Coordination in Tree Adjoining Grammars: Formalization and Implementation

Anoop Sarkar and Aravind Joshi
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
{anoop,joshi}@linc.cis.upenn.edu

Abstract

In this paper we show that an account for coordination can be constructed using the derivation structures in a lexicalized Tree Adjoining Grammar (LTAG). We present a notion of derivation in LTAGs that preserves the notion of fixed constituency in the LTAG lexicon while providing the flexibility needed for coordination phenomena. We also discuss the construction of a practical parser for LTAGs that can handle coordination including cases of non-constituent coordination.

1 Introduction

Lexicalized Tree Adjoining Grammars (LTAG) and Combinatory Categorial Grammar (CCG) (Steedman, 1997) are known to be weakly equivalent but not strongly equivalent. Coordination schema have a natural description in CCG, while these schema have no natural equivalent in a standard LTAG.

In (Joshi and Schabes, 1991) it was shown that in principle it is possible to construct a CCG-like account for coordination in the framework of LTAGs, but there was no clear notion of what the derivation structure would look like. In this paper, continuing the work of (Joshi and Schabes, 1991), we show that an account for coordination can be constructed using the derivation structures in an LTAG.

Using the notions given in this paper we also discuss the construction of practical parser for LTAGs that can handle coordination including cases of non-constituent coordination. This approach has been implemented in the XTAG system (XTAG Research Group, 1995) thus extending it to handle coordination. This is the first full implementation of coordination in the LTAG framework.

2 LTAG

An LTAG is a set of trees (elementary trees) which have at least one terminal symbol on its frontier called the anchor. Each node in the tree has a unique address obtained by applying a Gorn tree addressing scheme, shown in the tree α(cooked) (Fig. 1). Trees can be rewritten using substitution and adjunction. A history of these operations on elementary trees in the form of a derivation tree can be used to reconstruct the derivation of a string recognized by a LTAG. In Fig. 1, the tree β(dried) adjoins into α(beans) and trees α(John) and α(beans) substitutes into α(cooked) to give a derivation tree for John cooked dried beans. Each node in the derivation tree is the name of an elementary tree. The labels on the edges denote the address in the parent node where a substitution or adjunction has occurred.

3 Trees as Structured Categories

In (Joshi and Schabes, 1991) elementary trees as well as derived trees in an LTAG were con-
Figure 1: Example of an LTAG and an LTAG derivation

Considered as structured categories defined as a 3-tuple of an elementary or derived tree, the string it spanned and the functional type of the tree, e.g. \( \langle \sigma_1, l_1, \tau_1 \rangle \) in Fig. 2. Functional types for trees could be thought of as defining un-Curried functions corresponding to the Curried CCG counterpart. A functional type was given to sequences of lexical items in trees even when they were not contiguous; i.e. discontinuous constituents were also assigned types. They were, however, barred from coordinating.

Figure 2: Structured Category for eats cookies

Coordination of two structured categories \( \sigma_1, \sigma_2 \) succeeded if the lexical strings of both categories were contiguous, the functional types were identical, and the least nodes dominating the strings spanned by the component tree have the same label. For example, in Fig. 3 the tree corresponding to eats cookies and drinks beer would be obtained by:

1. equating the NP nodes\(^1\) in \( \sigma_1 \) and \( \sigma_2 \), preserving the linear precedence of the arguments.

2. coordinating the VP nodes, which are the least nodes dominating the two contiguous strings.

3. collapsing the supertrees above the VP node.

4. selecting the leftmost NP as the lexical site for the argument, since precedence with the verb is maintained by this choice.

Figure 3: Coordination of eats cookies and drinks beer

The process of coordination built a new derived structure given previously built pieces of derived structure (or perhaps elementary structures). There is no clear notion of a derivation structure for this process.

4 Coordination in TAG

An account for coordination in a standard LTAG cannot be given without introducing a notion of sharing of arguments in the two lexically anchored trees because of the notion of locality of arguments in LTAG. In (1) for instance, the NP the beans in the “right node raising” construction has to be shared by the two elementary trees (anchored by cooked and ate respectively).

(1) ((Harry cooked) and (Mary ate)) the beans)

We introduce a notation that will enable us to talk about this more formally. In Fig. 4 the notation \( \downarrow \) denotes that a node is a non-terminal and hence expects a substitution operation to occur. The notation * marks the as the analysis presented in this paper are not deletion analyses.

\(^1\) This notion of sharing should not be confused with a deletion type analysis of coordination. The scheme presented in [Joshi and Schabes, 1991] as well
foot node of an auxiliary tree. Making this explicit we can view an elementary tree as a
ordered pair of the tree structure and a ordered set\(^2\) of such nodes from its frontier\(^3\),
e.g. the tree for *cooked* will be represented as \(\langle \alpha(\text{cooked}), \{1,2.2\} \rangle\). Note that this
representation is not required by the LTAG formalism. The second projection of this ordered
pair is used here for ease of explication. Let the second projection of the pair minus the
foot nodes be the substitution set. We will occasionally use the first projection of the
elementary tree to refer to the ordered pair.

Setting up Contractions. We introduce an
operation called build-contraction that takes
an elementary tree, places a subset from its
second projection into a contraction set and
assigns the difference of the set in the second
projection of the original elementary tree and
the contraction set to the second projection of
the new elementary tree. The contents of the
contraction set of a tree can be inferred from
the contents of the set in the second projec-
tion of the elementary tree. Hence, while we
refer to the contraction set of an elementary
tree, it does not have to be stored along with
its representation.

Fig. 4 gives some examples; each node
in the contraction set is circled in the figure.
In the tree \(\langle \alpha(\text{cooked}), \{1,2.2\} \rangle\) application of the operation on the NP node at
address 2.2 gives us a tree with the con-
traction set \(\{2.2\}\). The new tree is denoted
by \(\langle \alpha(\text{cooked})_{2.2}, \{1\} \rangle\), or \(\alpha(\text{cooked})_{2.2}\) for short. Placing the NP nodes at addresses 1
and 2.2 of the tree \(\alpha(\text{cooked})\) into the con-
traction set gives us \(\alpha(\text{cooked})_{\{1,2.2\}}\).

We assume that the anchor cannot be in-
volved in a build-contraction. This assump-
tion needs to be revised when gapping is con-
sidered in this framework (§8).

The Coordination Schema. We use the
standard notion of coordination shown in
Fig. 5 which maps two constituents of like
type, but with different interpretations, into
a constituent of the same type\(^4\).

![Figure 5: Coordination schema](image)

We add a new rewriting operation to the
LTAG formalism called conjoin\(^5\). While sub-
stitution and adjunction take two trees to give
a derived tree, conjoin takes three trees and
composes them to give a derived tree. One of
the trees is always the tree obtained by spe-
cializing the schema in Fig. 5 for a particular
category\(^6\).

Informally, the conjoin operation works as
follows: The two trees being coordinated are
substituted into the conjunction tree. This
notion of substitution differs from the tra-
ditional LTAG substitution operation in the
following way: In LTAG substitution, always
the root node of the tree being substituted
is identified with the substitution site. In
the conjoin operation however, the node sub-
stituting into the conjunction tree is given
by an algorithm, which we shall call Find-
Root that takes into account the contraction
sets of the two trees. FindRoot returns the
lowest node that dominates all nodes in
the substitution set of the elementary tree\(^7\).

---

\(^1\) The ordering is given by the fact that the elements of the set are Gorn addresses.

\(^2\) We shall assume there are no adjunction constraints in this paper.

\(^4\) In this paper, we do not consider coordination of unlike categories, e.g. *Pat is a Republican and proud of it.* \cite{sarkar:joshi:1996} discusses such cases, following \cite{borgersen:abeille:1992}.

\(^5\) Later we will discuss an alternative which replaces this operation by the traditional operations of substitution and adjunction.

\(^6\) The tree obtained will be a lexicalized tree, with the lexical anchor as the conjunction: *and, but, etc.*

\(^7\) This ensures the node picked by FindRoot always dominates a contiguous string in a derivation. This captures the string continuity condition that was used in \cite{joshi:shabes:1991}. A coordinated node will never dominate multiple foot nodes. Such a case oc-
e.g. \(\text{FindRoot}(\alpha(\text{cooked})_{2.2})\) will return the root node, i.e. corresponding to the \(S \text{ conj} S\) instantiation of the coordination schema. \(\text{FindRoot}(\alpha(\text{cooked})_{1.2.2})\) will return node address 2.1, corresponding to the \(V \text{ conj} V\) instantiation.

The conjoin operation then creates a contraction between nodes in the contraction sets of the trees being coordinated. The term contraction is taken from the graph-theoretic notion of edge contraction. In a graph, when an edge joining two vertices is contracted, the nodes are merged and the new vertex retains edges to the union of the neighbors of the merged vertices.\(^8\) The conjoin operation supplies a new edge between each corresponding node in the contraction set and then contracts that edge. As a constraint on the application of the conjoin operation, the contraction sets of the two trees must be identical.

Another way of viewing the conjoin operation is as the construction of an auxiliary structure from an elementary tree. For example, from the elementary tree \(\langle \alpha(\text{drinks}), \{1, 2.2\} \rangle\), the conjoin operation would create the auxiliary structure \(\langle \beta(\text{drinks})_{\{1\}}, \{2.2\} \rangle\) shown in Fig. 6. The adjunction operation would now be responsible for creating contractions between nodes in the contraction sets of the two trees supplied to it. Such an approach is attractive for two reasons. First, it uses only the traditional operations of substitution and adjunction. Secondly, it treats \(\text{conj } X\) as a kind of "modifier" on the left conjunct \(X\). We do not choose between the two representations but continue to view the conjoin operation as a part of our formalism.

For example, applying conjoin to the trees \(\text{Conj(and)}, \alpha(\text{eats})_{\{1\}}\) and \(\alpha(\text{drinks})_{\{1\}}\) gives us the derivation tree and derived structure for the constituent in (2) shown in Fig. 7.

\(\text{The Effects of Contraction.}\) One of the effects of contraction is that the notion of a derivation tree for the LTAG formalism has to be extended to an acyclic derivation graph.\(^9\) Simultaneous substitution or adjunction modifies a derivation tree into a graph as can be seen in Fig. 8.

If a contracted node in a tree (after the conjoin operation) is a substitution node, then the argument is recorded as a substitution into the two elementary trees as for example in the sentences (3) and (4).

(3) Chapman eats cookies and drinks beer.
(4) Keats steals and Chapman eats apples.

Fig. 8 contains the derivation and derived structures for (3) and Fig. 9 for (4). In Fig. 9 the derivation graph for sentence (4) accounts for the coordinations of the traditional nonconstituent "Keats steals" by carrying out the

\(^8\) Merging in the graph-theoretic definition of contraction involves the identification of two previously distinct nodes. In the process of contraction over nodes in elementary trees it is the operation on that node (either substitution or adjunction) that is identified.

\(^9\) We shall use the general notation derivation structure to refer to both derivation trees and derivation graphs.
coordination at the root, i.e. $S \text{ conj } S$. No constituent corresponding to “Keats steals” is created in the process of coordination.

The derived structures in Figs. 8 and 9 are difficult to reconcile with traditional notions of phrase structure. However, the derivation structure gives us all the information about dependency that we need about the constituents. The derivation encodes exactly how particular elementary trees are put together. Obtaining a tree structure from a derived structure built by the conjoin operation is discussed in (Sarkar and Joshi, 1996).

Considerations of the locality of movement phenomena and its representation in the LTAG formalism (Kroch and Joshi, 1986) can also now explain constraints on coordinate structure, such as across-the-board exceptions to the well known coordinate structure constraint, see Fig. 10. Also in cases of unbounded right node raising such as Keats steals and Chapman eats apples, Chapman thinks simply adjoins into the right conjunct of the coordinate structure.

5 Contractions on Anchors

An LTAG along with the operations of substitution and adjunction also has the implicit operation of lexical insertion (represented as the diamond mark in Fig. 11). Under this view, the LTAG trees are taken to be templates. For example, the tree in Fig. 11 is now represented as $\langle \alpha(eat), \{1,2,1,2\} \rangle$.

Also in cases of unbounded right node raising such as Keats likes and Chapman thinks Mary likes beans, Chapman thinks simply adjoins into the right conjunct of the coordinate structure.

McCawley (1982) raised the heterodox view that a discontinuous constituent structure should be given for right node raising cases, having the same notion of constituency as our approach. However, no conditions on the construction of such a structure was given. In fact, his mechanism also covered cases of parenthetical placement, scrambling, relative clause extraposition and heavy NP shift.

10 A comparison of this paper’s approach with the derivational machinery in CCG and the devices of 3-D coordination is done in (Sarkar and Joshi, 1996).
mond mark is a preterminal. Previous uses of conjoin applied to two distinct trees. If the lexicalization operation is to apply simultaneously, the same anchor projects two elementary trees from the lexicon. The process of contraction ensures that the anchor is placed into a pair of LTAG tree templates with a single lexical insertion.

Gapping. Using this extension to conjoin, we can handle sentences that have the “gapping” construction like sentence (5).

(5) John ate bananas and Bill strawberries.

The conjoin operation applies to copies of the same elementary tree when the lexical anchor is in the contraction set. For example, let $\alpha(eats)$ be the tree selected by eats. The coordination of $\alpha(eats)\{2,1\}$ with a copy of itself and the subsequent derivation tree is depicted in Fig. 12.

Figure 12: Handling the gapping construction using contractions.

An extension of the approach here will be to permit the conjoin operation to create contractions on all the nodes in contraction sets that it dominates during a derivation, allowing us to recognize cases of gapping such as: John wants Penn to win and Bill, Princeton.

Coordinating Ditransitive verbs. In sentence (6) if we take the position that the string Mary a book is not a constituent (i.e. give has a structure as in Fig. 13), then we can use the notion of contraction over the anchor of a tree to derive the sentence in (6). The structure we derive is shown in Fig. 14.

(6) John gave Mary a book and Susan a flower.

(7) John loves Mary and Bill too.

6 Parsing Issues

This section discusses parsing issues that arise in the modified TAG formalism that we have presented. We do not discuss general issues in parsing TAGs, rather we give the appropriate modifications that are needed to the existing Earley-type parsing algorithm for TAGs due to Schabes and Joshi (1988).

The algorithm relies on a tree traversal that scans the input string from left to right while recognizing the application of the conjoin operation. The nodes in the elementary trees are visited in a top-down left to right manner.

12 In English, following Ross (1970), the anchor goes to the left conjunct.

13 Whether this should be derived syntactically is controversial, for example, see Steedman, 1990.
Each dot in Fig. 15 divides the tree into a left context and a right context, enabling the algorithm to scan the elementary tree while trying to recognize possible applications of the conjoin operation.

The derived structure corresponding to a coordination is a composite structure built by applying the conjoin operation to two elementary trees and an instantiation of the coordination schema. The algorithm never builds derived structures. It builds the derivation by visiting the appropriate nodes during its tree traversal in the following order (see Fig. 16).

The algorithm must also compute the correct span of the string for the nodes that have been identified via a contraction. Fig. 16 gives the possible scenarios for the position of nodes that have been linked by a contraction. When foot nodes undergo contraction, the algorithm has to ensure that both the foot nodes share the subtree pushed under them, e.g. 9····10 and 9′····10′ in Fig. 16(a). Similarly, when substitution nodes undergo contraction, the algorithm has to ensure that the tree recognized due by predicting a substitution is shared by the nodes, e.g. 11····12 and 11′····12′ in Figs. 16(b) and 16(c). The traversals 9····10 should span the same length of the input as 9′····10′, similarly for 11····12 and 11′····12′. Various positions for such traversals is shown in Fig. 16. A derivation is valid if the input string is accepted and each node in a contraction spans a valid substring in the input. The complete and formal description of the parsing algorithm is given in (Sarkar and Joshi, 1996).

7 Conclusion

We have shown that an account for coordination can be given in a LTAG while maintaining the notion of a derivation structure which is central to the LTAG approach. We showed that fixed constituency can be maintained at the level of the elementary tree while accounting for cases of non-constituent coordination and gapping. We discussed the construction of a practical parser for LTAG that can handle these cases of coordination.

References

[Jorgensen and Abeillé1992] Jorgensen, H. and A. Abeillé. 1992. Coordination of “Unlike” Categories in TAG. In Proceedings of the 2nd TAG Workshop, Philadelphia, PA.

[Joshi and Schabes1991] Joshi, Aravind and Yves Schabes. 1991. Fixed and flexible phrase structure: Coordination in Tree Adjoining Grammar. Presented at the DARPA Workshop on Spoken Language Systems. Asilomar, CA.

[Kroch and Joshi1986] Kroch, A. and A. K. Joshi. 1986. Analyzing extraposition in a tree adjoining grammar. In G. Huck and A. Ojeda, editors, Syntax and Semantics: Discontinuous Constituents. Academic Press, New York.

[McCawley1982] McCawley, James. 1982. Parentheticals and discontinuous constituent structure. Linguistic Inquiry, 13(1):91–106.

[Ross1970] Ross, John. 1970. Gapping and the order of
constituents. In M. Bierwisch and K. Heidolph, editors, Progress in Linguistics. Mouton, The Hague.

[Sarkar and Joshi1996] Sarkar, Anoop and Aravind Joshi. 1996. Handling Coordination in a Tree Adjoining Grammar. Technical report, Dept. of Computer and Info. Sc., Univ. of Pennsylvania, Philadelphia, PA.

[Schabes and Joshi1988] Schabes, Yves and Aravind K. Joshi. 1988. An Earley-type parsing algorithm for tree adjoining grammars. In 26th Meeting of the Association for Computational Linguistics, Buffalo, NY.

[Steedman1990] Steedman, Mark. 1990. Gapping as constituent coordination. Linguistics and Philosophy, 13:207–264.

[Steedman1997] Steedman, Mark. 1997. Surface Structure and Interpretation: Unbounded and Bounded Dependency in Combinatory Grammar. Linguistic Inquiry monograph (to appear). MIT Press.

[XTAG Research Group1995] XTAG Research Group. 1995. A Lexicalized Tree Adjoining Grammar for English. Technical report, IRCS Tech Rpt. 95-03, University of Pennsylvania, Philadelphia, PA.