# Fixed and Flexible Phrase Structure: Coordination in Tree Adjoining Grammars[1]

*Aravind K. Joshi and Yves Schabes*

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104

Phrase-structure grammars assign a unique phrase structure (constituency) to an unambiguous sentence. Thus, for example, *John likes apples* will be bracketed as follows (ignoring the phrase labels and ignoring some brackets not essential for our present purpose):

(1) (John (likes apples))

There are systems, however, for example, Combinatory Categorial Grammars (CCGs)(Steedman, 1990) which assign multiple structures to unambiguous strings. Thus CCG assigns the following two groupings to *John likes apples*:

(2) (John (likes apples))
(3) ((John likes) apples)

The justification in CCG for such multiple structures is their use in coordination and in defining intonational phrases. Thus the bracketing (2) is necessary for (4) and the bracketing (3) for (5).

(4) (John ((likes apples) and (hates pears)))
(5) (((John likes) and (Bill hates)) beans)

Also, (2) corresponds to the intonational phrasing if the previous context is (6) and (3) if the previous context is (7).

(6) Who likes apples?
(John (likes apples))

(7) What does John like?
((John likes) apples)

[1]This research is partially supported by Darpa grant N0014-85-K0018, ARO grant DAAL03-89-C-0031PRI and NSF grant-IRI84-10413 A02.
We are grateful to Jamie Henderson, Anthony Kroch, Mitch Marcus, Stuart Shieber, Mark Steedman and K. Vijay-Shanker for providing valuable comments.

The work on CCG was presented by Mark Steedman in an earlier DARPA SLS Workshop (Steedman, 1989).

In this paper, we show how a CCG-like account for coordination can be constructed in the framework of lexicalized tree-adjoining grammars (TAGs) (Joshi, 1987; Schabes et al., 1988; Schabes, 1990).[2]. In particular, we show how a fixed constituency can be maintained at the level of the elementary trees of lexicalized TAGs and yet be able to achieve the kind of flexibility needed for dealing with the so-called non-constituents. This is the key significance of this contribution. In a CCG, being a constituent is the same as being a function. We show that this need not be the case and standard notions of constituency can be maintained. The key idea is that we use the lexicalized trees of TAG as *structured* categories with the associated functional types. Because of lack of space, we will illustrate our ideas by examples only.

Lexicalized TAGs (with substitution and adjunction) are similar to CCGs in the sense that for each lexical item the elementary tree(s) which is (are) anchored on that lexical item can be regarded as the (structured) category (categories) associated with that item. Figure 1 and Figure 2 give examples of elementary trees that can be found in lexicon for a Lexicalized TAG. The associated functional types are shown below each elementary tree.

Furthermore, each tree can be interpreted as a function on the types of arguments it requires. For example, we say that the category of the verb *gave* (see Figure 1) is the elementary tree associated with it and not the primitive category *V*; the functional interpretation of its category, $NP \times NP \times NP \rightarrow S$, is a function expecting three trees rooted in *NP* and which returns an S-rooted tree. By combining elementary trees with substitution or adjunction, we can assign a structured category (the derived tree) and a functional interpretation to sequences of lexical items even in the cases when the sequence is discontinuous or when it does not define a constituent in the conventional sense. See

[2]It is known that TAGs are *weakly* equivalent to CCGs, i.e., they both generate the same sets of strings but not *strongly* because they do not assign the same structural descriptions

Figure 3 for some examples.

The coordination schema (&) combines two lexical strings with their structured categories and their functional types: $(l_1, \sigma_1, \tau_1)$ & $(l_2, \sigma_2, \tau_2) = (l, \sigma, \tau)$, where: $l_1, l_2, l$ are lexical strings; $\sigma_1, \sigma_2, \sigma$ are structured categories (trees); and $\tau_1, \tau_2, \tau$ are functional types.

The lexical strings in Figure 3 are *John eats*, *eats cookies*, *thinks John eats*, and *gave NP D book*. The first three strings are *contiguous* but the fourth string is *not contiguous*, in the sense that it is interrupted by one or more nonterminals, which are the argument positions for the associated functional type. We will say that the first three strings satisfy the Lexical String Contiguity (LSC) condition and the fourth string does not satisfy LSC[3]. Our structured categories are like un-Curried functions. LSC allows us to achieve Currying in a certain sense. Henceforth we will require that the structured categories that enter into coordination as well as the structured category resulting from coordination always satisfy LSC.

The coordination $(l_1, \sigma_1, \tau_1)$ & $(l_2, \sigma_2, \tau_2)$ succeeds if:

- the lexical strings $l_1$ and $l_2$ both satisfy LSC;
- the functional types are identical ($\tau_1 = \tau_2 = \tau$);
- the least nodes dominating $l_1$ in $\sigma_1$ and $l_2$ in $\sigma_2$ have the same label.

The resulting structured category, $\sigma = \sigma_1$ & $\sigma_2$, is obtained by:

1. equating the corresponding shared arguments in $\sigma_1$ and $\sigma_2$ (preserving linear precedence of arguments in $\sigma_1$ and $\sigma_2$;)
2. coordinating at the least nodes dominating $l_1$ and $l_2$;
3. collapsing the supertrees above the nodes at which coordination was made;
4. selecting the argument positions such that LSC holds for $\sigma$;
5. if the anchor of $\sigma_2$ is the same as the anchor if $\sigma_1$, then the anchor of $\sigma_2$ is erased and equated with the anchor of $\sigma_1$ (the decision to erase the anchor of $\sigma_2$ is based on the fact that the complements of the anchor must always be in the appropriate direction, on the right for English).

Now we will give a series of examples to illustrate the coordination schema. Figure 4 shows how *Mary a book* and *Susan a flower* can be coordinated to derive sentences like:

[3]LSC is not a syntactic constraint. It can be regarded as a *phonological* constraint in a certain sense. More details will be provided in an expanded version of this paper at a late date.

(8) John gave Mary a book and Susan a flower

In Figure 4, the tree corresponding to *gave Mary a book and Susan a flower* has been obtained by:

1. equating the NP nodes in $\sigma_1$ and $\sigma_2$;
2. coordinating the $VP$ nodes;
3. collapsing the supertrees above the $VP$ nodes;
4. selecting the left most $NP$ as argument.
5. erasing the anchor (*gave*) in $\sigma_2$ and equating the anchor node in $\sigma_2$ with the $V$ node in $\sigma_1$.

Similarly, the sentence,

(9) John likes and Bill hates bananas

is obtained by coordinating *John likes* and *Bill hates* (see Figure 5).

Note that *John likes* and *Bill hates* have been 'coordinated' but *John likes* and *Bill hates* have not been grouped together (i.e., bracketed as constituents). The phrase structure of the elementary trees has been preserved. This is in contrast to the analysis provided by CCG. CCG groups *John likes* and *Bill hates* as constituents and then invokes the coordination schema $X = XandX$ where $X$ is a constituent. *John likes* is turned into a constituent in a CCG by 'type-raising' *John* to a category which essentially encodes the information that *John* is in the subject position. In the elementary tree $\sigma_1$ the structure already encodes the information that whatever is substituted for the leftmost $NP$ in $\sigma_1$ is in the subject position.

Some additional examples follow.

(10) John eats cookies and drinks beer (see Figure 6)
(11) John cooked and ate the beans (see Figure 7)

Examples in which $\sigma_1$ and $\sigma_2$ invoke more than one elementary tree can also be handled in a similarly fashion. We will only give the examples and not show the trees due to the lack of space.

(12) *John thinks Mary and Bill believes Susan* will win.
(13) John gave *Mary three and Susan four* recently published novels.

So far, we have not said anything about the so-called gapped sentences, for example

(14) John likes apples and Bill pears.

It can be shown that the gapped sentences and other sentences related to gapped sentences have to be obtained by assuming that the left conjunct ($\sigma_1$) is built up to $S$, i.e., its functional type is a constant, $S$. A structured category, $\sigma$, (where the functional type is a constant $S$) can be viewed *retroactively* as corresponding to various functional types, for example, $NP \times likes \times NP \rightarrow S$.

Note that this functional type cannot be obtained by staring with $\sigma$ in Figure 2, where the functional type is $NP \times NP \rightarrow S$.

We now take $\sigma_2$ to be of the same functional type as $\sigma_1$ , i.e., $NP \times likes \times NP \rightarrow S$ and instantiate the coordination schema as before. Note that the lexical anchor of $\sigma_2$ is guaranteed to be the same as the lexical anchor of $\sigma_1$ because both have the functional type $NP \times likes \times NP \rightarrow S$. Hence, the anchor in $\sigma_2$ will be erased following the specification in the coordination schema described earlier[4]. We will not discuss all the details of this retroactive approach due to lack of space. This approach also handles sentences which are closely related to gapping, for example,

(15) John likes apples and pears (too)

The *too* is introduced to show that the interpretation is different from the case where *apples and pears* is obtained by *NP and NP* coordination. In (15) we have *S and S* coordination.

In summary, we have shown how constituency and functional types can be kept apart and still the kind of flexibility in the constituent structures that CCG allow can be realized by using lexicalized TAG with an associated coordination schema. In an expanded version of this paper, we will describe several details concerning (1) how this approach extends to coordination with cross-serial dependencies (as in Dutch) as well as as to languages with constituent orders different from English, (2) some processing implications and (3) the computation of the semantic interpretation using the machinery of synchronous TAG (Shieber and Schabes, 1990).

# REFERENCES


Aravind K. Joshi. 1987. An Introduction to Tree Adjoining Grammars. In A. Manaster-Ramer, editor, *Mathematics of Language*. John Benjamins, Amsterdam.

Yves Schabes, Anne Abeillé, and Aravind K. Joshi. 1988. Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars. In *Proceedings of the 12th International Conference on Computational Linguistics (COLING'88)*, Budapest, Hungary, August.

Yves Schabes. 1990. *Mathematical and Computational Aspects of Lexicalized Grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, August. Available as technical report (MS-CIS-90-48, LINC LAB179) from the Department of Computer Science.

Shieber, Stuart and Schabes, Yves, 1990. Synchronous Tree Adjoining Grammars. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90)*. Helsinki.

Mark Steedman. 1989. Intonation and syntax in spoken language systems. In *DARPA Speech and Natural Language Workshop*, Cape Cod, MA, October.

Mark Steedman. 1990. Gapping as constituent coordination. *Linguistics and Philosophy*, 13:207–263, April.


[4]This approach is inspired by Steedman's approach to gapping, which depends on type-raising. Steedman requires an additional stipulation to rule out certain unwanted consequences of type-raising. It appears that in our approach this problem can be avoided. Space does not permit us to give all the details.

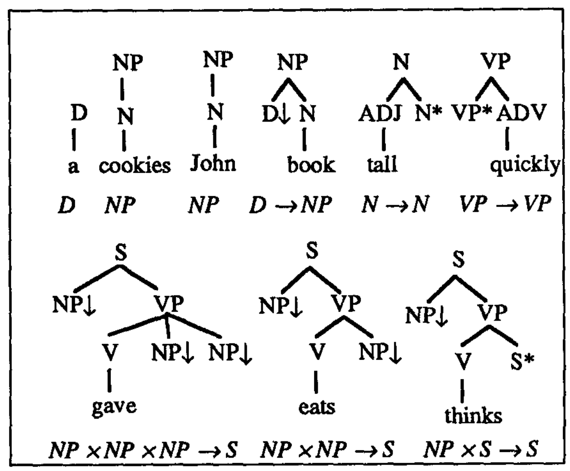


**Figure 1:** Examples of elementary trees with their functional type.

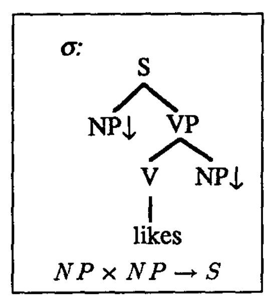


**Figure 2:** Structured category for *likes*

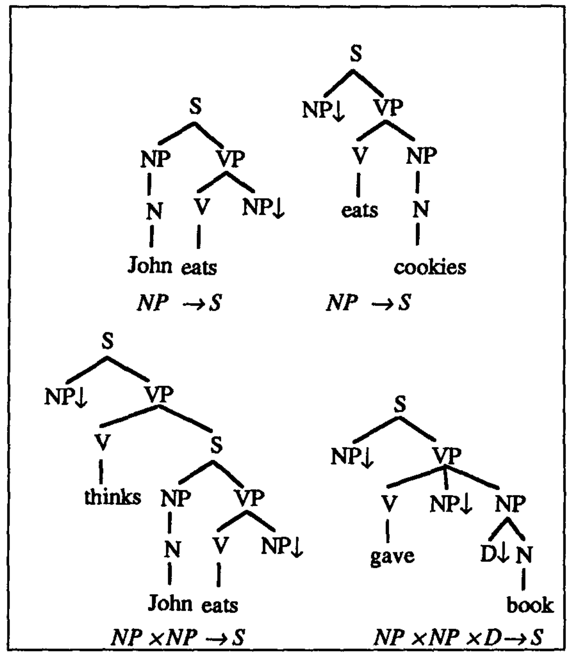


**Figure 3:** Examples of derived trees with their functional types.

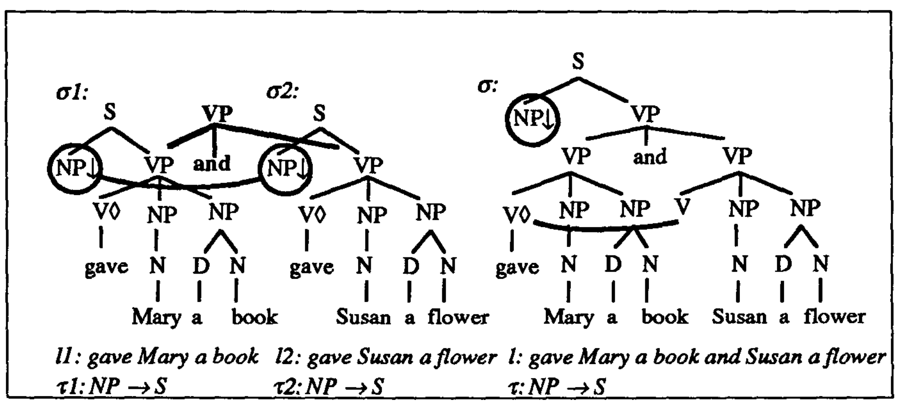


**Figure 4:** Coordination of *Mary a book* and *Susan a flower*.

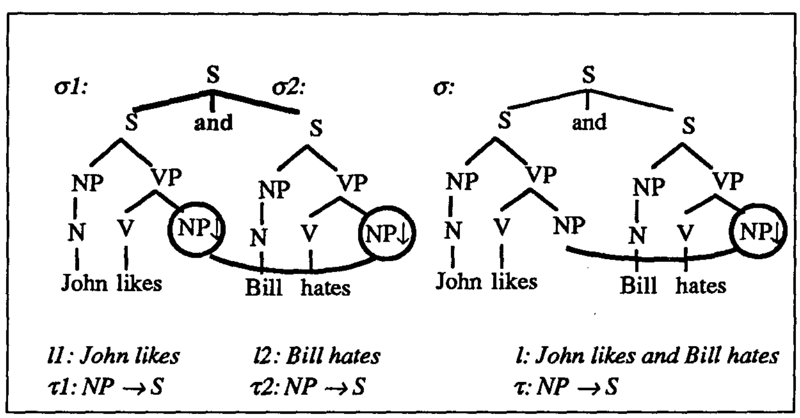


**Figure 5:** Coordination of *John likes* and *Bill hates*.

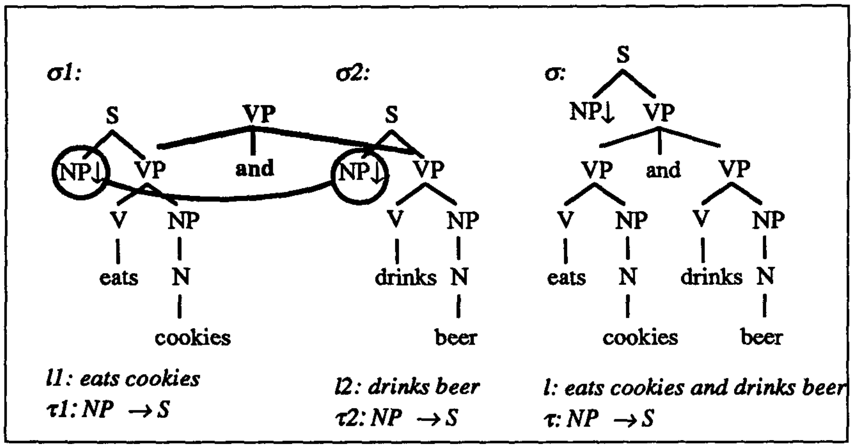


**Figure 6:** Coordination of *eats cookies* and *drinks beer* in (10).

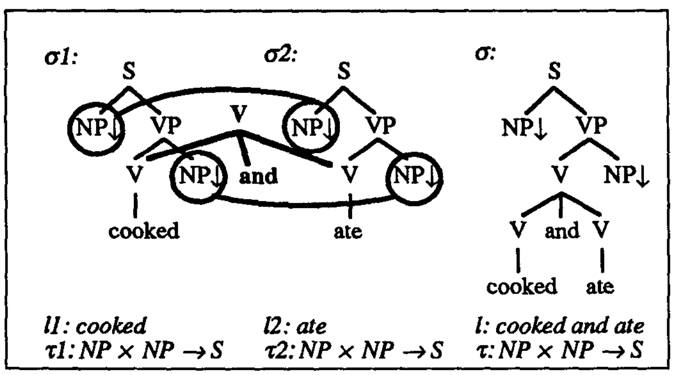


**Figure 7:** Coordination of *cooked* and *ate* in (11).