Practical Instructions for Working with the Formalism of Lexical Functional Grammar*

Michael T. Wescoat

0. Introduction

This paper is intended to provide a set of step by step instructions for using the formalism of Lexical Functional Grammar (henceforth LFG). With the information contained in this paper, linguists previously unfamiliar with the formalism of this theory should find it possible to interpret and to compose the sorts of rules and lexical items standardly employed in LFG. These instructions will demonstrate how to build ANNOTATED CONSTITUENT STRUCTURES (c-structures) and FUNCTIONAL STRUCTURES (f-structures)—the two levels of representation that LFG assigns to sentences. [1] C-structure indicates the hierarchical composition of words into larger units or phrasal constituents, while f-structure is a representation of grammatical functions like subject, object, etc.

The reader is warned at the outset that we do not intend to present any truly linguistic claims in this text, that is, claims about the structure and workings of natural language. We wish rather to provide linguists with some knowledge of a formal system which may in turn be employed to express such claims. For this reason the rules and lexical entries employed are quite basic. Nonetheless, a maximally simple example grammar will hopefully enhance the readability of this document, and the technical expertise which the reader will gain will facilitate comprehension of more insightful analyses expressed by means of the LFG formalism. [2]

The paper will take the following form. First (Sec. 1.) we will display some of the basic notations employed in published works on LFG. Next (Sec. 2) we will describe how it is that one creates an annotated constituent structure. The reason that we use the word *annotated* here is that the nodes in the tree are marked with certain expressions or *annotations*. These annotations are interpretable, once they are filled out with some information derived from the tree structure. This process of filling in information is known as *instantiation*, and the method of instantiating annotations in the tree is described in Section 3. The result of instantiation is a set of expressions, known as the FUNCTIONAL DESCRIPTION, which fully specifies how we are to build the corresponding f-structure (Sec. 4). In Section 5 we discuss the nature of f-structures and the procedures for creating them from the functional description. With the ability to build f-structures we may then shift (Sec. 6) to more linguistic issues concerning the role of f-structures in predicting the well-formedness of natural language sentences. In Section 7 some residual formal issues are taken up: these concern special types of functional equations.

1. The Notational Conventions of LFG

Below are discussed some of the notations that a person is likely to encounter in a grammar written in LFG.

1.1. The Form of Syntactic Rules in LFG. In outward form, rules in LFG resemble the context free rules of the base component of a transformational grammar in Standard Theory.[3,4] The rules of a Lexical Functional Grammar, however, contain expressions known as FUNCTIONAL SCHEMATA, which are associated with the symbols that appear on the right hand side of the $\rightarrow$ arrow. The following figure shows the usual format for writing rules in LFG.

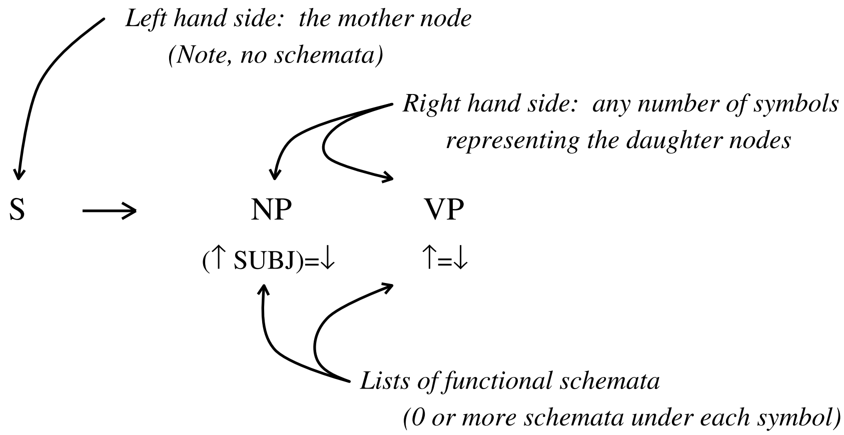


FIGURE I: FORMAT OF LFG RULES

The following rules may be combined with a lexicon to generate a small fragment of English.

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

(2) $VP \rightarrow \underset{\uparrow=\downarrow}{\text{V}}\ \left(\underset{(\uparrow\ \text{OBJ})=\downarrow}{\text{NP}}\right)$

(3) $NP \rightarrow \left(\underset{\uparrow=\downarrow}{\text{DET}}\right)\ \underset{\uparrow=\downarrow}{\text{N}}$

The expressions (↑ SUBJ)=↓, ↑=↓, and (↑ OBJ)=↓ are all functional schemata. We will discuss in Section 2 how one interprets these rules in order to construct annotated c-structures.

1.2. The Form of Lexical Items in LFG. Like the rules just described, lexical items are also supplied with functional schemata in LFG. In most works dealing with LFG, we find lexical entries that contain at least three things: [5] a representation of the form of the item, [6] the syntactic category [7] to which the item belongs, and a list of functional schemata. [8] They are usually written out as schematized below.

Representation of item

Syntactic category

John N ($\uparrow$ PRED)='JOHN'
($\uparrow$ PERS)=3
($\uparrow$ NUM)=SING

List of functional schemata

FIGURE III: FORMAT OF LFG LEXICAL ITEMS

The following three lexical items may be combined with the rules from Subsection 1.1 to generate a couple of English sentences.

(4) John N ($\uparrow$ PRED)='JOHN'
($\uparrow$ NUM)=SING
($\uparrow$ PERS)=3

(5) sees V ($\uparrow$ PRED)='SEE<($\uparrow$ SUBJ) ($\uparrow$ OBJ)>'
($\uparrow$ SUBJ NUM)= SING
($\uparrow$ SUBJ PERS)= 3

(6) Mary N ($\uparrow$ PRED)='MARY'
($\uparrow$NUM)=SING
($\uparrow$ PERS)=3

1.3. A Caution . We should point out at this juncture that researchers working within LFG have invented many abbreviatory devices to emphasize systematic aspects of word, phrase, and functional structure, and linguists naturally prefer these, because they provide perspicuous encodings of generalizations about natural language. However, in this document we shall discuss only the most fundamental conventions employed in LFG, which, happily, are basic to all writings in the theory.

2. Creating Annotated Constituent Structures

In the present section we shall construct the annotated constituent structure tree for the sentence in (7).

(7) John sees Mary.

The c-structure representation is a source of two types of information. First it indicates the hierarchical structure of phrasal constituents in the string in a fashion familiar to most any linguist. More importantly for our purposes, the so-called FUNCTIONAL ANNOTATIONS (functional schemata transferred into the tree) may be interpreted to derive information about the functional structure.

Because context free phrase structure rules and phrase structure trees are generally familiar objects, creating an annotated c-structure will be a very simple matter. The only additional task is the insertion of the functional schemata; however, this too is quite uncomplicated.

First consider the syntactic rules. When a rule is applied, a piece of the tree is constructed and the annotations prescribed by the rule are written in above the appropriate nodes in the fashion schematized in Figure III.

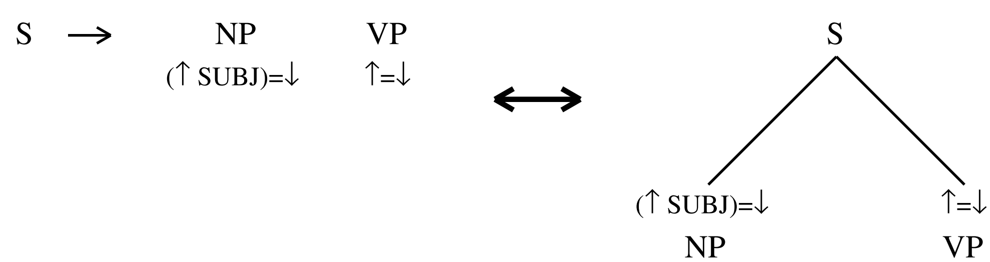
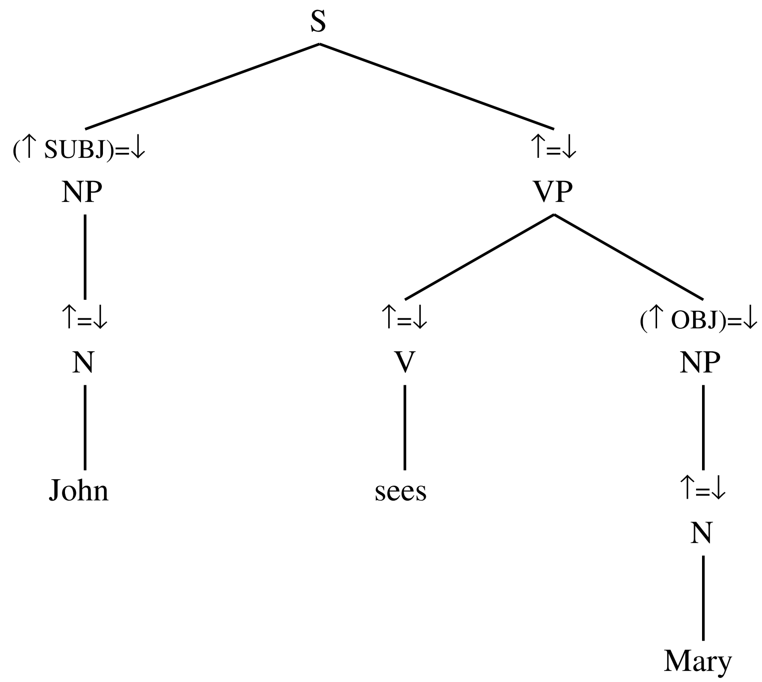


FIGURE III: RELATION BETWEEN RULES AND ANNOTATIONS IN THE TREE

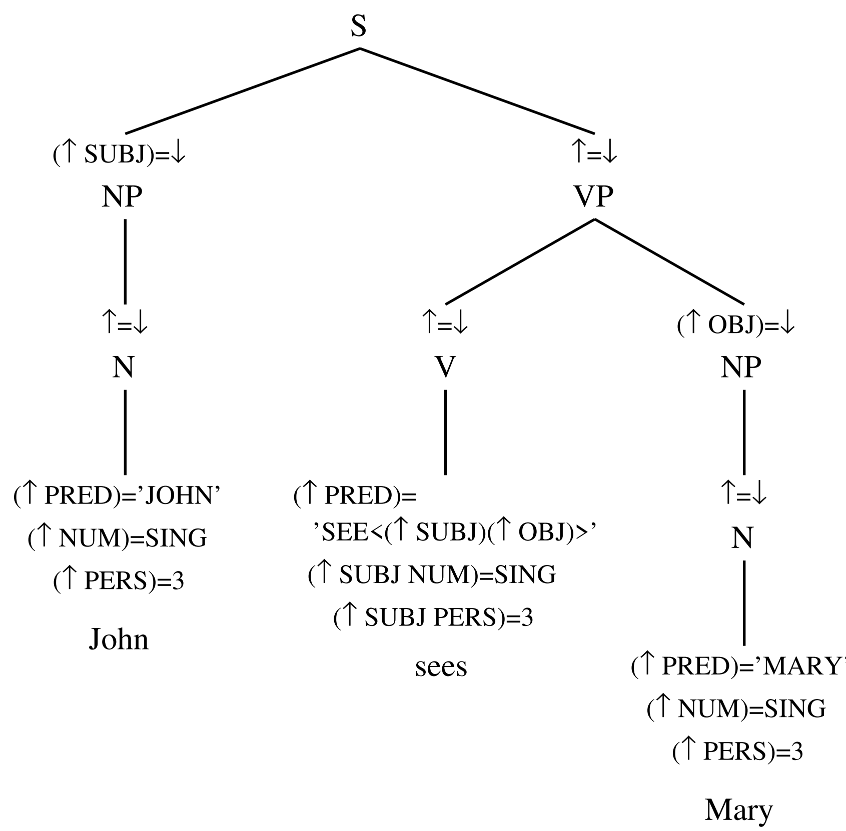
Thus, the rules in (1)-(3) predict that (7) will be dominated by the tree in (8).

(8)



This represents, though, only a stage in building the annotated tree for (7); the c-structure is complete only after introducing the annotations specified by the lexical entries for *John*, *sees*, and *Mary*. Hence, for each lexical item in the tree, we consult the corresponding lexical entry and copy all the functional schemata found in that entry into the tree above the appropriate word. Figure (9) gives the complete annotated constituent structure tree for (7).

(9)



Now that the functional schemata have been transferred into the tree, they are ready to be interpreted. As we shall see in Section 3, it is their place within the tree that eventually provides these schemata with a meaning.

3. Instantiation

From the annotated c-structures discussed in the previous section one may build f-structures, the other level of syntactic representation employed in Lexical Functional Grammars. The ↑ and ↓ arrows employed in the schemata assume a referential value now that the schemata have been placed into the tree: the reader will see that these arrow characters were chosen for their symbolic value, since they *point*, in a manner of speaking, to things above and below the location of the schema in the tree. Determining the referents of the situated ↑ and ↓ arrows and recording this information in the schemata is known as INSTANTIATION. Instantiation transforms the schemata into FUNCTIONAL EQUATIONS, fully specified expressions in a formal language used to talk about f-structures. In Subsection 3.1 we discuss the relation between nodes in the c-structure tree and f-structures. With this knowledge in hand, we may then move on to the the matter of finding the referents for the $\uparrow$ and $\downarrow$ arrows.

In this section we will be discussing f-structures without being concerned for the moment with their contents. In other words we shall be thinking in terms of *some potential f-structure*, and the reader should not feel ill at ease if s/he doesn't have any notion yet about what would go into an f-structure. Graphically f-structures are represented in the literature as material enclosed in large square brackets, and for now we shall use empty pairs of square brackets in figures to represent our abstract f-structures.

3.1. The Relation Between C-Structure Nodes and F-Structures. One of the assumptions of LFG is that there is some f-structure associated with each node in the constituent structure tree. Figure IV schematizes how we might conceptualize this relation.

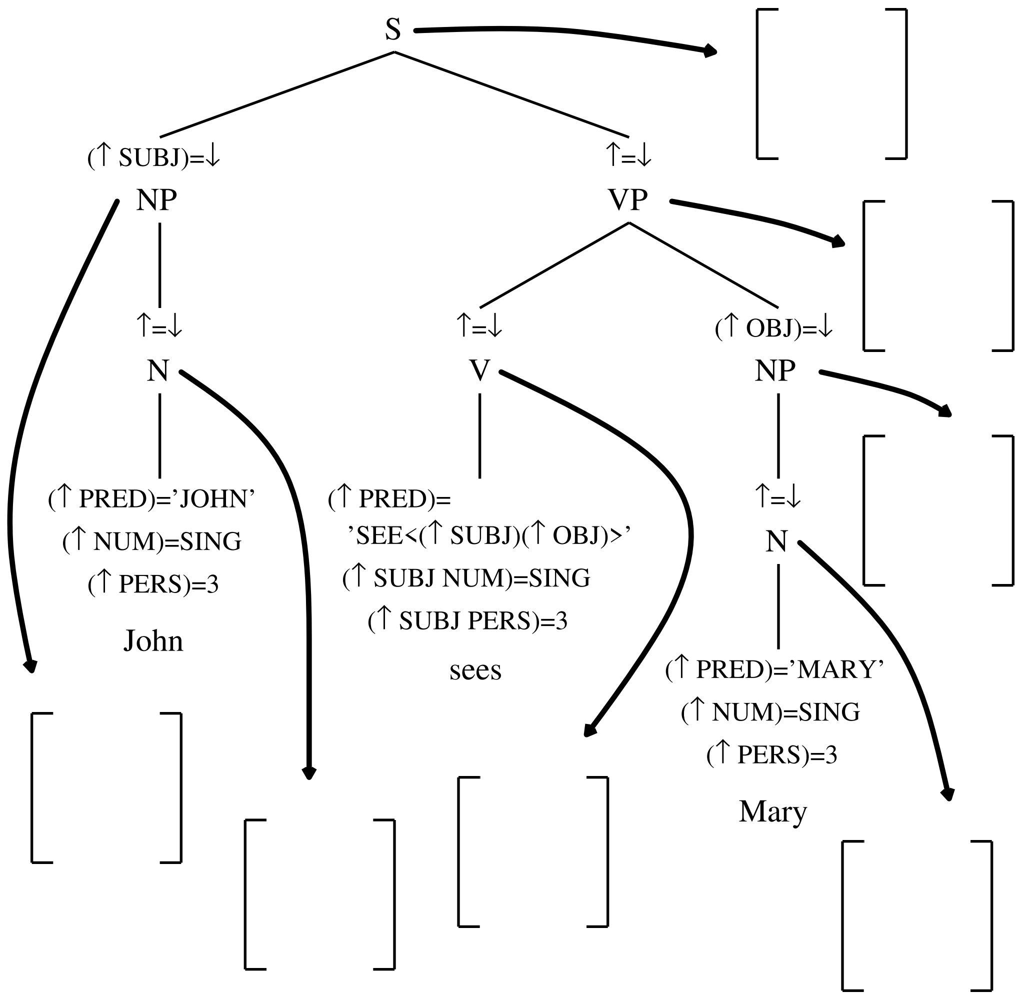


FIGURE IV: RELATION BETWEEN NODES IN THE TREE AND F-STRUCTURES

To facilitate talking about functional structures, we can associate arbitrary names or variables with each f-structure. In order to prevent any possible confusion, we emphasize that the choice of a variable for a given f-structure is *completely arbitrary*. It has become something of a tradition to use f-structure variables that consist of the letter *f* followed by a number, e.g. f_{123}, different numbers making for distinct variables. Graphically we shall write the identifying variable of an f-structure immediately outside that f-structure's left bracket. Furthermore, to express in a compact way the association of an f-structure with a node in the constituent structure tree, we can record the variable of the f-structure next to the node to which it corresponds. This process could be called coindexing the node. Our conceptualization from Figure IV could then be rendered as in Figure V.

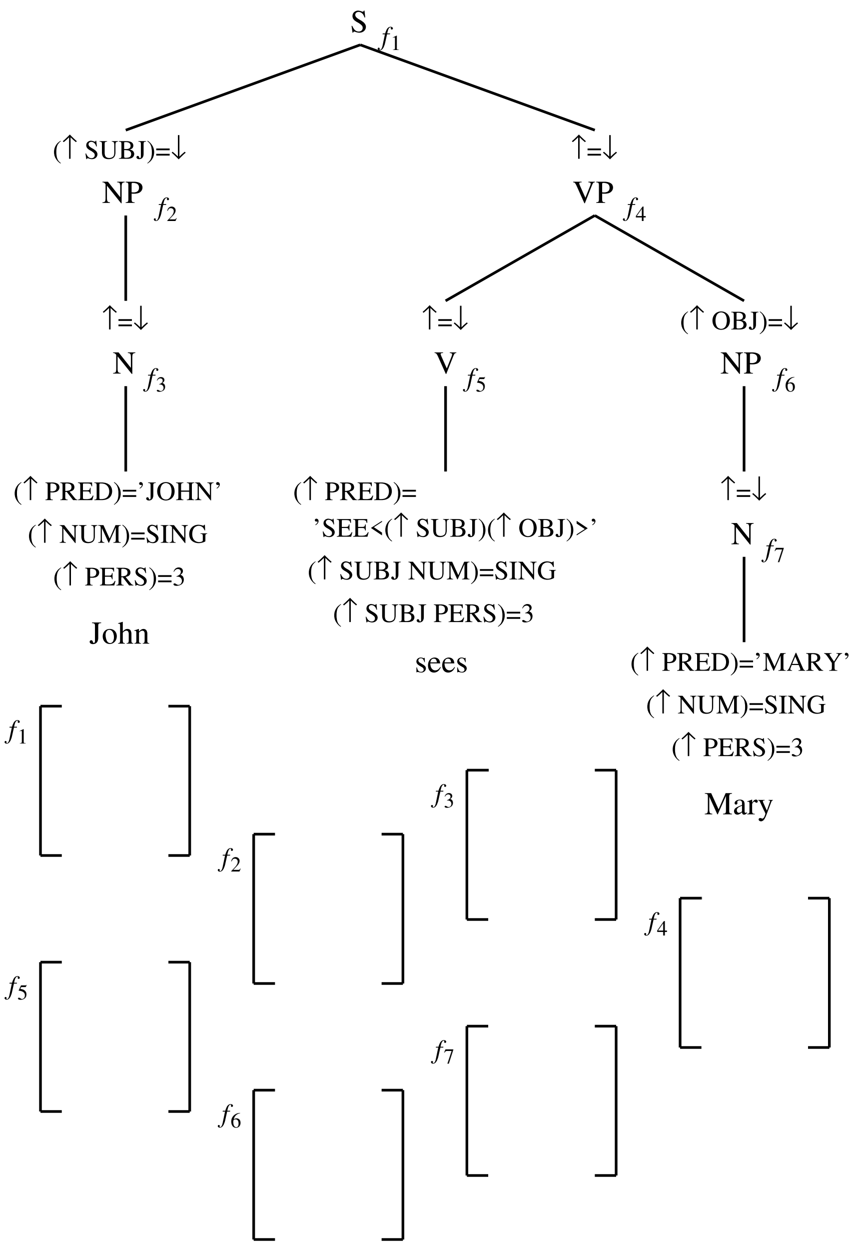


FIGURE V: PROVIDING NAMES FOR F-STRUCTURES AND COINDEXING TREE NODES

3.2. Finding the referents for ↑ and ↓. Now we can begin the process of instantiating the schemata in the tree in order to convert them into fully specified functional equations. The first step here is to determine what it is that the ↑ and ↓ arrows refer to in the annotated tree. In fact these refer to f-structures, and all that needs to be done, then, in order to make these schemata into equations is to insert the variables which name the appropriate f-structures in place of the arrows. Since the ↑ and ↓ arrows are symbols which are replaced by f-structure *variables*, they are often referred to as METAVARIABLES in the literature.

Finding the referents of the ↑ and ↓ arrows is a simple matter. The ↓ is known as the EGO or SELF metavariable: it refers to the f-structure associated with the node above which the schema containing the ↓ appears. The other metavariable, the ↑, is called the MOTHER metavariable: this refers to the f-structure associated with the mother of the node above which the schema containing the ↑ appears. This set of relations is schematized in the following figure, where the ↑ and ↓ arrows are provided with continuations which lead to their respective referents.

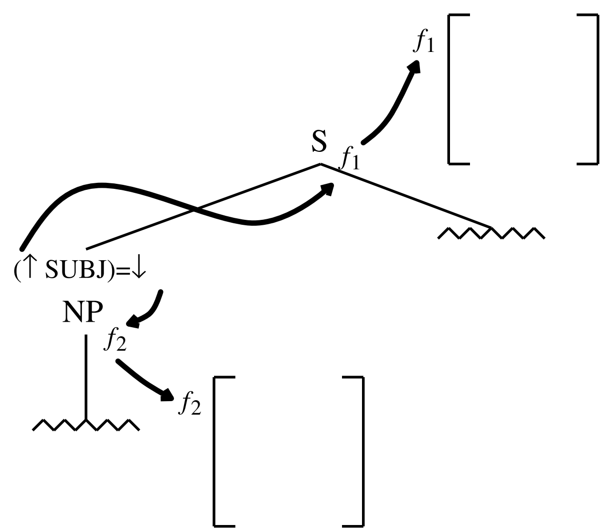


FIGURE VI: DETERMINING REFERENTS FOR THE SELF AND MOTHER METAVARIABLES

Hence, we complete the instantiation process by replacing all metavariables with the names of the f-structures to which they refer, in the fashion schematized in the next figure, which is a modification of figure VI.

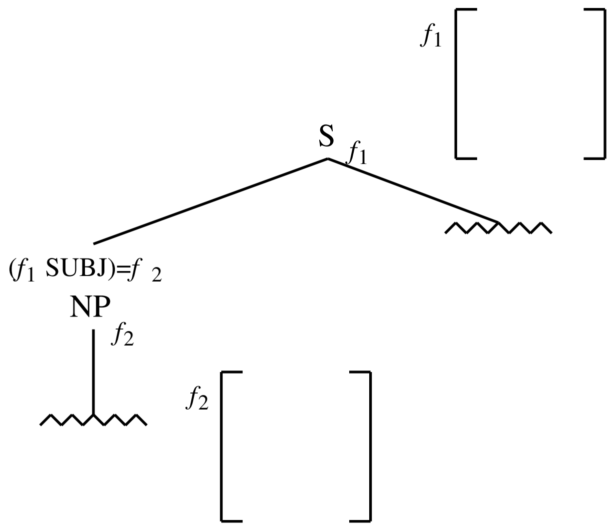
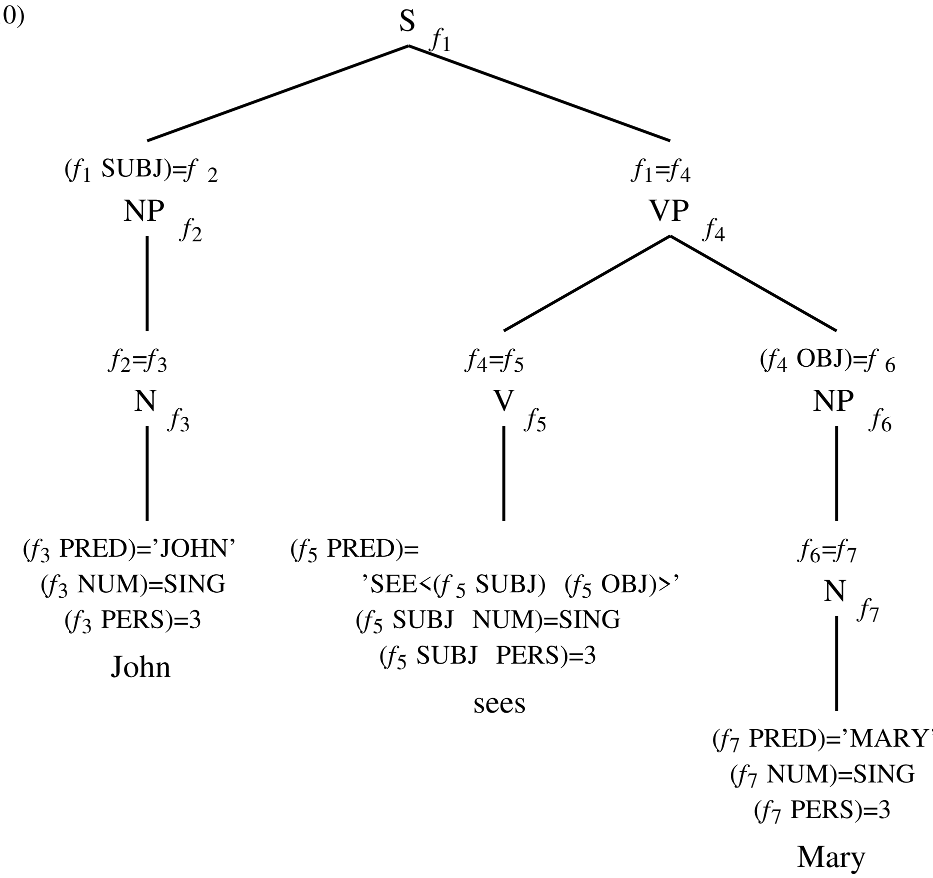


FIGURE VII: COMPLETING INSTANTIATION BY COINDEXING

(10) shows the finished tree for example (7) with instantiated equations.

(10)



3.3. Consolidation. It should be noted that the foregoing sequence of figures with their empty f-structures, pointers, and the like were intended merely as aids to conceptualize how it is that the instantiation process works and what it represents. In practice there is no need to draw complex pictures: the essentials of the algorithm described above may be condensed to the following. Having built up the annotated c-structure ((a) of Figure VIII shows part of such a structure), provide each node in that c-structure with a distinct variable, which will then be taken to name the f-structure for that node (see (b) of Figure VIII). The choice of variables is utterly arbitrary, save only for the restriction that no two nodes have the same variable. With this done, consider each functional schema, replacing all instances of the self metavariable with the variable associated with the node above which the schema appears. Also replace all instances of the mother metavariable with the variable applied to the mother of the node above which the schema appears (see (c) of Figure VIII).

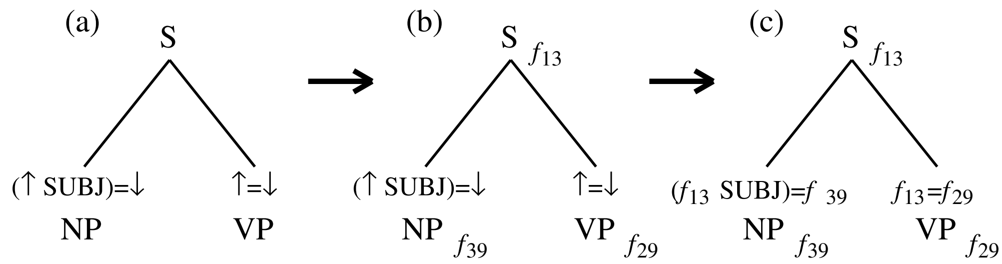


FIGURE VIII: CONCISE INSTANTIATION ALGORITHM

4. The Functional Description

The set of all of the instantiated functional equations in the tree is called the FUNCTIONAL DESCRIPTION. The functional description for (10) is listed below. It is from the functional description that f-structures are constructed, and for the rest of the process of constructing the f-structure the tree is no longer considered. This is an opportune place to issue a warning about the eventual content of the f-structure. A common expectation among people approaching LFG for the first time is that categorial information will be carried over from the tree into the f-structure (for instance that *verb* will somehow be directly mentioned in the f-structure). We stress the fact that only information present in the equations of the functional description will be incorporated into the eventual f-structure.

(11)
a. (f_1 SUBJ)=f_2
b. f_2=f_3
c. (f_3 PRED)='JOHN'
d. (f_3 NUM)=SING
e. (f_3 PERS)=3
f. f_1=f_4
g. f_4=f_5
h. (f_5 PRED)='SEE<(f_5 SUBJ) (f_5 OBJ)>'
i. (f_5 SUBJ NUM)=SING
j. (f_5 SUBJ PERS)=3
k. (f_4 OBJ)=f_6
l. f_6=f_7
m. (f_7 PRED)='MARY'
n. (f_7 NUM)=SING
o. (f_7 PERS)=3

5. F-Structures: Their Nature and Construction

Subsection 3.2.1 discusses some of the characteristics of f-structures. Passing to Subsection 3.2.2, we take up the question of what it is that functional equations mean with regard to the f-structures they describe. Subsection 3.2.3 describes the process of constructing an f-structure, using sentence (7) as an example to clarify many of the more confusing points of this procedure.

5.1. Characteristics of F-Structures.

Some basic observations about f-structures follow.

5.1.1. The Form of F-Structures.

Graphically an f-structure takes the form shown in (12).

(12)
$$f_n \left[\begin{array}{ll} \mathrm{A} & f_m \left[\begin{array}{ll} \mathrm{B} & \mathrm{C} \\ \mathrm{D} & \mathrm{E} \end{array} \right] \\ & \\ \mathrm{F} & \mathrm{G} \\ \mathrm{H} & \mathrm{I} \end{array} \right]$$

Each f-structure consists of two columns of entries enclosed in large square brackets. The left hand column contains what are known as ATTRIBUTES and the other VALUES. Attributes and values are paired, and the members of a pair are written on the same horizontal line. Attributes are always simple symbols, like A, F, SUBJ, PRED, or whatever else the linguist might want to use. Values, however, may be simple symbols, subordinate f-structures, or semantic forms. Semantic Forms are recognizable by the fact that they are always flanked with single quotes; these will be discussed more below.

Outside of the lefthand bracket of an f-structure one may optionally write the name of the f-structure, as we have in (12).

The reader is cautioned that there is absolutely no significance attached to the order in which lines occur in an f-structure. If fact, (13)a and (13)b are two different representations of the very *same* f-structure.

(13) a. $\begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \\ \text{E} & \begin{bmatrix} \text{F} & \text{G} \\ \text{H} & \text{I} \end{bmatrix} \end{bmatrix}$ b. $\begin{bmatrix} \text{C} & \text{D} \\ \text{E} & \begin{bmatrix} \text{H} & \text{I} \\ \text{F} & \text{G} \end{bmatrix} \\ & \\ \text{A} & \text{B} \end{bmatrix}$

5.1.2. The Uniqueness Condition on F-Structures. F-structures must conform to a uniqueness condition: if there is a value, say K, associated with an attribute J in f_p, and the distinct value L is also associated with J in f_p, then the f-structure is ill-formed.[9]

(14) $f_p \begin{bmatrix} \ldots & \ldots \\ \text{J} & \text{K} \\ \text{J} & \text{L} \\ \ldots & \ldots \end{bmatrix}$

5.2. The Meaning of Functional Equations. As has already been stated, functional equations are meaningful expressions in a formal language: they convey information about f-structures. Having gained a rudimentary knowledge of the form of f-structures, we are now in a position to discuss what functional equations express, that is, to provide a semantics for this formal language. Figure IX below provides a scheme for interpreting functional equations.

(f_p ATT)=VAL

In the f-structure f_p there is a line where,
the attribute is ATT,
and the value is VAL.

FIGURE IX: MEANING OF FUNCTIONAL EQUATIONS

Let us take a concrete example, say the equation in (15).

(15) (f_n F)=G

According to this equation, the f-structure f_n contains a line where F is the attribute, and G is the value. Let the f-structure pictured in (12) (repeated here for convenience) be f_n.

(12)
$$f_n \left[\begin{array}{ll} \mathrm{A} & f_m \left[\begin{array}{ll} \mathrm{B} & \mathrm{C} \\ \mathrm{D} & \mathrm{E} \end{array}\right] \\ \mathrm{F} & \mathrm{G} \\ \mathrm{H} & \mathrm{I} \end{array}\right]$$

Now compare the above equation with f_n (i.e. (12)). We see that indeed there is a line in f_n (the second up from the bottom) where F is the attribute and G is the value. Hence, for our f_n, the equation in (15) holds true. Now consider another equation, assuming that our f_n is still (12).

(16) $(f_n\ \mathrm{Q})=\mathrm{R}$

Since there is no line in f_n where the attribute is Q and the value is R, (16) is false. In (17) we enumerate five equations that hold true of (12). Note that the first of these says that the f-structure f_m is the value of A in f_n and that the second two say something about the contents of f_m.

(17) $(f_n\ \mathrm{A})=f_m$ $(f_m\ \mathrm{B})=\mathrm{C}$ $(f_m\ \mathrm{D})=\mathrm{E}$ $(f_n\ \mathrm{F})=\mathrm{G}$ $(f_n\ \mathrm{H})=\mathrm{I}$

5.3. The Construction of F-Structures. If one understands what the equations of a functional description are saying about a given f-structure, one can certainly draw that f-structure: the goal is to put together an f-structure in such a way that all of the functional equations contained in the functional description are true. For instance, take (18) as an example functional description.

(18) $(f_r\ \mathrm{A})=\mathrm{B}$
$(f_r\ \mathrm{C})=\mathrm{D}$

Let us take the first of these two equations and consider what it would take to make it true. There would have to be an f-structure called f_r, and it would have to contain a line where A is the attribute and B is the value. Now if we know that the f-structure has these characteristics, we can go ahead and start building it, inserting all of the features that we just determined it must have.

(19) $$f_r \begin{bmatrix} \text{A} & \text{B} \end{bmatrix}$$

Now we move on to the next equation, which requires the following in order to be true: f_r must have a line where C is the attribute, and D is the value. With this knowledge we can modify the representation in (19) to reflect the new information.

(20) $$f_r \begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \end{bmatrix}$$

5.3.1. Minimality. In (20) we have the f-structure called f_r which makes both of the equations in (18) true. However, it is important to note that it is the MINIMAL f-structure which is desired. We may best discuss minimality in reference to an example. Let us consider the functional description in (18) in relation to the two f-structures in (21).

(21) a. $$f_r \begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \end{bmatrix}$$ b. $$f_r \begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \\ \text{E} & \text{F} \end{bmatrix}$$

The f-structure in (21)a is the same one that we have already seen, and we have already observed that it makes both of the equations in (18) true. Notice too that both of the equations in (18) are true of the f-structure in (21)b also. Now observe that if we take away either of the two lines that compose (21)a, one or the other of the equations in (18) will cease to be true. When all of the functional equations in a functional description are true of a given f-structure, and removing any line out of that f-structure would invalidate some equation in the functional description, then and only then is that f-structure considered the *minimal* f-structure for the functional description in question. Consider now (21)b. As we noted, both equations in (18) are true of (21)b as it is displayed above. However, we could take away a line from (21)b (specifically the last line), and both equations would remain true of the resultant f-structure. Thus, (21)b is *not* the minimal f-structure for the functional description in (18).

5.3.2. Construction of an Example F-Structure. At this point we can proceed with constructing the f-structure for (7). Doing so will highlight many of the potentially tricky aspects of building f-structures according to the specifications of functional equations.

5.3.2.1. Simple Equations. There is a large group of functional equations in (11) that we can handle easily, given the foregoing discussion, that is, the discussion concerning examples (18)-(20).

(22)
c. $(f_3$ PRED)='JOHN'
d. $(f_3$ NUM)=SING
e. $(f_3$ PERS)=3
h. $(f_5$ PRED)='SEE<$(f_5$ SUBJ) $(f_5$ OBJ)>'
m. $(f_7$ PRED)='MARY'
n. $(f_7$ NUM)=SING
o. $(f_7$ PERS)=3

There are three f-structure names used in the equations in (22), so let us keep a record of all three structures, drawing them separately. They are pictured in (23).

(23)

$$f_5 \begin{bmatrix} \text{PRED} & \text{'SEE<}(f_5\ \text{SUBJ})\ (f_5\ \text{OBJ})\text{>'} \end{bmatrix}$$

$$f_3 \begin{bmatrix} \text{PRED} & \text{'JOHN'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{bmatrix} \qquad f_7 \begin{bmatrix} \text{PRED} & \text{'MARY'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{bmatrix}$$

5.3.2.2. Equations of the Form f_m=f_n. Let us now consider all the equations from (11) which take of the form f_m=f_n, i.e., those equations listed in (24).

(24)
b. f_2=f_3
f. f_1=f_4
g. f_4=f_5
l. f_6=f_7

All of these statements equate f-structures: they say that the two variables mentioned in the equation actually name the same f-structure. In another manner of speaking, we might say that the two equated variables are alternative labels for one f-structure. In graphic conventions, equivalent names for an f-structure are written one on top of the other on the left side of that f-structure. Let's consider the example functional description in (25).

(25) $(f_s\ \mathrm{A}){=}\mathrm{B}$
$(f_t\ \mathrm{C}){=}\mathrm{D}$
$f_s{=}f_t$

Considering the first equation, we can easily take the first step in building an f-structure to satisfy this functional structure, building up the representation in (26).

(26) $$f_s \begin{bmatrix} \mathrm{A} & \mathrm{B} \end{bmatrix}$$

Now consider the second equation, which says that there is an f-structure called f_t which contains a line where C is the attribute and D is the value. But the final equation indicates that f_s and f_t name the same f-structure, a fact which leads us to infer that the f-structure in (26), known alternatively as f_s or f_t, should have not only the characteristics specified for f_s but also those specified for f_t. Hence, we add to (26) the line which the second equation prescribed for f_t. We can record the fact that (26) has two names, f_s and f_t, by writing both of these outside of the left bracket of the f-structure. These modifications are made in (27).

(27) $$\begin{matrix} f_s \\ f_t \end{matrix} \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{C} & \mathrm{D} \end{bmatrix}$$

There is another conceptual issue which the reader should grasp with regard to the equations in (24). Recall Figure IV, which was a graphic conceptualization of the association of c-structure nodes with f-structures. Each node was paired with a distinct f-structure icon. This representation was an abstraction which begged the question of whether or not each one of those icons represented a distinct f-structure. That would be an entirely possible state of affairs; however, it need not be the case that nodes are paired one to one with f-structures. Obviously the equations in (24) are telling us that in our particular example it is not true that each node is paired with a distinct f-structure. Modifying Figure IV to be somewhat less abstract with regard to this matter, we could draw Figure X. It is worth studying the relation between the nodes and f-structures in Figure X, in order to better understand how it is that functional information from several c-structure nodes may be pooled together in f-structure.

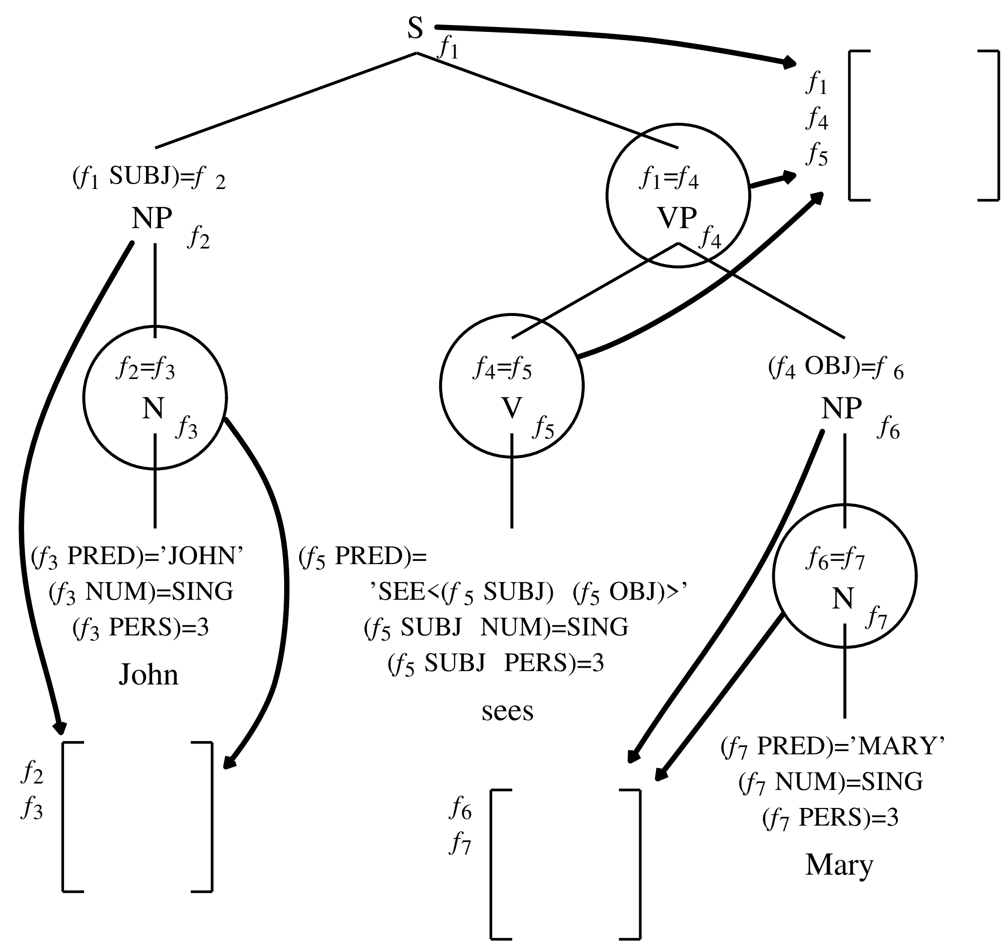


FIGURE X: ASSOCIATION OF F-STRUCTURES AND NODES REVISED

Getting back to the matter of building the f-structure for the functional description in (11), we can record all of the information about f-structure name equivalencies by listing the complete set of names on all f-structures, as in (28) (c.f. (26)).

(28)

$$\begin{array}{l} f_1 \\ f_4 \\ f_5 \end{array} \left[\begin{array}{ll} \text{PRED} & \text{'SEE}\langle (f_5\ \text{SUBJ})\ \ (f_5\ \text{OBJ}) \rangle\text{'} \\ & \\ & \end{array} \right]$$

$$\begin{array}{l} f_2 \\ f_3 \end{array} \left[\begin{array}{ll} \text{PRED} & \text{'JOHN'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array} \right] \qquad \begin{array}{l} f_6 \\ f_7 \end{array} \left[\begin{array}{ll} \text{PRED} & \text{'MARY'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array} \right]$$

5.3.2.3. Left Associativity. Some equations in (11)—specifically those in (29)—feature two attributes.

(29) i. $(f_5$ SUBJ NUM)=SING
j. $(f_5$ SUBJ PERS)=3

Let us take (29)i as the object of our discussion. To interpret this equation one needs to realize that the form of the equation seen above employs an abbreviatory convention called LEFT ASSOCIATIVITY : the unabbreviated version of (29)i would be that in (30).

(30) $((f_5$ SUBJ) NUM)=SING

Now let us simply apply the interpretation schema for functional equations (Subsection 5.2) to (30).

$((f_5$ SUBJ) NUM)=SING

In the f-structure $(f_5$ SUBJ) *there is a line where*
the attribute is NUM,
and the value is SING.

FIGURE XI: MEANING OF (30)

So the expression (f_5 SUBJ) designates an f-structure, as we can infer from the meaning of (30) as it was deciphered in Figure XI. In other words, the f-structure f_5 should contain a line in it where the attribute is SUBJ. Furthermore, the value found in that line should be an f-structure, and that f-structure will have the characteristics detailed in Figure XI. So the modification we must make the f-structures we have been building up in (28) is seen in (31).

(31)

$$\begin{array}{l} f_1 \\ f_4 \\ f_5 \end{array}\left[\begin{array}{ll} \text{PRED} & \text{'SEE<}(f_5\ \text{SUBJ})\ \ (f_5\ \text{OBJ})\text{>'} \\ \text{SUBJ} & \left[\begin{array}{ll}\text{NUM} & \text{SING}\end{array}\right] \\ & \end{array}\right]$$

$$\begin{array}{l} f_2 \\ f_3 \\ \\ \end{array}\left[\begin{array}{ll} \text{PRED} & \text{'JOHN'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \qquad \begin{array}{l} f_6 \\ f_7 \\ \\ \end{array}\left[\begin{array}{ll} \text{PRED} & \text{'MARY'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right]$$

This is an opportune place to make a point about expressions of the form (f_n ATT), such as (f_5 SUBJ). Recall the fact that f-structures are subject to a uniqueness constraint such that an f-structure may not contain two lines having the same attribute but different values (Subsection 5.1.2). As a consequence of this restriction, a given value can be uniquely identified by mentioning the f-structure in which it is found and the attribute with which it is paired. Thus, an expression of the form (f_p ATT) uniquely names something; i.e., the value paired with ATT in the f-structure f_p. Hence, (f_5 SUBJ) names or designates uniquely the f-structure that we added to f_5 in (31), and any subsequent use of the expression (f_5 SUBJ) will refer to that same f-structure.

Now we can move to the next equation in (29), reproduced as (32).

(32) i. (f_5 SUBJ NUM)=SING
j. (f_5 SUBJ PERS)=3

Applying the same process as we did in dealing with the previous equation we write out (32) in the unabbreviated form in (33).

(33) ((f_5 SUBJ) PERS)=SING

Now we know that the (f_5 SUBJ) designates a particular f-structure, and we simply add to that f-structure a line with PERS as attribute and 3 as value, as in (34).

(34)
$$\begin{array}{l}
\begin{array}{l} f_1 \\ f_4 \\ f_5 \end{array}
\left[\begin{array}{ll}
\text{PRED} & \text{'SEE<}(f_5\ \text{SUBJ})\ \ (f_5\ \text{OBJ})\text{>'} \\
\text{SUBJ} & \left[\begin{array}{ll} \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \\
 &
\end{array}\right] \\[1em]
\begin{array}{l} f_2 \\ f_3 \end{array}
\left[\begin{array}{ll}
\text{PRED} & \text{'JOHN'} \\
\text{NUM} & \text{SING} \\
\text{PERS} & 3
\end{array}\right]
\qquad
\begin{array}{l} f_6 \\ f_7 \end{array}
\left[\begin{array}{ll}
\text{PRED} & \text{'MARY'} \\
\text{NUM} & \text{SING} \\
\text{PERS} & 3
\end{array}\right]
\end{array}$$

Now we have only two equations left to examine in the functional description for sentence (7). These are listed below.

(35) a. $(f_1\ \text{SUBJ})=f_2$
b. $(f_4\ \text{OBJ})=f_6$

The second of these equations can be gotten out of the way rapidly and easily. This equation says that the f-structure f_4 contains a line where the attribute OBJ is paired with the f-structure f_6. Hence, we simply place f_6, which we have been building up, inside of f_4 beside the attribute OBJ in the manner illustrated below.

(36)
$$\begin{array}{l}
\begin{array}{l} f_1 \\ f_4 \\ f_5 \end{array}
\left[\begin{array}{lll}
\text{PRED} & \multicolumn{2}{l}{\text{'SEE<}(f_5\ \text{SUBJ})\ \ (f_5\ \text{OBJ})\text{>'}} \\
\text{SUBJ} & & \left[\begin{array}{ll} \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \\
\text{OBJ} & \begin{array}{l} f_6 \\ f_7 \end{array} & \left[\begin{array}{ll} \text{PRED} & \text{'MARY'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right]
\end{array}\right] \\[1em]
\begin{array}{l} f_2 \\ f_3 \end{array}
\left[\begin{array}{ll}
\text{PRED} & \text{'JOHN'} \\
\text{NUM} & \text{SING} \\
\text{PERS} & 3
\end{array}\right]
\end{array}$$

The last equation says that the value paired with SUBJ in f_1 is f_2. However, this time there is a pre-existing value for SUBJ in f_1, that is the f-structure pictured in (37).

(37) $\begin{bmatrix} \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{bmatrix}$

This means that the f-structure in (37) and f_2 are in fact (partial) representations of the same thing. In this case the value of SUBJ is the the MERGER of the two f-structures.

MERGER:

The merger of two instances of the same atomic name consists of that atomic name. Atomic names which are not identical do not merge. Semantic forms (flanked by '...') *never* merge. To effect the merger of two f-structures—let us call these f_m and f_n—we select one of these f-structures, say f_m, and for each attribute a in f_m, we attempt to find an instance of a in f_n. Let us call the value associated with a in f_m v. If a does not occur in f_n, then we add the attribute a and the value v to f_n. Contrarily, if a is already present in f_n, and its value is v', then the merger of v and v' becomes the new value of a in f_n. If all of the subsidiary mergers are successful, then the modified version of f_n represents the merger of f_m and f_n.

If the merger of these two f-structures were to fail, then there would be no valid f-structure for the f-description in (11), and the grammar would predict the sentence to be ungrammatical. It happens in this case that the f-structures merge successfully: the following figure schematizes the process of construction. Finally, the finished f-structure appears as in (38).

F-Struc. / Attr.	f_2		(37)		MERGER
PRED	'JOHN'		Ø	$\longrightarrow$	'JOHN'
NUM	SING	$\longleftrightarrow$	SING	$\longrightarrow$	SING
PERS	3	$\longleftrightarrow$	3	$\longrightarrow$	3

FIGURE XI: MERGER OF f_2 AND (37)

(38)
$$
\begin{array}{l} f_1 \\ f_4 \\ f_5 \end{array}
\left[\begin{array}{lll}
\text{PRED} & & \text{'SEE<}(f_5\ \text{SUBJ})\ \ (f_5\ \text{OBJ})\text{>'} \\
\text{SUBJ} & \begin{array}{l} f_2 \\ f_3 \end{array} & \left[\begin{array}{ll} \text{PRED} & \text{'JOHN'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \\
\text{OBJ} & \begin{array}{l} f_6 \\ f_7 \end{array} & \left[\begin{array}{ll} \text{PRED} & \text{'MARY'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right]
\end{array}\right]
$$

6. F-Structure and the Grammaticality of Sentences

In order for a sentence to be predicted grammatical by a Lexical Functional Grammar, it must satisfy two criteria. First, the grammar must be able to assign it a constituent structure tree, and, second, the grammar must assign it a well formed f-structure. The first criterion is familiar to any linguist who has any training in syntax. As for the second, this means that one must be able to construct an f-structure for the functional description yielded by the fully instantiated annotated c-structure and also that that f-structure must be in accord with certain principles on the form of f-structures.

6.1. Consistency. Consistency is a property of functional descriptions: a functional description is consistent if there exists an f-structure that can make all of the equations in that functional description true. This point merits elaboration. Recall that we stated above that an f-structure may associate at most one value with a given attribute. Given this fact it should be obvious that not every collection of functional equations can lead to an f-structure which adheres to this uniqueness condition. For instance the two equations listed below are incompatible.

(38') a. $(f_{43}\ \text{PERS})=2$
b. $(f_{43}\ \text{PERS})=3$

Taken together, they imply that both of the values 2 and 3 may be assigned to the attribute PERS in f_{43}, thus violating the uniqueness condition. When it is impossible to construct an f-structure from a functional description, we say that that functional description is INCONSISTENT.

6.2. Completeness and Coherence. Next we consider conditions placed on completed f-structures. Up to this point we have been thinking about f-structure in a rather mechanical fashion; now, however, it is time to step back and reflect on what sort of linguistic concepts an f-structure is supposed to express. Basically, f-structures are the repository of all information about grammatical functions, like *subject*, *object*, etc. In LFG grammatical functions are thought to be primitive parts of natural language syntax, that is, not derivable from any other aspect of the grammar. In other words, there seem to be generalizations about regularities in sentential syntax, morphology, paraphrase relations, etc. which are not fully expressible when we limit the apparatus available for distinguishing among elements within clause to notions like *case* (e.g. *nominative NP*), *constituent structure* (e.g. *NP immediately dominated by S*), or of *semantic argumenthood* (e.g. *agent* in a system of semantics that distinguishes among types of arguments). Traditional notions of grammatical functions (e.g. *subject*) appear, however, to provide a useful classification of clausal components, which facilitates the formulation of certain linguistically interesting generalizations, into which the previously mentioned notions did not provide many insights. Arguments in favor of this position abound in the LFG literature. Consequently one of the principle motivations behind LFG has been the desire to construct a theory of grammatical functions, and f-structures have been employed as a means of displaying relations among these functions.

Consider the finished f-structure for example (7) pictured in (38). The outer f-structure is that which corresponds to the whole clause. The reader may find it useful to refer to Figure X. Inside of that f-structure are found lines where the attributes are SUBJ, OBJ and PRED. There could, of course, be an entirely different set of attributes, if we were dealing with another structure. These names are, of course, transparent abbreviations for *subject*, *object*, and *predicate*, all familiar concepts used in discussions of natural language grammar. So it will come as no surprise then that the values of these attributes are to be interpreted as representations of these linguistically relevant categories.

In this section we wish to discuss constraints on f-structures which are essentially statements about the relations between grammatical functions manifested in the sentence and the sentence's main predicate. One of the central assumptions in LFG is that the workings of grammatical functions are regulated through the so-called predicate argument structure found in the semantic form paired with PRED. Semantic forms appear graphically as material flanked with single quotes.

(39) a. 'JOHN'
b. 'SEE<(↑ SUBJ)(↑ OBJ)>'

All semantic forms contain an expression, e.g. JOHN or SEE, which is interpreted in the semantics, and some incorporate PREDICATE ARGUMENT STRUCTURES, i.e. lists of arguments which the semantic entity expresses a relation on. These argument lists are flanked with angled brackets, and each place within these brackets stands for a semantic argument.[10]

Now, to highlight an obvious point, the arguments in the predicate argument structure are elements employed at the level of semantic representation. The grammar must express how it is that the syntactic level of representation is related to the semantics, and the syntactic entities to which LFG associates semantic arguments are grammatical functions. The association of grammatical functions to semantic arguments is established by recording the names of grammatical functions in given argument positions (see (39)b). When a grammatical function is associated with a semantic argument, it is said to be GOVERNED by the argument. It is through this linking of arguments and grammatical functions that LFG handles various paraphrase relations; the principles governing variations in this association and their diverse morphological and syntactic consequences have been the focus of much research in LFG. The linking of grammatical functions to arguments also determines to a large extent what structures may or may not occur as elements of the sentence. That is to say that these predicate argument structures enforce subcategorization restrictions. This is made possible by the existence of two principles known as COMPLETENESS and COHERENCE.

Completeness states that if a grammatical function is mentioned in the predicate argument structure, then it must be represented in the f-structure. The principle of completeness accounts for the deviance of examples like that seen in (40).

(40) John likes.

We may assume that the lexical entry for *likes* would contain the following equations.

(41) likes V (↑ PRED)='LIKE<(↑ SUBJ)(↑ OBJ)>'
(↑ SUBJ NUM)=SING
(↑ SUBJ PERS)=3

The reader may verify that (42) is the f-structure that our example grammar, augmented with the lexical item in (41), would provide for (40).

(42)

$$\left[\begin{array}{ll} \text{PRED} & \text{'LIKE}\langle(\uparrow \text{SUBJ}) (\uparrow \text{OBJ})\rangle\text{'} \\ \text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{'JOHN'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \end{array}\right]$$

The f-structure does not display any variables naming its components, so we have adopted the easily interpretable convention of retaining the mother metavariables within the semantic form for *like*. The referent of these metavariables will be the immediately dominating f-structure. Since (42) does not provide a value for OBJ, completeness is violated. Note that for there to be a value for OBJ in (42), the corresponding sentence would have had to contain a postverbal NP: a glance at our example grammar will confirm this. Thus, one can easily see how completeness assumes the positive role of subcategorization, i.e. that of ensuring the presence of certain elements.

Coherence could be viewed as the inverse of completeness. This principle limits the occurrence of governable grammatical functions within f-structures. If any lexical entry in the grammar mentions a given grammatical function in the predicate argument structure of its predicate, then that grammatical function is governable. Coherence states that if a governable grammatical function occurs in an f-structure, then it must actually be governed by some argument in the predicate argument of that f-structure's predicate. This obviously fulfills the negative role of subcategorization, the exclusion of elements not specifically licensed by the predicate. Thus, given the lexical entry in (43), the sentence in (44) would yield the f-structure in (45).

(43) falls V ($\uparrow$ PRED)='FALL$\langle$($\uparrow$ SUBJ)$\rangle$'
($\uparrow$ SUBJ NUM)=SING
($\uparrow$ SUBJ PERS)=3

(44) John falls Mary.

(45)
$$\left[\begin{array}{ll} \text{PRED} & \text{'FALL}\langle(\uparrow \text{SUBJ})\rangle\text{'} \\ \text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{'JOHN'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \\ \text{OBJ} & \left[\begin{array}{ll} \text{PRED} & \text{'MARY'} \\ \text{NUM} & \text{SING} \\ \text{PERS} & 3 \end{array}\right] \end{array}\right]$$

The f-structure in (45) displays a predicate argument structure where there is no mention of the object. This accurately expresses the obvious problem with (44): the object, *Mary*, cannot be governed by any semantic argument and is therefore superfluous.

7. Constraining Equations and Booleans

In this final section we wish to point out the existence of some additional features of functional equations.

7.1. Constraining Equations. Constraining equations are identifiable by the letter *c* which occurs in the symbol $=_c$. As for the mechanical aspects of using this device, one begins by partitioning the functional description into two sets, one containing only constraining equations, and one containing only non-constraining equations. This done, one considers the set of non-constraining equations and builds the minimal f-structure which these specify. Only after the f-structure is built up are the constraining equations put to use. These equations are then examined to determine if they hold true of the finished f-structure. If all of the constraining equations are made true by the f-structure constructed from the non-constraining equations, then that f-structure is well formed. Otherwise it is inconsistent.

The following example illustrates the use of constraining equations. Consider the following functional description.

(46) a. $(f_n\ \text{A})=\text{B}$
b. $(f_n\ \text{C})=\text{D}$
c. $(f_n\ \text{C})=_c \text{D}$

We first examine the non-constraining equations in (46), that is (46)a and (46)b. The minimal functional structure which these equations describe is that in (47).

(47) $f_n \begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \end{bmatrix}$

Note that (47) is the completed f-structure for the functional description in (46): equation (46)c holds true of (47), that is, f_n contains a line pairing the attribute C with the value D. Since f_n has the desired characteristic, the constraining equation is satisfied, and the f-structure is coherent.

7.2. Negatives. Functional equations may take a negative operator. This may be written as follows.

(48) $\neg(f_n \text{ E})=\text{F}$

The resultant equation is a variant form of constraining equation. That is, it too is applies after all normal equations have been considered and the minimal f-structure is completed. However, a negative equation, say $\neg eq$ is satisfied by a given f-structure, only in the case where eq is false with regard to that f-structure. This characteristic is illustrated below.

We assume the following functional description.

(49) a. $(f_n \text{ A})=\text{B}$
b. $(f_n \text{ C})=\text{D}$
c. $\neg(f_n \text{ E})=\text{F}$

Using (49)a and (49)b we construct (50).

(50) $f_n \begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \end{bmatrix}$

Turning to (49)c we consider its polar opposite, (51).

(51) $(f_n \text{ E})=\text{F}$

It happens that (51) is false with regard to (50), thus making (49)c true. Hence, (49) is consistent.

7.3. Conjunction. We have been tacitly employing conjunction throughout the latter half of this document. In f-descriptions, like (11), there are typically many equations, and all of them must be satisfied. This is the essence of conjunction: the conjunction of the equations $eq_1, ..., eq_n$ is true, precisely when each of the component equations $eq_1, ..., eq_n$ is true considered in isolation. Therefore, let us now state explicitly that we assume a conjunction whenever we write down a sequence of two or more equations. It sometimes becomes necessary however to disambiguate expressions involving more than one boolean. For instance, suppose we see (52): does the negation apply just to the first equation or to the conjunction of both equations.

(52) $\neg(f_n\ \mathrm{A}){=}\mathrm{B}$
$(f_n\ \mathrm{C}){=}\mathrm{D}$

We must establish a convention to decide such issues. Let us follow the traditional practice of assuming that negation applies to the smallest entity to is immediate right. This means that in the case of (52), only the first equations is negated. In order to indicate that the negation of a conjunction is implied, we might then employ one of the bracketing conventions in (53)a and b.

(53) a. $\neg\begin{bmatrix}(f_n\ \mathrm{A}){=}\mathrm{B}\\(f_n\ \mathrm{C}){=}\mathrm{D}\end{bmatrix}$

b. $\neg\big[(f_n\ \mathrm{A}){=}\mathrm{B}\quad(f_n\ \mathrm{C}){=}\mathrm{D}\big]$

7.4. Disjunction. The final boolean used in LFG is disjunction. This joins together two or more equations, as was the case with the conjunction operator. A disjunction of equations like that in (54) would be true if either or both of the component equations is true.

(54) $\begin{Bmatrix}(f_n\ \mathrm{A}){=}\mathrm{B}\\(f_n\ \mathrm{C}){=}\mathrm{D}\end{Bmatrix}$

However, the nature of f-structures entails certain consequences which may not be expected.

The first question to ask is what does it mean to say that either or both of the component equations of a disjunction will be true. In fact, this means that the functional description which contains the disjunction, may correspond to more than one f-structure. To see this point let us take the following functional description as an example.

(55) a. $\left\{\begin{matrix}(f_n \text{ A})=\text{B} \\ (f_n \text{ C})=\text{D}\end{matrix}\right\}$

b. $(f_n \text{ E})=\text{F}$

Note that there are three obvious candidates for f-structures that make the two equations in (55) true.

(56) a. $f_n \begin{bmatrix} \text{A} & \text{B} \\ \text{E} & \text{F} \end{bmatrix}$ b. $f_n \begin{bmatrix} \text{C} & \text{D} \\ \text{E} & \text{F} \end{bmatrix}$ c. $f_n \begin{bmatrix} \text{A} & \text{B} \\ \text{C} & \text{D} \\ \text{E} & \text{F} \end{bmatrix}$

All of these f-structures make (55)b true, since each one contains a line where the attribute is E and the value is F. The f-structure in (56)a makes (55)a true, because its first line validates the first disjunct (component equation of the disjunction) of (55)a, making the whole disjunct true. Similarly the first line of (56)b makes the second disjunct of (55)a true. Finally, we note that the first two lines of (56)c validate both disjuncts of (55)a. In sum all of these f-structures satisfy the functional description in (55).

Now let us consider the three f-structures in (56) in terms of minimality. Starting with (56)a, we note that removing the first line invalidates (55)a, and subtracting the second line invalidates (55)b. Since neither line may be taken away, (56)a is minimal. Turning to the second f-structure, the reader may verify that neither of its two lines may be removed without invalidating one or the other of the equations in (55). Thus, (56)b is also minimal. Finally let us consider (56)c. Notice here that if we subtract either (but not both) of the first two lines of the f-structure, both of the equations in (55) will remain true of the resultant f-structure. We know then that this f-structure is not minimal, and therefore is not licensed by (55).

Finally let us state explicitly that equations inside of a disjunction are always interpreted as separate disjuncts. They are never treated as a conjunction within the disjunction, unless there is explicit bracketing to indicate that interpretation. Thus, (57)a contains exactly three disjuncts, and (57)b two.

(57) a. $\left\{\begin{array}{l}(f_n\ \text{A})=\text{B}\\(f_n\ \text{C})=\text{D}\\(f_n\ \text{E})=\text{F}\end{array}\right\}$

b. $\left\{\begin{array}{c}(f_n\ \text{A})=\text{B}\\\left[\begin{array}{l}(f_n\ \text{C})=\text{D}\\(f_n\ \text{E})=\text{F}\end{array}\right]\end{array}\right\}$

8. Final Note

We hope that the information contained in the foregoing sections will suffice to allow the reader to tackle theoretical and descriptive works based on LFG. Finally, in order to facilitate future exploration of the LFG literature, we point out that there are two convenient collections of articles in LFG, Bresnan (1982) and Levin et al. (1983), both of which contain papers of major importance.

Notes

*I would like to express my thanks to Joan Bresnan, Mary Dalrymple, Jeff Goldberg, Jonni Kanerva, and Mariko Saiki, who provided helpful comments on the contents of this paper. I am also especially grateful to Mariko Saiki for her help in the production of this manuscript.

[1]Another representation known variously as the SEMANTIC or SITUATION STRUCTURE is also associated with sentences in some works. However, we shall not deal with this issue here.

[2]The readers interested in a more formal specification of LFG's are directed to Kaplan and Bresnan (1982).

[3]The reader not familiar with the context free rewriting rule formalism is advised to consult either Baker (1978) or Hopcroft and Ullman (1979); the first work has a linguistic slant, while the other is oriented towards formal language theory. Incidentally, these comments should not be interpreted as implying that LFG's are equivalent to context free grammars.

[4]The resemblance to the rules of the base component in the Standard Theory of transformational grammar is one of *form*, not necessarily one of *content*. In other words the reader should be sure to grasp the distinction between context free rule viewed as a type of formal device on the one hand and the actual base rules which have been employed over and over again in the essays of transformational grammarians on the other. To consider an example that is extremely relevant, because it arises regularly the the discussions of many introductory courses in LFG, take the question of whether or not a verb phrase node (VP) is a necessary part of constituent structure descriptions of a given language. Many transformationalists seem to automatically assume the existence of a VP in deep structure, accounting for any surface word order facts incompatible with this constituency by means of movement or scrambling transformations. An LFG however would generate the surface word order directly, avoiding the use of the unmotivated VP node (the operative word here is unmotivated; i.e., the node is not employed unless it is shown

to be necessary to provide a proper description of the facts). Thus, we have a striking example where rules in an LFG might differ in content from the phrase structure rules habitually employed in the base components of transformational grammars.

[5]We mean in no way to exclude the possibility of adding other relevant information to the entries in the lexicon. In particular, information about the morphological characteristics of an item are also found in the lexical entries given in some studies. In this description we wish merely to make perfectly clear the practices followed in most works dealing with LFG.

[6]In the examples given here the representation of the form of the item is given in the conventional orthography; however, one could easily imagine other alternatives, like rendering the form of the item phonemically or phonetically.

[7]In the present document we represent syntactic categories by means of atomic category symbols, although there is no principled reason why other sorts of categorial specifications, e.g. feature matrices, might not be used.

[8]A caveat is in order here: we do not wish to claim that the set of functional schemata displayed in this figure (or in any other lexical entry exhibited in this paper) is exactly appropriate for the optimal grammar of English. It may be that we need significantly more schemata in order to fully render all of the intricacies of the grammar of English; however, we wish at all times to stress that it is the description that captures the most generalizations in the most economical and elegant way that we must strive for. For instance, a truly good theory of features might allow us to minimize the amount of machinery employed to such an extent that we could actually reduce the number of schemata required.

[9]It may be helpful to note here that f-structures are functions in the formal set theoretic sense. Only the mode of graphic representation differs: example (12) would be written as (i) in the standard notation of set theory.

(i) $\{\langle \mathrm{A}, \{\langle \mathrm{B}, \mathrm{C}\rangle, \langle \mathrm{D}, \mathrm{E}\rangle\}\rangle, \langle \mathrm{F}, \mathrm{G}\rangle, \langle \mathrm{H}, \mathrm{I}\rangle\}$

For those knowledgeable in set theory, be advised that all the axioms and theorems that pertain to set theoretic functions pertain to f-structures as well. Readers who do not feel very at home in set theory should not feel at a disadvantage, however, as no experience with set theory is assumed in this exposition, and one does not appeal to such knowledge when using the LFG formalism to construct linguistic analyses.

[10]The reader is advised that there is a prevalent and yet unspoken convention among researchers in LFG concerning the left to right order of arguments according to thematic roles. Agent and experiencer arguments seem always to precede all other roles. The theme argument appears next, followed by the goal. All other arguments usually follow the above four, and their ordering seems arbitrary.

Bibliography

Baker, C. L. 1978. Introduction to generative-transformational syntax. Englewood Cliffs, NJ: Prentice-Hall.

Bresnan, Joan, ed. 1982. The mental representations of grammatical relations. Cambridge Mass.: The MIT Press.

Hopcroft, John E. and Jeffrey D. Ullman. 1979. Introduction to automata theory, languages and computation. Reading Mass.: Addison-Wesley.

Kaplan, Ronald M. and Joan Bresnan. 1982. Lexical-Functional Grammar: a formal system for grammatical representation. in Bresnan 1982.

Levin, Lori, Malka Rappaport, and Annie Zaenen, eds. 1983. Papers in Lexical-Functional Grammar. Bloomington, Ind.: Indiana University Linguistics Club.