the nominal head of the NP which this relative clause refers to. By formulating a constraining equation over these f-structures, requiring that they (and equally the corresponding partial f-structure within the NP) will have a value for WH, we make sure that the NP actually contains an appropriate pronoun at some unpredictably deep level of embedding.

Modifying the NP-rule as in (R5) we will obtain for a string like (15) c-structure (C1) and f-structure (F1).

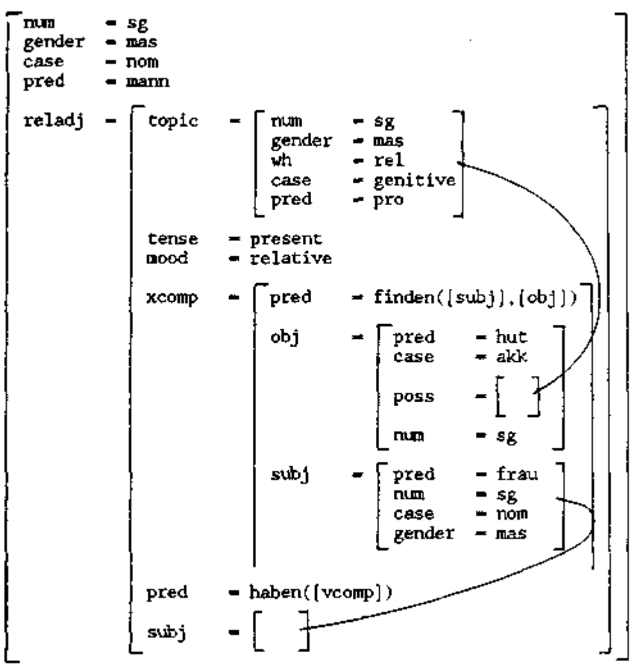
(R5) NP -> DET ↑ = ↓
N' ↑ = ↓
[S' (↑ RELADJ) = ↓
(↑ AGR) = (↓ TOPIC AGR)
(↓ TOPIC WH) =c rel].

(14) der Mann, dessen Hut die Frau gefunden hat.

(15) (C1)



(F1)



To derive the main clauses (WH-, Y/N-interrogatives and declaratives) we will expand C in the following way:

(R6) C -> [NP (↑ XCCMP* G) = ↓
(/(↑ S-MOOD) = assertive
/(↑ QU-FOCUS) = (↓ POSS*)
(↑ QU-FOCUS WH) =c qu
(↑ S-MOOD) = interrogative /}]
V ↑ = ↓
(/(↑ S-MOOD) =c assertive
/(↑ S-MOOD) = interrogative /).

The constraining equation at the verb will guarantee, that it is preceded by a constituent in declarative clauses. In the other cases either an additional QU-FOCUS will be contained in the f-structure marking a WH-interrogative and its focus, or C will exhaustively dominate the verb indicating (Y/N-)interrogative mood.
Noun and verbs requiring a specific sentential complement can then be subcategorized accordingly:

(L6) behauptet: V, (↑ PRED) = "behaupt <(SUBJ) (COMP)>"
(T COMP S-MCOD) -c assertive
…
(L7) Behauptung: N, (↑ PRED) = "behaupt <(SUBJ) (COMP)>"
(↑ COMP S-MOOD) =c assertive
…
(L8) Frage: N, (↑ PRED) = "frag <(SUBJ) (COMP)>"
(↑ COMP S-MOOD) =c interrogative
(↑ COMP S-TYPE) =c subordinate
…

The expansion of N' in the NP-rule will then be:

(R7) N' -> N ↑ = ↓
[S' (↑ COMP) = ↓].

and cover the data given in (8) - (13).

II.3. SEPARABLE VERB-PREFIXES

In section II.2. we assumed, that the finite verb will always be in a complementary distribution with subordinative conjunctions and WH-constituents in sub-ordinate clauses. This means that if the finite verb cannot be derived within the C-position it has to be derived where all the other (non-finite) verbforms also have to be derived: to the right of their respective arguments in a clause final position (modulo extraposed constructions):

```
(16) Er wird  das Buch gelesen haben.
(17) daß  er das Buch gelesen haben wird.
(18) wer     das Buch gelesen haben wird, .. .
(19) daß  er das Buch zu lesen haben wird.
```

One of the interesting phenomenon connected with this clause-final position are separable verb prefixes:

```
(20)Er hat        das Problem dargestellt  .
    he has        the problem presented.
(21)Daß        er das Problem dargestellt hat.
(22)daß er        das Problem darzustellen hat.
(23)Er stellt     das Problem dar.
(24)Daß        er das Problem darstellt.
(25)*Er darstellt das Problem.
```

Clearly, it is only the finite stem that may occur in a sentence initial position, in all other cases the verb occurs together with the prefix, being separated only by a participle ('ge') or infinitive marker ('zu)'. (Actually one could interpret this distribution as an indication that it is not the finite verb itself, but the finite features that are bound to the C-position. In cases where this position is not realized by a conjunction, which would 'expel' the finite features, some phonological material categorized as V has to 'move' from its clause-final position to the initial position in order to take up and phonologically realize these abstract features.)

In order to account for these data we assume that the (finite and non-finite) verb forms occurring in the sentence final position correspond to a higher projection V of the lexical category V. [fn.7]

As mentioned in II.2. we also assume that this position is immediately dominated by S and that all non-finite forms will be derived from a left-branching verb complex:

```
(R8)  S    ->      NP* (↑ XCOMP* G) = ↓
                   [V"  (↑ XCCMP) = ↓
                   [V’  ↑ = ↓].
(R9)  V"   ->     [V" (↑ XCCMP) = ↓]
                  V  ↑ = ↓.

(R10) V    ->     [ZU (↑  INF) = zu]
                  v ↑ = ↓.
```

These rules will work fine with simple verb entries like the following:

```
(L9) wird:   V,        (↑ PRED) = "werden < (XCCMP) > (SUBJ)"
                       (↑ SUBJ) = (↑ XCOMP SUBJ)
                       (↑ XCOMP INF) = bare.

(L10) haben: V,        (↑ PRED) = "haben < (XCOMP) > (SUBJ)"
                       (↑ SUBJ) = (↑ XCOMP SUBJ)
                        {/ (↑ XCOMP INF) = ge
                         /  (↑ XCOMP INF) = zu /}.
```

```
(L11) liest:    V,      (↑ PRED) = "lesen < (SUBJ) (OBJ) >"
                        (↑ TENSE) = pres
                        (↑ SUBJ NUM) = sg
                        (↑ SUBJ CASE) = nom
                        (↑ OBJ CASE) = akk.

(L12) lesen:  V,         (↑ PRED) = "lesen < (SUBJ) (OBJ) >"
                         (↑  OBJ CASE) = akk.

(L13) gelesen: V,       (↑ PRED) = "lesen <(SUBJ) (OBJ)>"
                        (↑ OBJ CASE) = akk
                        (↑ INF) = ge.
```

To capture the various constructions with separable pre-fixes, we assume the following: Forms like 'darstellt', 'darzustellen' are categorized as V, i.e. as morpho-syntactically complex forms. (If we assume that the zu-infinitive may either be produced by a lexical or by a syntactic rule, it is more adequate to let forms, that already incorporate the 'zu'- and 'ge'- infinitival marker, be exhaustively dominated by V rather than inserting them under the V nodes as the right sister of an optional zu-infinitiv marker. If darzustellen were categorized as V there would be nothing to prevent zu as cooccurring as its left sister; the two f-schemata 'INF = zu' would simply be unified.) Most important, categorizing the finite forms like 'darstellt' as V will automatically prevent complex forms to be inserted under the C-node in (R6). [fn.8]

```
(L14) darstellt: V,
                        (↑ PRED) = "darstellen <(SUBJ) (OBJ)>"
                        (↑ TENSE) = pres
                        (↑ SUBJ NUM) = sg
                        (↑ SUBJ CASE) = nom
                        (↑ OBJ CASE) = akk.
(L15) darzustellen: V',
                        (↑ PRED) = "darstellen <(SUBJ) (OBJ)>"
                        (↑ OBJ CASE) = akk
                        (↑ INF) = zu.

(L16) dargestellt: V',
                        (↑ PRED) = "darstellen <(SUBJ) (OBJ)>"
                        (↑ OBJ CASE) = akk
                        (↑ INF) = ge.
```

To account for the constructions where the prefix is separated from the verb, there are basically three options open to us (in fact one might want to use all of these possibilities to account for the whole range of phenomena from prefix verbs to idiomatized complements).

1) We could introduce a feature PREF and identify the values of PREF with governable functions in the semantic forms of the simple verb. In the V-rule (R11) we intro-duce the prefix as PRE annotating it by (↑ (↓ PREF)) = ↓, analogously to the treatment of prepositional objects by (↑ (↓ PCASE)) = ↓.

```
(L17) macht: V,
               (↑ PRED) = "machen < (SUBJ) (OBJ) (AUF PREF) >"


(L18) auf:    PRE,       (↑ PREF) = AUF PREF
                          (↑ PRED) = "auf".

(L19) aufmacht: V',
               (↑ PRED) = "machen < (SUBJ) (OBJ) (AUF PREF) >"  '
               (↑ AUF PREF PRED) = "auf".

(Rll)  V'     - >           [PRE    (↑ (v PREF)) = ↓
                            [ZU     (↑ INF) = zu ]
                            [V      ↑ = ↓]
```

The principles of consistency would then take care of all the rest. This could be motivated to a certain degree by the fact that especially with verbs of movement the prefix binds one argument position, that would otherwise be taken by an obligatory directional adverbial phrase:

(26) Er ging weg / hinein / nach Hause ...
(27) Er warf das Buch weg / hinein / in den Mülleimer ...

However there is a serious objection to be raised against this approach: It is certainly not easy to justify that these prefixes should (always) have the same linguistic status as SUBJ, OBJ etc., i.e. that prefixes should be arguments rather than part of the predicate. Especially in the case of clearly lexicalized forms like 'darstellen', 'übersetzen' it seems rather counterintuitive to assume an atomic predicate 'stellen' taking a (quite meaningless) argument 'dar', rather than having an atomic predicate 'darstellen' right away. As mentioned, we do not want to rule out this approach right from the start and for all complex verbs, however if we go for it, it might put a heavy load on (compositional) semantics.

2) To avoid this drawback we could have a lexical rule, that will generate for every complex entry PREF#VERB an entry for the verb, which will specify the meaning of the whole complex as an atomic predicate in its semantic form, as well as an entry for the prefix, which will specify its form by means of a FORM-feature:

```
(L20) stellt:  V        (↑ PRED) = "darstellen < (SUBJ) (OBJ) >"
                        (↑ FORM) =c dar
                        ...
(L21) dar:   PRE,       (↑ FORM) = dar.
```

This solution however requires that simple verb forms that must not cooccur with a prefix must be assigned a negative constraint:

(28) *Er liest das Buch dar.

```
(L22) liest:  V,        (↑ PRED) = "lesen < (SUBJ) (OBJ) >"
                        -(↑ FORM)
                        ...
```

3) The third possibility may look strange at first sight, but might be the most adequate solution from a linguistic point of view. If we look at coordinate structures like (29) they suggest, that it is actually the prefix that 'carries the semantic information' :

(29) Peter macht die Türe auf und das Fenster zu.

If 'macht' in these cases is assigned a predicate 'aufmachen' or 'zumachen' we would have to introduce the complementary predicate by means of some coordination rule. However if we allow this we may also get two predicates ('ansehen' and 'sehen') in cases like (30) (which in 'my idiolect' is clearly out in a serious, non-punning context)

(30) Peter sieht die Frau und das Auto an.

The solution we have decided on therefore is to annotate the PRED-entry at the prefix and conversely a FORM-feature at the verb besides the respective TENSE and AGR-features:

```
(L23) dar:  PRE,        (↑ PRED) = "darstellen < (SUBJ) (OBJ) >"
                        (↑ FORM) =c stellen.

(L24) stellt:  V,       {/(↑ PRED) = "stellen < ...>"
                         /(↑ FORM) = stellen /
                         (↑TENSE) = present
```

In case the 'wrong' disjunct were chosen (i.e. if there were a FORM feature without an appropriate prefix being present or conversely two PRED's), the f-structures would be rejected as ill-formed. (Note that this solution does not involve any additional rule schemata beyond those required also in the second solution.)

To incorporate this solution in the S-rule we have to spell out some of the permissible combinations within V : As the V rule should always derive a V-constituent if it is non-finite [fn.8], we have to spell out at least some of the possible combinations in the clause final position immediately dominated by S: a) PRE, b) PRE + V, c) V, d) 0.
The cases b)/c) will be covered by the V'-rule (R10) as it stands. To capture a) and d) we simply modify the S-rule (R8) notating an alternative option PRE or an optional V':

```
(R12)  S    - >      NP*  (↑ XCOMP* G) = ↓
                     [V"  (↑ XCOMP) = ↓ ]
                     (/[V   ↑ = v
                              - (^ FORM)]
                      /P    ↑ = ↓/).
```

(The negative constraint on the form feature will prevent a verb with FORM from being inserted in the clause final position next to a verb in V-initial position.)

HI. GENERATION

III.1. GENERAL REMARKS ON THE DESIGN

The algorithm for the generator-compiler constructs a generator for an arbitrary given LFG G. The generator is simply a recognizer for the set of (well-formed) f-structures of G. If it accepts an input structure, it builds up terminal strings as control structures. The set of control structures constructed for an input structure is exactly the set of all sentences of the language which have this structure as f-structure. Thus for each LFG the algorithm specifies a computable (binary) generability relation for which it is provable that a terminal string x is generable from a structure f iff x is derivable in G with f-structure f [fn.10). A language recognizer defines the relation between the set of terminal strings and the set of DAGs over the grammars functional vocabulary which is the inverse of the corresponding generator.

The generation process is based on the monostratal conception of derivation for LFGs. The structures of these monostratal derivations are partial f-structures augmented by additional information about derived symbols and their linear order. The concept of monostratal derivation is strengthened for generation in such a way that derivations are directed by an input f-structure. This structure-driven derivation forms a simple top-down recognition procedure of f-structures which has the properties enumerated in the first paragraph.

In the rest of this section, monostratal derivation and the generator construction algorithm will be described informally. For a more detailed and formally more explicit description see [17],[18].

III.2. MONOSTRATAL **DERIVATION**

For the class of LFGs fulfilling the conditions expressed by the contemporary version of the theory (cf. besides [9] especially [10]) it is possible to define an algorithm for the construction of equivalent monostratal variants.

According to the usual multistratal version (cf. [9]) a terminal string x is regarded as well-formed iff it satisfies the following conditions:

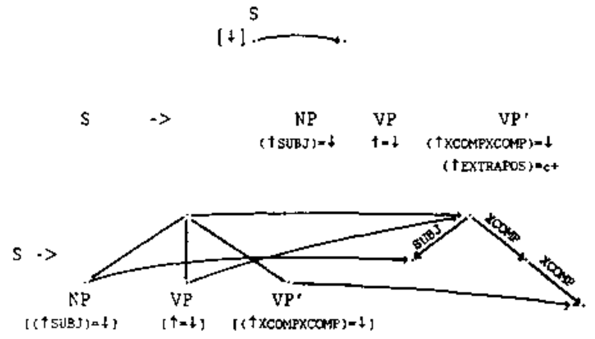
i. There is a c-structure for x that can be derived by the context-free base of the grammar.

ii. There is an f-structure d and a mapping f from the set of nodes of c to the set of subDAGs of d such that d is the only minimal f-structure that satisfies the annotations associated with the c-structure nodes. (The f-description solution algorithm constructs f and d.)

iii. All constraints in the f-description are satisfied by d.

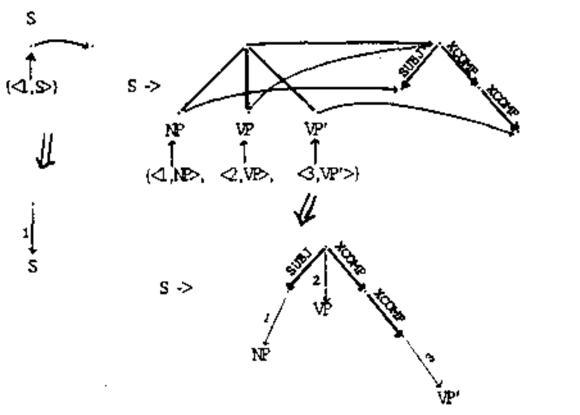
iv. d is complete and coherent.

In the Kaplan/Bresnan version a derivation of an annotated c-structure is a sequence of annotated c-structures, where, except for the initial structure, each annotated c-structure results from the application of an annotated context-free rule to a terminal node of the preceding structure. A rule introduces an annotated c-structure. The start structure for the derivation of well-formed strings is an S-labelled node annotated with 'V. If one applies the f-description solution algorithm to those annotated c-structures, it constructs in each successful case the only minimal partial f-structure and a mapping from the c-structure nodes to the substructures of the partial f-structure. The figure below shows as examples the results of the application of the algorithm to the start entity and a c-structure introduced by a rule [fn.11]:



S
[↓]
S -> NP VP VP'
(↑SUBJ)=↓ ↑=↓ (↑XCOMP XCOMP)=↓
(↑EXTRAPOS)=c+
S ->
SUBJ
XCOMP
XCOMP
NP VP VP'
[(↑SUBJ)=↓] [↑=↓] [(↑XCOMP XCOMP)=↓]

If one represents strings as sequences in the set theoretical sense the corresponding augmented f-structures are easily constructable: The occurrences (ordered pairs) of the string represented by the labelled terminal nodes of a c-structure tree are attached as additional labelled edges to the startnodes of those substructures of the partial f-structure to which the corresponding c-structure nodes are mapped. For the examples given above one will obtain the following entities (notationally these constructed augmented structures and rules are denotated by symbols with a lower a-index) [fn.12]



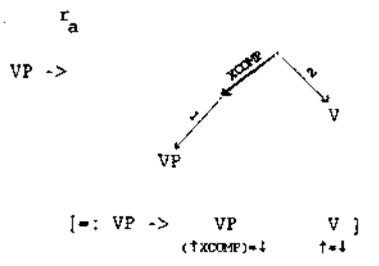
The string of such an augmented structure is the set of all integer labelled edges, [fn.13]

A structure s'_a that is directly derivable from a structure s_a by a rule r_a has to be equal to a structure which results from s_a
(i.) by a replacement with respect to the underlying string and
(ii.) by an extension with respect to the underlying f-structure.

A structure s'_a is directly derivable from a structure s_a by a rule r_a iff

I. the lefthand variable of the rule equals a variable of an integer labelled edge of s_a .

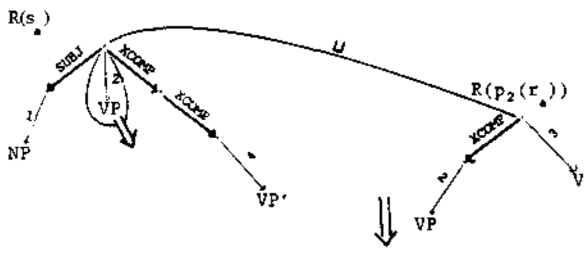
If we assume that we have derived the structure (s_a) which is introduced by the rule given above, this condition is met by the rule

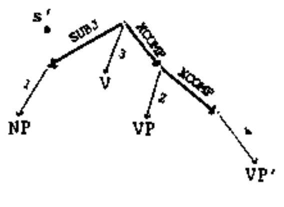


whose lefthand variable equals the variable of the <2,VP> edge. As the string of the derived DAG has to be the result of the replacement of (<1,VP>,<2,V>) for the second occurrence in the string of s_a (<2,VP>), one has to construct from s_a and the structure introduced by r_a new structures (R(s) , $R(p_2(r_a))$) whose integer labelled edges represent together the result of the replacement ((<1,NP>,<2,VP>,<3,V>,<4,VP'»); i.e.

$R(p_2(r_a))$ equals the structure introduced by r_a except that the integer labels of the edges are increased depending on the index of the edge the rule has to be applied to, and

$R(s_a)$equals s_a with the exception that i. the edge to which the rule has to be applied is eliminated in $R(s_a)$ and ii. the labels of the edges which correspond to the right context of the substituted occurrence in s_a are increased in $R(s_a)$ depending on the length of the string of the structure introduced by r_a .





The second defining condition is then:
II. s'_a equals the minimal extension of $R(s_a)$ which results from the unification of $R(p_2(r_a))$ with the subDAG of $R(s_a)$ to which the removed edge was attached.

If one defines the derivability concepts along the usual way, the <u>language</u> of an LFG is defined as the set of all terminal strings of those structures that are
- derivable from the start entity
- complete and coherent and
- fulfill all constraints.

III.3. GENERATION AS F-STRUCTURE-DRIVEN DERIVATION

As mentioned above, the generator constructed for an LFG accepts the set of f-structures of that LFG. Thus one can use an arbitrary superset of the set of f-structures as possible input-set. To exclude right from the start those structures as possible input which cannot fulfill some directly decidable necessary conditions for wellformed-ness, one can restrict the input-set. The following conditions for DAGs over the functional vocabulary of an LFG seem to express the most natural restriction:

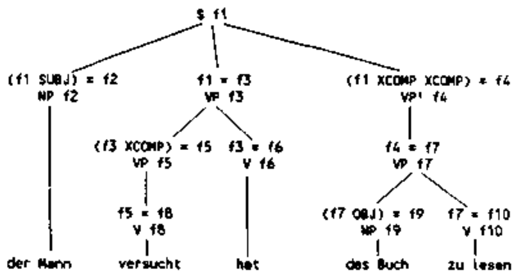
- nonterminal edges are labelled with complex-valued features (grammatical functions),
- terminal edges are labelled with atomic-valued features whose values are in the range respectively assigned to them (e.g. CASE cannot have the value SG),
- the structure is complete and coherent.

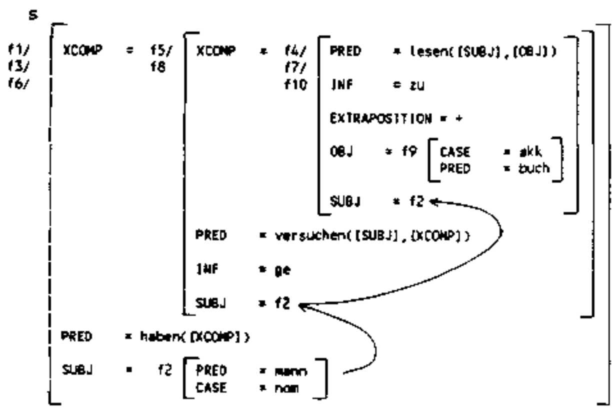
To start the derivation over an input structure in the sense described above, it is necessary to attach the <1,S> edge to the startnode of this DAG. The following condit-ions ensure that a terminal string is derivable iff it has the input structure as its f-structure:

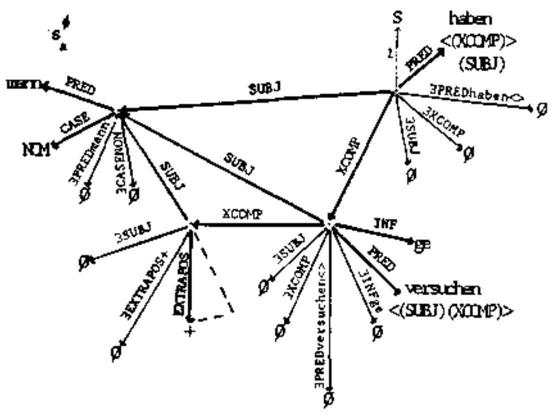
COHerence: The structure is not expanded by the rule applications except by additional integer labelled edges .

COMpleteness: Every function and every atomic feature value pair of the structure (and its substructures) are introduced by the rule applications. [fn.14]

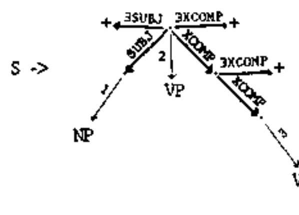
In order to check the COM-condition we introduce for each function and each atomic feature value pair a special existential feature. (e.g. OBJ - > ∃OBJ CASE=NOM - > ∃CASENOM) Each subDAG of an input structure is thus expanded by the empty-set-valued existential features for its atomic feature-value pairs and complex-valued features (functions). The start structures constructed by this extension from input structures are denotated by symbols which are indexed by an additional upper empty-set sign (to indicate the additional empty-set valued features). Given for example the following input (f-) structure s for the sentence with the c-structure tree, given below, the constructed start entity s is (represented as DAG) the following structure: (for the sake of clarity it is slightly simplified).







If the rules are augmented analogously with '+ - valued' existential features, the control of the COM-condition becomes a check on the non-emptiness of the existential feature values of the derived structure. The generator rules which are constructed from augmented rules are denotated by symbols with an upper '+' index. The generator rule constructed from the rule, given above, together with its (explicitly stated) constraint set is the following [fn.15]:



{XCOMP XCOMP EXTRAPOS=+}

Depending on the method for checking the COM-condition different ways to define the concept of direct generability are possible. Below we give the simplest (and least efficient) possibility. It simply amounts to ignoring the check during the derivation and integrating it as postponed filter in the definiens of generability [fn.16].

A structure s'_a is directly generable from a structure s_a by a generator rule r_a+ iff
I. The lefthand variable of the rule equals a variable of an integer labelled edge of s_a .
II. The substructure to which this edge is attached to is an extension of the functional structure underlying the structure introduced by the rule.

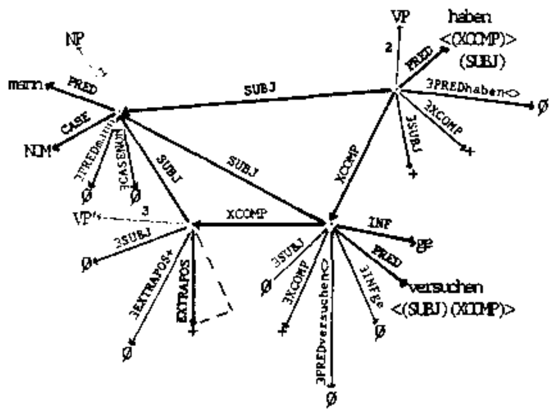
The underlying structure results from cutting off all integer labelled edges from the structure introduced by r_a(!). This condition ensures the COHerence.

III. The substructure to which this edge is attached, satisfies all constraints expressed by the rule.

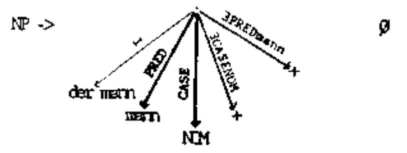
Since the structures of a generation process are complete-ly specified with respect to the underlying functional structure it is possible to check the constraints expressed by the rules in tandem with the generation.

IV. s'_a equals the minimal extension of $R(s_a)$ which results from the unification of $R(p_2(r_a+))$ with the subDAG of $R(s_a)$ to which the removed edge was attached.

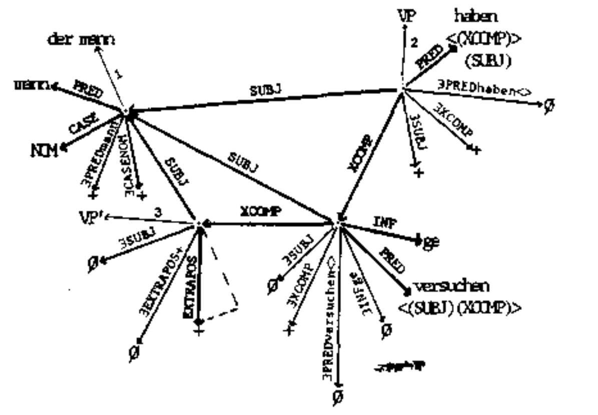
The S-rule is applicable to the start edge of ^s_s (I.), because s is an extension of the partial f-structure underlying "the structure introduced by the rule (II.; indicated by bold edges) and s satisfies the constraint set of the rule (III.). The new DAG in this case simply equals the unification of the structure introduced by the rule with s^o_a after having removed the start edge of s^o_a .



c-structure above)



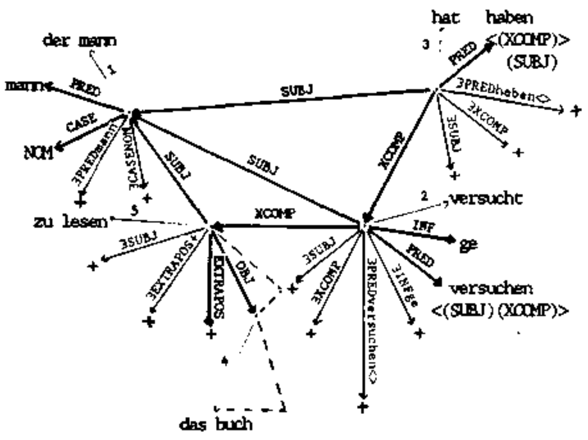
to the <1,NP> edge results in the following structure:



To define the generability relation, we have to take into account that COMpleteness is ensured iff no attribute of the derived structure and its substructures is empty-set-valued.

A terminal string x is generable from a structure f iff x is the string of a structure d, which is generable from the start structure f^o_a , and the feature values of each substructure of d are ^{a}non-empty.

A structure with a terminal string meeting the COM-condition which is derived from s^o_a is then [fn.17]



FOOTNOTES

[*] This paper is based on research sponsored by the BUNDESMINISTER FÜR FORSCHUNG UND TECHNOLOGIE (BMFT), grant no. 1013207 0.
For helpful discussion and criticism we would like to thank Ulrich BLOCK, Gosse BOUMA, Werner FREY, Stefan MOMMA and Stanley PETERS.

[fn.l] In case this hypothesis should be falsified, there are algorithms to construct from f-structures semantic representations (DRSs) that could be used as additional source for the transfer, (cf. [5], [15])

[fn.2] For a more detailed description of the compiler see [3], [4].

[fn.3] For more detailed descriptions and discussions see [6], [7], [13], [14].

[fn.4] We prefer C over Comp in this case to avoid any confusion with the function name COMP. C may actually still be a rather misleading name, suggesting that this position always has to contain a complementizer or conjunction. Unfortunately German is a rather exotic language in this respect, exhibiting a complementary distribution of the finite verb and genuine complementizers in this position. Other names have been suggested like CONFL comprising COMP and INFL, however as this notational idiosyncrasy has little consequences on the actual f-structures we will just stick to C for the moment, i.e. it should be kept in mind that we do not consider C as a lexical category but rather as a structural position.

[fn.5] The format of the rules given below follows basically the notation in [2] with some minor differences: {/ a / b /.../} indicates a disjunction of (sets of) functional schemata a, b, . . . ; square brackets [ ] are used to notate optionality of a constituent.

[fn.6] We carefully note, but ignore for the moment, that genitive phrases in prespecifying position do not always fulfill a POSSessive function. It should also be mentioned that POSS is a function that has to be covered by the extended coherence condition.

[fn.7] V in fact should be seen as a way of expressing that verbs may be complex lexical categories. Another way of expressing this, would be to treat all verbforms as V and the complex verbs as being built up by a Pref + sublexical category V-l,

[fn.8] Of course one could take an entirely different approach and try to derive all prefix verbs on the syntactic level. There are several arguments against this: T. HOEHLE has pointed out that there are some complex forms like 'urauffuehren', 'auferstehen' which clearly follow the pattern of separable prefix verbs, but are very resistant to being broken up into PREF and V: They are perfectly fine when inserted in a clause final position, but won't agree to an analysis into PREF and V which would be the prerequisite for a verb-initial construction:

(1) als sie das Drama urauffuehrten, ...
(2) weil sie das Drama uraufzufuehren haben, .. .
(3) ?/*Sie fuehrten das Drama urauf
(4) *Sie auffuehrten das Drama ur

Similarly (cf [8]) it seems, that non-finite prefix verbs behave as if they were one single constituent on the syntactic level. Although the 'Vorfeld' position in V-second clauses may be taken by simple non-finite verbs, this verb must not be fronted, if it leaves a prefix behind:

(5) Küssen will er die Studentin.
(6) Darstellen will er das Problem.
(7) *Stellen will er das Problem dar.
(8) *Dar will er das Problem stellen.

[fn.9] This is mainly due to our treatment of extraposition, which we do not want to elaborate upon here.

[fn.10] The provability of this statement implies that we departed from a strong adequacy condition for the generator. Thus the structural transfer can be built up more universally, i.e. for classes of languages rather than for a given target language. Suppose one has (for example) structural transfer rules to capture morphological as well as periphrastical tenses, then a wrong choice among the transfer rules causes backtracking since the input structure of the generator is not well-formed.

[fn.ll] The examples are from the grammar presented in [11].

[fn.12] The augmented rules are actually triples consisting of a variable, an augmented structure and a set of constraints. The treatment of the constraints will be dealt with only with respect to generation. The treatment of sets is omitted here for the sake of simplicity.

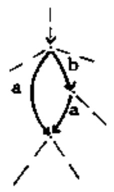
[fn.13] Since even in a graph theoretical reconstruction the atomic feature-values and symbols of the vocabulary have not to be represented as labelled terminal nodes, the feature and integer labelled edges are (together with their values) conceived as ordered pairs.

[fn.14] As these conditions are in some sense extended completeness and coherence conditions, normally expressed by a predicate's subcategorization frame, they are named identically here.

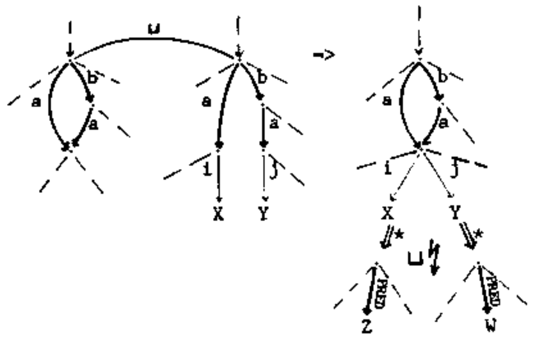
[fn.15] There are several possibilities to encode the constraints in such a way that the third rule component becomes superfluous. Since this rather complicates an informal description of the basic principles, we dispense with this here (cf. [18]).

[fn.16] For more efficient methods cf [18].

[fn.17] If two paths of the input f-structure converge on the same node



they have to be generated in exactly the same way by (a control schema) of a rule. This is a consequence of the fact, that on such paths only functions can occur, whose f-structures contain PRED features. If the generation rules produced these paths leading to different nodes and if the nodes were identified by the unification with the input structure, then the generation would fail as soon as the PRED's for the f-structures designated by the two paths are generated, because two instances of the feature PRED are not unifiable.



BIBLIOGRAPHY

[1] BRESNAN, J. (1982a), Control and Complementation. In: [2] 282-390

[2] BRESNAN, J. (ed.) (1982b), The Mental Representation of Grammatical Relations. Cambridge, Mass.

[3] EISELE, A. (1985), A Lexical Functional Grammar System in Prolog, in: LDV-Forum 2/85.

[4] EISELE, A. / J. DOERRE (1986), A Lexical functional Grammar System in Prolog, in: Proceedings of COLING 86

[5] FREY, W. (1985) Syntax and Semantics of Noun Phrases. In: J. LAUBSCH (ed.), Proceedings of GWAI 1984. Berlin/Heidelberg

[6] HAIDER, H. (1984), Topic, Focus and V-Second. In: GAGL 25

[7] HAIDER, H. (1986), Verb second in German. In: HAIDER, H. / PRINZHORN, M. (eds.), Verb Second. Dordrecht.

[8] HOEHLE, T. (1985), On Composition and Derivation, in: J. TOMAN (ed.), Studies in German Grammar. Dordrecht, 319-376

[9] KAPLAN, R. / J.BRESNAN (1982), Lexical Functional Grammar. A Formal System for Grammatical Representation. In: [2] 173-281

[10] KAPLAN, R. /A. ZAENEN (1985). Functional Uncertainty in LFG. Ms. Stanford

[11] NETTER, K. (1986a), Getting Things Out of Order, in: Proceedings of COLING 86

[12] NETTER, K. (1986b), Auxiliarkomplex und Wortstellung im Deutschen. To appear in: U. KLENK / P. SCHERBER / M. THALLER (eds.), Akten der GLDV - Jahrestagung (= Linguistische Datenverarbeitung Vol. 6). Hildesheim

[13] REIS, M. (1980), On Justifying Topological Frames. In: DRLAV, Revue de linguistic, no. 22/23, 59-95

[14] REIS, M. (1985), Satzeinleitende Strukturen im Deutschen. Über COMP, Haupt- und Nebensätze, w-Bewegung und die Doppelkopfanalyse. In: W. ABRAHAM (ed.) Erklärende Syntax des Deutschen. Tübingen, 271-313

[15] REYLE, U. (1985), Grammatical Functions, Discourse Referents and Quantification. Proceedings of IJCAI 85

[16] WEDEKIND, J. (1986a), Ein Vorschlag zur nicht-konfigurationalen Kodierung grammatischer Funktionen in der LFG. in: Forum angewandte Linguistik, vol. 14, Tuebingen

[17] WEDEKIND, J. (1986b), A Concept of Derivation for LFG. in: Proceedings of COLING 86

[18] WEDEKIND, J. (1986c), Derivation and Generation in LFG. Ms. Stuttgart