TREATMENT OF LONG DISTANCE DEPENDENCIES IN LFG AND TAG: FUNCTIONAL UNCERTAINTY IN LFG IS A COROLLARY IN TAG*

Aravind K. Joshi
Dept. of Computer & Information Science
University of Pennsylvania
Philadelphia, PA 19104
joshi@linc.cis.upenn.edu

K. Vijay-Shanker
Dept. of Computer & Information Science
University of Delaware
Newark, DE 19716
vijay@udel.edu

ABSTRACT

In this paper the functional uncertainty machinery in LFG is compared with the treatment of long distance dependencies in TAG. It is shown that the functional uncertainty machinery is redundant in TAG, i.e., what functional uncertainty accomplishes for LFG follows from the TAG formalism itself and some aspects of the linguistic theory instantiated in TAG. It is also shown that the analyses provided by the functional uncertainty machinery can be obtained without requiring power beyond mildly context-sensitive grammars. Some linguistic and computational aspects of these results have been briefly discussed also.

1 INTRODUCTION

The so-called long distance dependencies are characterized in Lexical Functional Grammars (LFG) by the use of the formal device of *functional uncertainty*, as defined by Kaplan and Zaenan [3] and Kaplan and Maxwell [2]. In this paper, we relate this characterization to that provided by Tree Adjoining Grammars (TAG), showing a direct correspondence between the functional uncertainty equations in LFG analyses and the elementary trees in TAGs that give analyses for "long distance" dependencies. We show that the functional uncertainty machinery is redundant in TAG, i.e., what *functional uncertainty* accomplishes for LFG follows from the TAG formalism itself and some fundamental aspects of the linguistic theory instantiated in TAG. We thus show that these analyses can be obtained without requiring power beyond mildly context-sensitive grammars. We also briefly discuss the linguistic and computational significance of these results.

Long distance phenomena are associated with the so-called movement. The following examples,

1. *Mary Henry telephoned.*
2. *Mary Bill said that Henry telephoned.*
3. *Mary John claimed that Bill said that Henry telephoned.*

illustrate the long distance dependencies due to topicalization, where the verb *telephoned* and its object *Mary* can be arbitrarily apart. It is difficult to state generalizations about these phenomena if one relies entirely on the surface structure (as defined in CFG based frameworks) since these phenomena cannot be localized at this level. Kaplan and Zaenan [3] note that, in LFG, rather than stating the generalizations on the c-structure, they must be stated on f-structures, since long distance dependencies are predicate argument dependencies, and such functional dependencies are represented in the f-structures. Thus, as stated in [2, 3], in the sentences (1), (2), and (3) above, the dependencies are captured by the equations (in the LFG notation[1]) by $\uparrow TOPIC = \uparrow OBJ$, $\uparrow TOPIC = \uparrow COMP\ OBJ$, and $\uparrow TOPIC = \uparrow COMP\ COMP\ OBJ$, respectively, which state that the topic *Mary* is also the object of *telephoned*. In general, since any number of additional complement predicates may be introduced, these equations will have the general form

$$\uparrow TOPIC = \uparrow COMP\ COMP\ \ldots\ OBJ$$

Kaplan and Zaenen [3] introduced the formal device of *functional uncertainty*, in which this general case is stated by the equation

*This work was partially supported (for the first author) by the DARPA grant N00014-85-K0018, ARO grant DAA29-84-9-0027, and NSF grant IRI84-10413-A02. The first author also benefited from some discussion with Mark Johnson and Ron Kaplan at the Titisee Workshop on Unification Grammars, March, 1988.

[1] Because of lack of space, we will not define the LFG notation. We assume that the reader is familiar with it.

$$\uparrow TOPIC = \uparrow COMP^{*}OBJ$$

The functional uncertainty device restricts the labels (such as $COMP^{*}$) to be drawn from the class of regular expressions. The definition of f-structures is extended to allow such equations [2, 3]. Informally, this definition states that if f is a f-structure and α is a regular set, then $(f\alpha) = v$ holds if the value of f for the attribute s is a f-structure f_1 such that $(f_1 y) = v$ holds, where sy is a string in α, or $f = v$ and $\epsilon \in \alpha$.

The functional uncertainty approach may be characterized as a *localization* of the long distance dependencies; a localization at the level of f-structures rather than at the level of c-structures. This illustrates the fact that if we use CFG-like rules to produce the surface structures, it is hard to state some generalizations directly; on the other hand, f-structures or elementary trees in TAGs (since they localize the predicate argument dependencies) are appropriate domains in which to state these generalizations. We show that there is a direct link between the regular expressions used in LFG and the elementary trees of TAG.

1.1 OUTLINE OF THE PAPER

In Section 2, we will define briefly the TAG formalism, describing some of the key points of the linguistic theory underlying it. We will also describe briefly Feature Structure Based Tree Adjoining Grammars (FTAG), and show how some elementary trees (auxiliary trees) behave as functions over feature structures. We will then show how regular sets over labels (such as $COMP^{*}$) can also be denoted by functions over feature structures. In Section 3, we will consider the example of topicalization as it appears in Section 1 and show that the same statements are made by the two formalisms when we represent both the elementary trees of FTAG and functional uncertainties in LFG as functions over feature structures. We also point out some differences in the two analyses which arise due to the differences in the formalisms. In Section 4, we point out how these similar statements are stated differently in the two formalisms. The equations that capture the linguistic generalizations are still associated with individual rules (for the c-structure) of the grammar in LFG. Thus, in order to state generalizations for a phenomenon that is not localized in the c-structure, extra machinery such as functional uncertainty is needed. We show that what this extra machinery achieves for CFG based systems follows as a corollary of the TAG framework. This results from the fact that the elementary trees in a TAG provide an *extended domain of locality*, and factor out recursion and dependencies. A computational consequence of this result is that we can obtain these analyses without going outside the power of TAG and thus staying within the class of constrained grammatical formalisms characterized as *mildly context-sensitive* (Joshi [1]). Another consequence of the differences in the representations (and localization) in the two formalisms is as follows. In a TAG, once an elementary tree is picked, there is *no uncertainty* about the functionality in long distance dependencies. Because LFG relies on a CFG framework, interactions between uncertainty equations can arise; the lack of such interactions in TAG can lead to simpler processing of long distance dependencies. Finally, we make some remarks as to the linguistic significance of restricting the use of regular sets in the functional uncertainty machinery by showing that the linguistic theory instantiated in TAG can predict that the path depicting the "movement" in long distance dependencies can be characterized by regular sets.

2 INTRODUCTION TO TAG

Tree Adjoining Grammars (TAGs) are tree rewriting systems that are specified by a finite set of *elementary* trees. An operation called *adjoining*[2] is used to compose trees. The key property of the linguistic theory of TAGs is that TAGs allow factoring of recursion from the domain of dependencies, which are defined by the set of elementary trees. Thus, the elementary trees in a TAG correspond to *minimal* linguistic structures that localize the dependencies such as agreement, subcategorization, and filler-gap. There are two kinds of elementary trees: the *initial trees* and *auxiliary trees*. The initial trees (Figure 1) roughly correspond to "simple sentences". Thus, the root of an initial tree is labeled by S or $\overline{S}$. The frontier is all terminals.

The auxiliary trees (Figure 1) correspond roughly to minimal recursive constructions. Thus, if the root of an auxiliary tree is labeled by a nonterminal symbol, X, then there is a node (called the foot node) in the frontier which is labeled by X. The rest of the nodes in the frontier are labeled by terminal symbols.

[2]We do not consider lexicalized TAGs (defined by Schabes, Abeillé, and Joshi [7]) which allow both adjoining and substitution. The results of this paper apply directly to them. Besides, they are formally equivalent to TAGs.

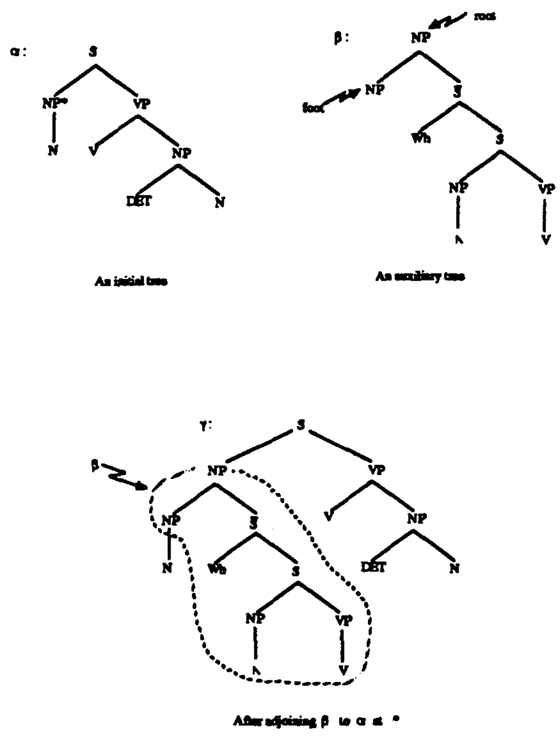

Figure 1: Elementary Trees in a TAG

We will now define the operation of adjoining. Consider the adjoining of β at the node marked with * in α. The subtree of α under the node marked with * is excised, and β is inserted in its place. Finally, the excised subtree is inserted below the foot node of α, as shown in Figure 1.

A more detailed description of TAGs and their linguistic relevance may be found in (Kroch and Joshi [5]).

2.1 FEATURE STRUCTURE BASED TREE ADJOINING GRAMMARS (FTAG)

In unification grammars, a feature structure is associated with a node in a derivation tree in order to describe that node and its relation to features of other nodes in the derivation tree. In a FTAG, with each internal node, η, we associate two feature structures (for details, see [9]). These two feature structures capture the following relations (Figure 2)

1. The relation of η to its supertree, i.e., the view of the node from the top. The feature structure that describes this relationship is called t_η.

2. The relation of η to its descendants, i.e., the view from below. This feature structure is called b_η.

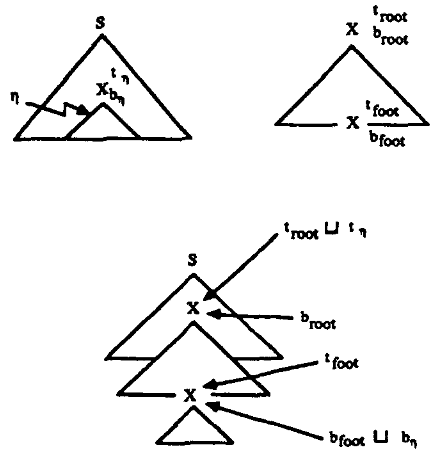

Figure 2: Feature Structures and Adjoining

Note that both the t_η and b_η feature structures hold for the node η. On the other hand, with each leaf node (either a terminal node or a foot node), η, we associate only one feature structure (let us call it t_η[3]).

Let us now consider the case when adjoining takes place as shown in the Figure 2. The notation we use is to write alongside each node, the t and b statements, with the t statement written above the b statement. Let us say that t_{root},b_{root} and $t_{foot} = b_{foot}$ are the t and b statements of the root and foot nodes of the auxiliary tree used for adjoining at the node η. Based on what t and b stand for, it is obvious that on adjoining the statements t_η and t_{root} hold for the node corresponding to the root of the auxiliary tree. Similarly, the statements b_η and b_{foot} hold for the node corresponding to the foot of the auxiliary tree. Thus, on adjoining, we unify t_η with t_{root}, and b_η with b_{foot}. In fact, this adjoining is permissible only if t_{root} and t_η are compatible and so are b_{foot} and b_η. If we do not adjoin at the node, η, then we unify t_η with b_η. More details of the definition of FTAG may be found in [8, 9].

We now give an example of an initial tree and an auxiliary tree in Figure 3. We have shown only the necessary top and bottom feature structures for the relevant nodes. Also in each feature structure

[3]The linguistic relevance of this restriction has been discussed elsewhere (Kroch and Joshi [5]). The general framework does not necessarily require it.

shown, we have only included those feature-value pairs that are relevant. For the auxiliary tree, we have labeled the root node S. We could have labeled it $\overline{S}$ with $COMP$ and S as daughter nodes. These details are not relevant to the main point of the paper. We note that, just as in a TAG, the elementary trees which are the domains of dependencies are available as a single unit during each step of the derivation. For example, in α_1 the topic and the object of the verb belong to the same tree (since this dependency has been factored into α_1) and are coindexed to specify the *movement* due to topicalization. In such cases, the dependencies between these nodes can be stated directly, avoiding the percolation of features during the derivation process as in string rewriting systems. Thus, these dependencies can be checked locally, and thus this checking need not be linked to the derivation process in an unbounded manner.

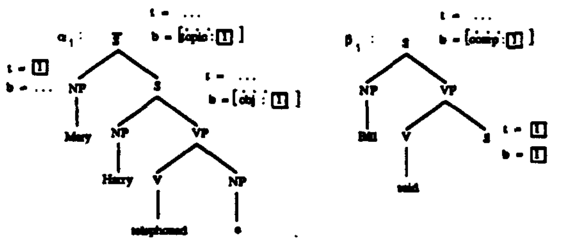

Figure 3: Example of Feature Structures Associated with Elementary Trees

2.2 A CALCULUS TO REPRESENT FTAG

In [8, 9], we have described a calculus, extending the logic developed by Rounds and Kasper [4, 6], to encode the trees in a FTAG. We will very briefly describe this representation here.

To understand the representation of adjoining, consider the trees given in Figure 2, and in particular, the node η. The feature structures associated with the node where adjoining takes place should reflect the feature structure after adjoining and as well as without adjoining. Further, the feature structure (corresponding to the tree structure below it) to be associated with the foot node is not known prior to adjoining, but becomes specified upon adjoining. Thus, the bottom feature structure associated with the foot node, which is b_{foot} before adjoining, is instantiated on adjoining by unifying it with a feature structure for the tree that will finally appear below this node. Prior to adjoining, since this feature structure is not known, we will treat it as a variable that gets instantiated on adjoining. This treatment can be formalized by treating the auxiliary trees as functions over feature structures (by λ-abstracting the variable corresponding to the feature structure for the tree that will appear below the foot node). Adjoining corresponds to applying this function to the feature structure corresponding to the subtree below the node where adjoining takes place.

Treating adjoining as function application, where we consider auxiliary trees as functions, the representation of β is a function, say f_β, of the form (see Figure 2)

$$\lambda f.(t_{root} \wedge \ldots (b_{foot} \wedge f))$$

If we now consider the tree γ and the node η, to allow the adjoining of β at the node η, we must represent γ by

$$(\ldots t_\eta \wedge f_\beta(b_\eta) \wedge \ldots)$$

Note that if we do not adjoin at η, since t_η and β_η have to be unified, we must represent γ by the formula

$$(\ldots t_\eta \wedge b_\eta \wedge \ldots)$$

which can be obtained by representing γ by

$$(\ldots t_\eta \wedge I(b_\eta) \wedge \ldots)$$

where I is the identity function. Similarly, we must allow adjoining by any auxiliary tree adjoinable at η (admissibility of adjoining is determined by the success or failure of unification). Thus, if $\beta_1, \ldots, \beta_n$ form the set of auxiliary trees, to allow for the possibility of adjoining by any auxiliary tree, as well as the possibility of no adjoining at a node, we must have a function, F, given by

$$F = \lambda f.(f_{\beta_1}(f) \vee \ldots \vee f_{\beta_n}(f) \vee f)$$

and then we represent γ by

$$(\ldots t_\eta \wedge F(b_\eta) \wedge \ldots).$$

In this way, we can represent the elementary trees (and hence the grammar) in an extended version of R-K logic (the extension consists of adding λ-abstraction and application).

3 LFG AND TAG ANALYSES FOR LONG DISTANCE DEPENDENCIES

We will now relate the analyses of long distance dependencies in LFG and TAG. For this purpose, we will focus our attention only on the dependencies due to topicalization, as illustrated by sentences 1, 2, and 3 in Section 1.

To facilitate our discussion, we will consider regular sets over labels (as used by the functional uncertainty machinery) as functions over feature structures (as we did for auxiliary trees in FTAG). In order to describe the representation of regular sets, we will treat all labels (attributes) as functions over feature structures. Thus, the label $COMP$, for example, is a function which given a value feature structure (say v) returns a feature structure denoted by $COMP : v$. Therefore, we can denote it by $\lambda v.COMP : v$. In order to describe the representation of arbitrary regular sets we have to consider only their associated regular expressions. For example, $COMP^*$ can be represented by the function $C*$ which is the fixed-point[4] of

$$F = \lambda v.(F(COMP : v) \vee v)^5$$

Thus, the equation

$$\uparrow TOPIC = \uparrow COMP^* OBJ$$

is satisfied by a feature structure that satisfies $TOPIC : v \wedge C*(OBJ : v)$. This feature structure will have a general form described by $TOPIC : v \wedge COMP : COMP : \ldots OBJ : v$.

Consider the FTAG fragment (as shown in Figure 3) which can be used to generate the sentences 1, 2, and 3 in Section 1. The initial tree α_1 will be represented by $cat : \overline{S} \wedge F(topic : v \wedge F(pred : telephoned \wedge obj : v))$. Ignoring some irrelevant details (such as the possibility of adjoining at nodes other than the S node), we can represent α_1 as

$$\alpha_1 = topic : v \wedge F(obj : v)$$

Turning our attention to β_1, let us consider the bottom feature structure of the root of β_1. Since its COMP is the feature structure associated with the foot node (notice that no adjoining is allowed at the foot node and hence it has only one feature structure), and since adjoining can take place at the root node, we have the representation of β_1 as

$$\lambda f.F(comp : f \wedge subj : (\ldots) \wedge \ldots)$$

where F is the function described in Section 2.2. From the point of view of the path from the root to the complement, the NP and VP nodes are irrelevant, so are any adjoinings on these nodes. So once again, if we discard the irrelevant information (from the point of view of comparing this analyses with the one in LFG), we can simplify the representation of β_1 as

$$\lambda f.F(comp : f)$$

As explained in Section 2.2, since β_1 is the only auxiliary tree of interest, F would be defined as $F = \lambda f.\beta_1(f) \vee f$. Using the definition of β_1 above, and making some reductions we have

$$F = \lambda f.F(comp : f) \vee f$$

This is exactly the same analysis as in LFG using the functional uncertainty machinery. Note that the fixed-point of F is $C*$. Now consider α_1. Obviously any structure derived from it can now be represented as

$$topic : v \wedge C*(obj : v)$$

This is the same analysis as given by LFG.

In a TAG, the dependent items are part of the same elementary tree. Features of these nodes can be related locally within this elementary tree (as in α_1). This relation is unaffected by any adjoinings on nodes of the elementary tree. Although the paths from the root to these dependent items are elaborated by the adjoinings, no external device (such as the functional uncertainty machinery) needs to be used to restrict the possible paths between the dependent nodes. For instance, in the example we have considered, the fact that $TOPIC = COMP : COMP \ldots : OBJ$ follows from the TAG framework itself. The regular path restrictions made in functional uncertainty statements such as in $TOPIC = COMP^* OBJ$ is redundant within the TAG framework.

4 COMPARISON OF THE TWO FORMALISMS

We have compared LFG and TAG analyses of long distance dependencies, and have shown that what functional uncertainty does for LFG comes out as a corollary in TAG, without going beyond the power of mildly context sensitive grammars.

[4] In [8], we have established that the fixed-point exists.
[5] We use the fact that $R^* = R^* R \cup \{\epsilon\}$.

Both approaches aim to *localize* long distance dependencies; the difference between TAG and LFG arises due to the *domain of locality* that the formalisms provide (i.e., the domain over which statements of dependencies can be stated within the formalisms).

In the LFG framework, CFG-like productions are used to build the c-structure. Equations are associated with these productions in order to build the f-structure. Since the long distance dependencies are localized at the functional level, additional machinery (functional uncertainty) is provided to capture this localization. In a TAG, the elementary trees, though used to build the "phrase structure" tree, also form the domain for localizing the functional dependencies. As a result, the long distance dependencies can be localized in the elementary trees. Therefore, such elementary trees tell us exactly where the *filler* "moves" (even in the case of such unbounded dependencies) and the functional uncertainty machinery is not necessary in the TAG framework. However, the functional uncertainty machinery makes explicit the predictions about the *path* between the "moved" argument (filler) and the predicate (which is close to the gap). In a TAG, this prediction is not explicit. Hence, as we have shown in the case of topicalization, the nature of elementary trees determines the derivation sequences allowed and we can confirm (as we have done in Section 3) that this prediction is the same as that made by the functional uncertainty machinery.

4.1 INTERACTIONS AMONG UNCERTAINTY EQUATIONS

The functional uncertainty machinery is a means by which infinite disjunctions can be specified in a finite manner. The reason that infinite number of disjunctions appear, is due to the fact that they correspond to infinite number of possible derivations. In a CFG based formalism, the checking of dependency cannot be separated from the derivation process. On the other hand, as shown in [9], since this separation is possible in TAG, only finite disjunctions are needed. In each elementary tree, there is no uncertainty about the kind of dependency between a filler and the position of the corresponding gap. Different dependencies correspond to different elementary trees. In this sense there is disjunction, but it is still only finite. Having picked one tree, there is *no uncertainty* about the grammatical function of the filler, no matter how many COMPs come in between due to adjoining. This fact may have important consequences from the point of view of relative efficiency of processing of long distance dependencies in LFG and TAG. Consider, for example, the problem of *interactions* between two or more uncertainty equations in LFG as stated in [2]. Certain strings in $COMP^*$ cannot be solutions for

$$(f\ TOPIC) = (f\ COMP^*\ GF)$$

when this equation is conjoined (i.e., when it interacts) with $(f\ COMP\ SUBJ\ NUM) = SING$ and $(f\ TOPIC\ NUM) = PL$. In this case, the shorter string $COMP\ SUBJ$ cannot be used for $COMP^*\ GF$ because of the interaction, although the strings $COMP^i\ SUBJ$, $i \geq 2$ can satisfy the above set of equations. In general, in LFG, extra work has to be done to account for interactions. On the other hand, in TAG, as we noted above, since there is no uncertainty about the grammatical function of the filler, such interactions do not arise at all.

4.2 REGULAR SETS IN FUNCTIONAL UNCERTAINTY

From the definition of TAGs, it can be shown that the paths are always context-free sets [11]. If there are linguistic phenomena where the uncertainty machinery with regular sets is not enough, then the question arises whether TAG can provide an adequate analysis, given that paths are context-free sets in TAGs. On the other hand, if regular sets are enough, we would like to explore whether the regularity requirement has a linguistic significance by itself. As far as we are aware, Kaplan and Zaenen [3] do not claim that the regularity requirement follows from the linguistic considerations. Rather, they have illustrated the adequacy of regular sets for the linguistic phenomena they have described. However, it appears that an appropriate linguistic theory instantiated in the TAG framework will justify the use of regular sets for the long distance phenomena considered here.

To illustrate our claim, let us consider the elementary trees that are used in the TAG analysis of long distance dependencies. The elementary trees, α_1 and β_1 (given in Figure 3), are good representative examples of such trees. In the initial tree, α_1, the *topic* node is coindexed with the empty NP node that plays the grammatical role of *object*. At the functional level, this NP node is the object of the S node of α_1 (which is captured in the bottom feature structure associated with the S node). Hence, our representation of

α_1 (i.e., looking at it from the top) is given by $topic : v \wedge F(obj : v)$, capturing the "movement" due to topicalization. Thus, the path in the functional structure between the topic and the object is entirely determined by the function F, which in turn depends on the auxiliary trees that can be adjoined at the S node. These auxiliary trees, such as β_1, are those that introduce complementizer predicates. Auxiliary trees, in general, introduce modifiers or complementizer predicates as in β_1. (For our present discussion we can ignore the modifier type auxiliary trees). Auxiliary trees upon adjoining do not disturb the predicate argument structure of the tree to which they are adjoined. If we consider trees such as β_1, the *complement* is given by the tree that appears below the foot node. A principle of a linguistic theory instantiated in TAG (see [5]), similar to the *projection principle*, predicts that the complement of the root (looking at it from below) is the feature structure associated with the foot node and (more importantly) this relation cannot be disrupted by any adjoinings. Thus, if we are given the feature structure, f, for the foot node (known only after adjoining), the bottom feature structure of the root can be specified as $comp : f$, and that of the top feature structure of the root is $F(comp : f)$, where F, as in α_1, is used to account for adjoinings at the root.

To summarize, in α_1, the functional dependency between the topic and object nodes is entirely determined by the root and foot nodes of auxiliary trees that can be adjoined at the S node (the effect of using the function F). By examining such auxiliary trees, we have characterized the latter path as $\lambda f.F(comp : f)$. In grammatical terms, the path depicted by F can be specified by right-linear productions

$$F \rightarrow F\ comp : f \mid \epsilon$$

Since right-linear grammars generate only regular sets, and TAGs predict the use of such right-linear rules for the description of the paths, as just shown above, we can thus state that TAGs give a justification for the use of regular expressions in the functional uncertainty machinery.

4.3 GENERATIVE CAPACITY AND LONG DISTANCE DEPENDENCY

We will now show that what functional uncertainty accomplishes for LFG can be achieved within the FTAG framework without requiring power beyond that of TAGs. FTAG, as described in this paper, is unlimited in its generative capacity. By placing no restrictions on the feature structures associated with the nodes of elementary trees, it is possible to generate any recursively enumerable language. In [9], we have defined a restricted version of FTAG, called RFTAG, that can generate only TALs (the languages generated by TAGs). In RFTAG, we insist that the feature structures that are associated with nodes are bounded in size, a requirement similar to the finite closure membership restriction in GPSG. This restricted system will not allow us to give the analysis for the long distance dependencies due to topicalization (as given in the earlier sections), since we use the COMP attribute whose value cannot be bounded in size. However, it is possible to extend RFTAG in a certain way such that such analysis can be given. This extension of RFTAG still does not go beyond TAG and thus is within the class of *mildly context-sensitive* grammar formalisms defined by Joshi [1]. This extension of RFTAG is discussed in [10].

To give an informal idea of this extension and a justification for the above argument, let us consider the auxiliary tree, β_1 in Figure 3. Although we coindex the value of the *comp* feature in the feature structure of the root node of β_1 with the feature structure associated with the foot node, we should note that this coindexing does not affect the *context-freeness* of derivation. Stated differently, the adjoining sequence at the root is independent of other nodes in the tree in spite of the coindexing. This is due to the fact that as the feature structure of the foot of β_1 gets instantiated on adjoining, this value is simply substituted (and not unified) for the value of the *comp* feature of the root node. Thus, the *comp* feature is being used just as any other feature that can be used to give tree addresses (except that *comp* indicates *dominance* at the functional level rather than at the tree structure level). In [10], we have formalized this notion by introducing graph adjoining grammars which generate exactly the same languages as TAGs. In a graph adjoining grammar, β_1 is represented as shown in Figure 4. Notice that in this representation the *comp* feature is like the features 1 and 2 (which indicate the left and right daughters of a node) and therefore not used explicitly.

5 CONCLUSION

We have shown that for the treatment of long distance dependencies in TAG, the functional un-

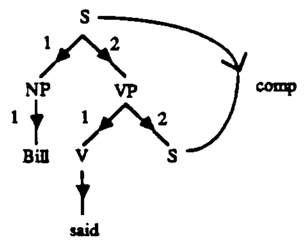

Figure 4: An Elementary DAG

certainty machinery in LFG is redundant. We have also shown that the analyses provided by the functional uncertainty machinery can be obtained without going beyond the power of mildly context-sensitive grammars. We have briefly discussed some linguistic and computational aspects of these results.

We believe that our results described in this paper can be extended to other formalisms, such as Combinatory Categorial Grammars (CCG), which also provide an *extended domain of locality.* It is of particular interest to carry out this investigation in the context of CCG because of their weak equivalence to TAG (Weir and Joshi [12]). This exploration will help us view this equivalence from the *structural* point of view.

REFERENCES

[1] A. K. Joshi. How much context-sensitivity is necessary for characterizing structural descriptions — Tree Adjoining Grammars. In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Natural Language Processing — Theoretical, Computational and Psychological Perspective*, Cambridge University Press, New York, NY, 1985. Originally presented in 1983.

[2] R. M. Kaplan and J. T. Maxwell. An algorithm for functional uncertainity. In 12th *International Conference on Comput. Ling.*, 1988.

[3] R. M. Kaplan and A. Zaenen. Long distance dependencies,constituent structure, and functional uncertainity. In M. Baltin and A. Kroch, editors, *Alternative Conceptions of Phrase Structure*, Chicago University Press, Chicago. IL, 1988.

[4] R. Kasper and W. C. Rounds. A logical semantics for feature structures. In 24th *meeting Assoc. Comput. Ling.*, 1986.

[5] A. Kroch and A.K. Joshi. *Linguistic Relevance of Tree Adjoining Grammars.* Technical Report MS-CIS-85-18, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, 1985. to appear in *Linguistics and Philosophy*, 1989.

[6] W. C. Rounds and R. Kasper. A complete logical calculus for record structures representing linguistic information. In *IEEE Symposium on Logic and Computer Science*, 1986.

[7] Y. Schabes, A. Abeille, and A. K. Joshi. New parsing strategies for tree adjoining grammars. In 12th *International Conference on Assoc. Comput. Ling.*, 1988.

[8] K. Vijayashanker. *A Study of Tee Adjoining Grammars.* PhD thesis, University of Pennsylvania, Philadelphia, Pa, 1987.

[9] K. Vijay-Shanker and A. K. Joshi. Feature structure based tree adjoining grammars. In 12th *International Conference on Comput. Ling.*, 1988.

[10] K. Vijay-Shanker and A.K. Joshi. Unification based approach to tree adjoining grammar. 1989. forthcoming.

[11] K. Vijay-Shanker, D. J. Weir, and A. K. Joshi. Characterizing structural descriptions produced by various grammatical formalisms. In 25th *meeting Assoc. Comput. Ling.*, 1987.

[12] D. J. Weir and A. K. Joshi. Combinatory categorial grammars: generative power and relationship to linear context-free rewriting systems. In 26th *meeting Assoc. Comput. Ling.*, 1988.