DIGITAL ACCESS TO SCHOLARSHIP AT HARVARD
DASH.HARVARD.EDU

HARVARD LIBRARY
Office for Scholarly Communication

Restricting the weak-generative capacity of synchronous tree-adjoining grammars.

Citation
Stuart M. Shieber. Restricting the weak-generative capacity of synchronous tree-adjoining grammars. Computational Intelligence, 10(4):371-385, November 1994. Also available as cmp-lg/9404003. The definitive version is available at www.blackwell-synergy.com.

Published Version
http://dx.doi.org/10.1111/j.1467-8640.1994.tb00003.x

Permanent link
http://nrs.harvard.edu/urn-3:HUL.InstRepos:2032465

Terms of Use
This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA

Share Your Story

The Harvard community has made this article openly available.
Please share how this access benefits you. Submit a story .

Accessibility

RESTRICTING THE WEAK-GENERATIVE CAPACITY OF SYNCHRONOUS TREE-ADJOINING GRAMMARS

STUART M. SHIEBER

Abstract. The formalism of synchronous tree-adjoining grammars, a variant of standard tree-adjoining grammars (TAG), was intended to allow the use of TAGs for language transduction in addition to language specification. In previous work, the definition of the transduction relation defined by a synchronous TAG was given by appeal to an iterative rewriting process. The rewriting definition of derivation is problematic in that it greatly extends the expressivity of the formalism and makes the design of parsing algorithms difficult if not impossible.

We introduce a simple, natural definition of synchronous tree-adjoining derivation, based on isomorphisms between standard tree-adjoining derivations, that avoids the expressivity and implementability problems of the original rewriting definition. The decrease in expressivity, which would otherwise make the method unusable, is offset by the incorporation of an alternative definition of standard tree-adjoining derivation, previously proposed for completely separate reasons, thereby making it practical to entertain using the natural definition of synchronous derivation. Nonetheless, some remaining problematic cases call for yet more flexibility in the definition; the isomorphism requirement may have to be relaxed. It remains for future research to tune the exact requirements on the allowable mappings.

Keywords: Synchronous tree-adjoining grammars, weak-generative capacity, machine translation, natural-language semantics.

1. Introduction

The formalism of synchronous tree-adjoining grammars (Shieber and Schabes, 1990), a variant of standard tree-adjoining grammars (TAG), was intended to allow the use of TAGs for language transduction in addition to language specification. Synchronous TAGs specify relations between language pairs; each language is specified with a standard TAG, and the pairing between strings in the separate languages is specified by synchronizing the two TAGs through linking pairs of elementary trees.

This paper concerns the formal definitions underlying synchronous tree-adjoining grammars. In previous work (Shieber and Schabes, 1990), the definition of the transduction relation defined by a synchronous TAG was given by appeal to an iterative rewriting process, much like the iterative rewriting of sentential forms defined by a context-free grammar except that the syntactic objects generated by the rewriting process were derived trees rather than strings. This sort of rewriting definition of derivation is problematic for several reasons. First, the weak-generative expressivity of TAGs is increased through the synchronization in the sense that the

Date: October 13, 1993.

This paper appeared in *Computational Intelligence*, volume 10, number 4, pages 371–385, and is available through the Computation and Language e-print archive as cmp-lg/9404003.

projection of the string pairs onto a single component, although the strings in that component are specified with a TAG, may not form a tree-adjoining language (TAL). Second, the lack of a simple recursive characterization of the derivation — a role filled by derivation trees for standard TAGs — makes the design of parsing algorithms difficult if not impossible.

In this paper, we describe how synchronous TAG derivation can be redefined so as to eliminate these problems. The redefinition relies on an independent redefinition of the notion of tree-adjoining derivation (Schabes and Shieber, 1994) that was motivated completely independently of worries about the generative capacity of synchronous TAGs, but which happens to solve this problem in an elegant manner. Furthermore, it allows for existing parsing algorithms to be generalized to synchronous TAG transduction in a natural way. However, because certain derivations in the rewriting sense have no analogue under the new definition, some linguistic analyses may no longer be statable. We comment on some possible negative ramifications of this fact.

2. The Rewriting Definition of Derivation

The original definition of derivation for synchronous TAGs was based on the iterative rewriting of one derived tree pair into another. In this section, we provide a more precise description of this approach to derivation, along with a discussion of its problems. First, however, we digress to discuss some notational issues.

2.1. **Notation.** We assume in this and later sections a general familiarity with tree-adjoining grammars and their formal foundations, as described, for instance, by Vijay-Shanker (1987).

We will use the following notational conventions for synchronous TAGs and related notions. A synchronous TAG G will be given as a set of triples $\{\langle L_i, R_i, \frown_i\rangle\}$ where the L_i and R_i are elementary trees, both initial and auxiliary, forming two component TAGs $G_L = \{L_i\}$ and $G_R = \{R_i\}$, and $\frown_i$ is the linking relation between tree addresses in L_i and R_i. Such a grammar is intended to define a language of pairs $L(G) = \{\langle l_i, r_i\rangle\}$; the exact manner in which $L(G)$ is determined is the subject of this paper. We will use the notation x_L and x_R to notate the projection of a pair x onto its left and right components, respectively, and generalize this notation to the first and second components of a triple and pointwise on sets of pairs and triples. Thus, the notations G_L and G_R previously introduced for the left and right component grammars are consistent with this notation.

2.2. **The Rewriting Process.** The rewriting process proceeds by choosing an initial tree pair $\langle I_L, I_R, \frown\rangle$ to be the *current derived tree pair* and repeatedly performing the following steps:

(1) Choose a link $t_L \frown t_R$ between two nodes in the current derived tree pair.
(2) Choose an auxiliary tree pair $\langle A_L, A_R, \frown'\rangle$ from the grammar such that A_L can adjoin at t_L in I_L and A_R can adjoin at t_R in I_R.
(3) Modify the current derived tree pair by adjoining the chosen trees at the end of the chosen link, yielding the modified derived tree pair

$$\langle I_L[A_L/t_L], I_R[A_R/t_R], \frown''\rangle \qquad .$$

This becomes the new current derived tree pair.

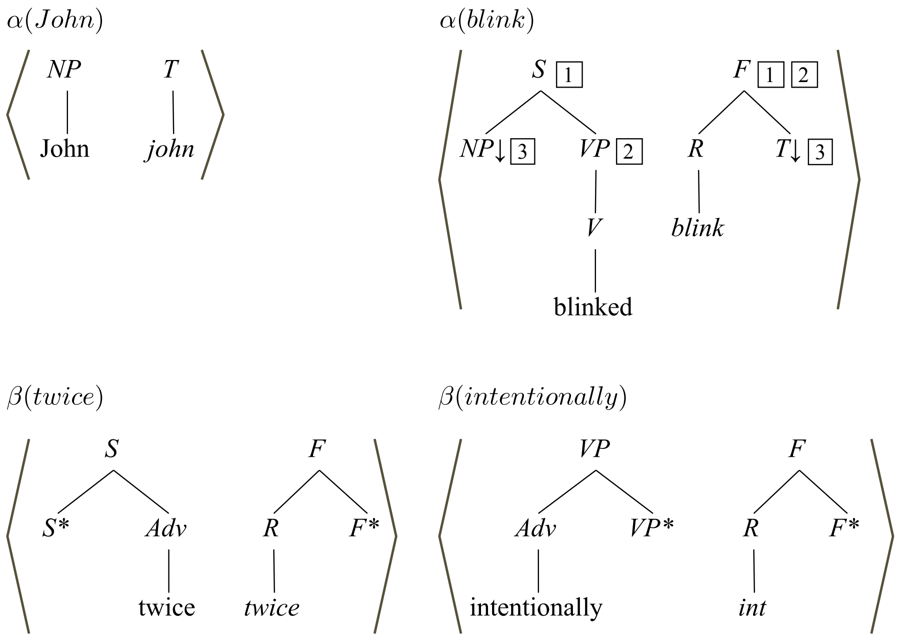


FIGURE 1. A synchronous TAG that describes the semantic ambiguity of the sentence 'John intentionally blinked twice'.

The operation $I[A/t]$ used above takes a tree I, an auxiliary tree A, and an address t in I and yields the result of adjoining A at address t in I. (The generalization to allow for substitution as well as adjunction as a primitive operation — both in this notation and the definition of derivation — should be clear.) A formal definition for this operation is given by Vijay-Shanker (1987, page 15) and by Schabes and Shieber (1994, appendix).

The definition of the link relation in the derived tree pair $\frown''$ is as follows: All links in $\frown$ and $\frown'$ are included in $\frown''$ (after suitable readdressing) except that the chosen link in $\frown$ is not itself included in $\frown''$. Other links that impinge on the nodes at the end of the chosen link are retained in the derived tree pair; they link to the root or foot of the newly adjoined tree as determined by whether the link itself is viewed as impinging on the top or the bottom of the node.[1]

2.3. **An Example of Rewriting.** By way of example, we present a sample synchronous TAG that transduces between a tiny fragment of English and a corresponding "logical form" semantic representation.

[1]In previous work, links were typically thought of as impinging on the top of a node unless otherwise stated. We will retain that convention here. Further flexibility can be obtained by allowing each link to specify whether it links to the top or bottom of the nodes. Thus, the link relation $\frown_i$ in a triple $\langle L_i, R_i, \frown_i\rangle$ can be thought of as being of type $dom(L_i) \times \{\uparrow,\downarrow\} \times dom(R_i) \times \{\uparrow,\downarrow\}$, where $dom(A)$ is the set of tree addresses in the tree A and $\uparrow$ and $\downarrow$ serve as markers to specify whether the link impinges on the top or bottom, respectively, of the specified node. All of this machinery becomes superfluous, however, in the context of the natural definition of derivation given in Section 3.

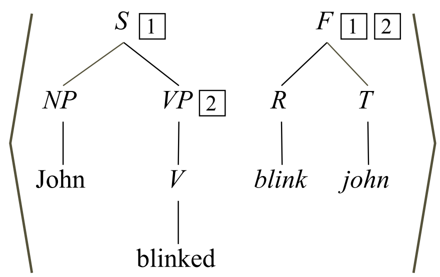


FIGURE 2. A tree pair derived by operation of the $\alpha(John)$ tree at link $\boxed{3}$ in the $\alpha(blink)$ tree. The pair specifies the transduction between the string 'John blinks' and its logical form $blink(john)$.

Figure 1 shows the sample synchronous TAG composed of a set of tree pairs, each with a left element that is part of an English TAG fragment and a right element that is part of a TAG fragment for the logical form language. Thus, the tree pair labeled $\alpha(John)$ pairs a noun phrase (*NP*) initial tree dominating the proper noun 'John' with a logical term (T) dominating the constant *john*. Similarly, the tree pair $\alpha(blink)$ pairs a verb tree for 'blinked' with a tree for a formula (F) constructed as the predication of the relation (R) given by the symbol *blink* to an unspecified argument term.

Rather than present the elements of the grammar as triples, we notate the links between nodes with diacritics. Thus, the $\alpha(blink)$ tree pair implicitly incorporates the link relation between tree addresses given by

$$\begin{gathered} \epsilon \frown \epsilon \\ 2 \frown \epsilon \\ 1 \frown 2 \end{gathered}$$

These three links are marked with the diacritics $\boxed{1}$, $\boxed{2}$, and $\boxed{3}$ respectively.

The $\boxed{3}$ link, for instance, connects the *NP* node in the left tree at address 1 with the T node in the right at address 2, thereby allowing the two trees of another tree pair to operate respectively at these two nodes. Since the two nodes are substitution nodes (as conventionally marked by the $\downarrow$), the operations on this link would be substitutions at both ends. For example, the initial tree pair $\alpha(John)$ can operate at this link, yielding the tree pair given in Figure 2. Note that the remaining links in the $\alpha(blink)$ tree labeled $\boxed{1}$ and $\boxed{2}$ are preserved in the derived tree pair.

Continuing on in this way, the resultant derived tree pair can be further acted upon, say, by the base pair $\beta(twice)$, whose elements can adjoin at the ends of the $\boxed{1}$ link, yielding the derived tree pair in Figure 3a. The issue of how to handle multiple links impinging on the same node becomes relevant here, since the right end of the remaining link $\boxed{2}$ in the derived tree pair impinges on a node at which adjunction has just occurred. Should the link now impinge on the root or the foot node of the tree adjoined at that node? We place the link at the root, as stipulated above, so that further rewriting of the $\boxed{2}$ link, say with the adverbial tree pair $\beta(intentionally)$ leads to the derived tree pair in Figure 3b, corresponding to the string 'John intentionally blinked twice'. In the associated logical form,

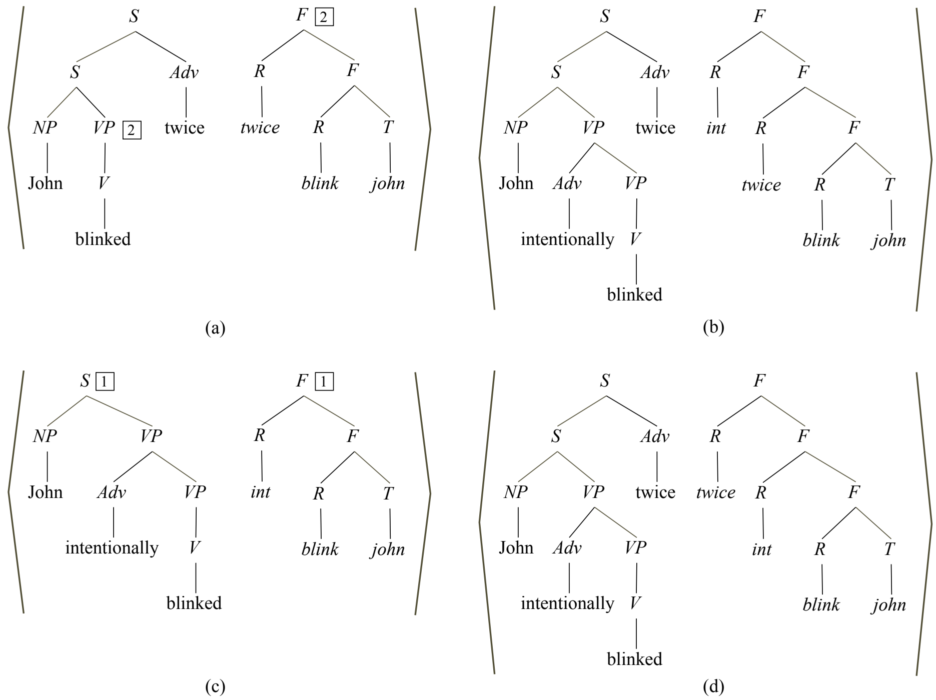


FIGURE 3. Derived tree pairs from the grammar of Figure 1. The derivation of the meaning $int(twice(blink(j)))$ proceeds through the derived tree pairs in (a) and (b). The derivation of the meaning $twice(int(blink(j)))$ proceeds through the derived tree pairs in (c) and (d).

the predication of int has scope over the proposition $twice(blink(john))$, and the sentence is taken to describe a single intentional act of blinking twice. Had the two links been rewritten in the other order — link $\boxed{2}$ first, yielding the pair in Figure 3c, and then link $\boxed{1}$ yielding the pair in Figure 3d — the generated logical form $twice(int(blink(j)))$ describes two intentional acts each of single blinkings.

Thus, this grammar manifests the ambiguity in the sentence 'John intentionally blinked twice'. Note that the ambiguity arises from the ability to perform two rewriting steps at the same node, the root F node in the logical form tree $\alpha(blink)_R$ corresponding to the word 'blinked'.

2.4. **Problems with the Rewriting Definition.** There are two problems with the rewriting definition of synchronous TAGs, having to do with the expressivity and implementability of the formalism under that definition.

2.4.1. *Expressivity.* Synchronous TAGs under this definition may specify non-tree-adjoining languages. More precisely stated, given a grammar G, although, by definition, $L(G_L)$ is a tree-adjoining language, $L(G)_L$ may not be.

A simple example of a synchronous TAG that can be projected onto a non-TAL is given in Figure 4. This grammar specifies the string relation that pairs all strings of the form $a^n b^n c^n d^n e^n f^n g^n h^n$ with the empty string. Its projection onto its first

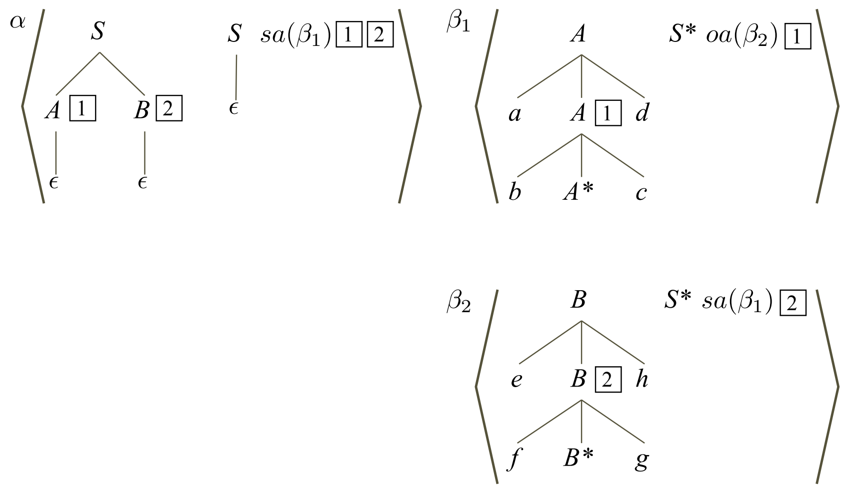


FIGURE 4. A synchronous TAG for a non-tree-adjoining language $a^n b^n c^n d^n e^n f^n g^n h^n$.

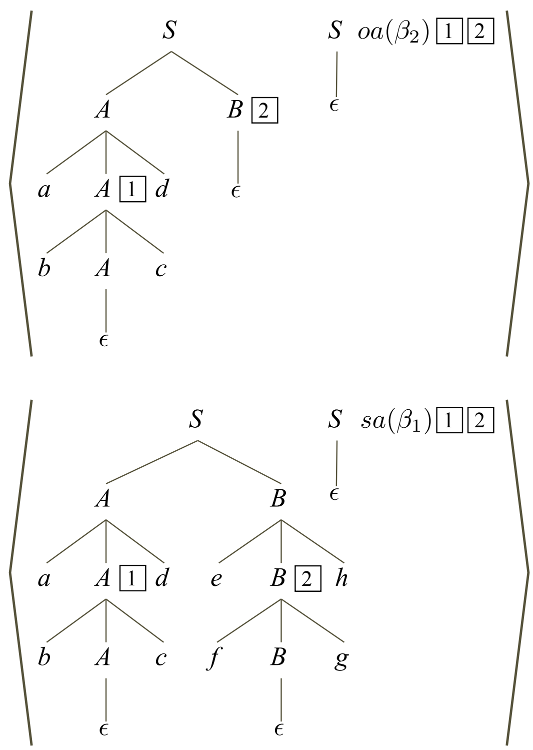


FIGURE 5. Steps in the derivation of *abcdefgh*. The left derived tree pair has a remaining obligatory adjoining constraint, which when satisfied yields the right derived tree pair.

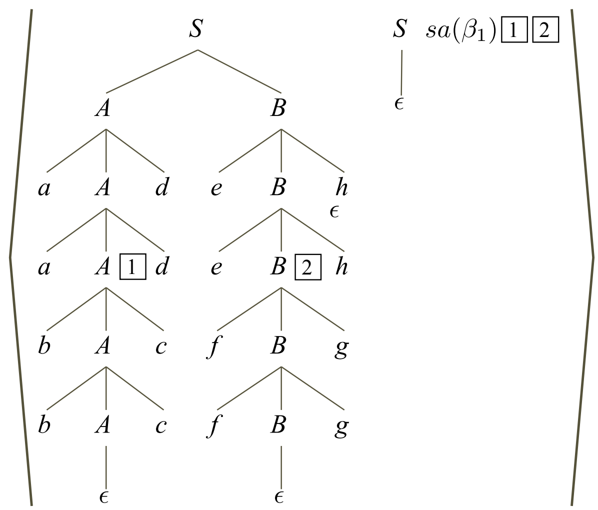


FIGURE 6. Derived tree pair for $a^2b^2c^2d^2e^2f^2g^2h^2$.

component is, therefore, a non-tree-adjoining language. Figure 5 shows the steps in the derivation of the $n = 1$ case. The derived tree pair for the $n = 2$ case is given in Figure 6.

2.4.2. *Implementability.* In addition to the expressivity problem, there is no natural way to use a synchronous grammar for transduction under this definition. To use a synchronous TAG G for transduction, a given string w_L is to be transduced to w_R just in case $\langle w_L, w_R\rangle \in L(G)$. This requires, intuitively speaking, parsing of the string w_L relative to G_L yielding a derivation D_L, reconstruction of the synchronous (rewriting) derivation D_S from D_L, and finally, generation of the string w_R according to this reconstructed derivation. Schematically, the process can be depicted as proceeding thus:

$$w_L \longrightarrow D_L \longrightarrow D_S \longrightarrow w_R$$

Unfortunately, the structure of a synchronous derivation bears no uniform relationship to the kind of derivation postulated for standard TAGs. (This point is discussed further in the next section.) Thus, if a standard TAG parsing algorithm is used for the first step in the process (so that D_L is a traditional TAG derivation tree), the second step is not well defined. It is therefore not clear how synchronous TAGs can be effectively used under this definition of derivation.

Note that this point is independent of whether the three conceptual phases of processing are interleaved in time. The possibility to interleave the computations of the phases does not make their definition any simpler.

3. The Natural Definition of Derivation

The notion of derivation just presented for synchronous TAGs is quite nonstandard for the TAG literature in being "flat" and rewriting oriented. Recall that the standard definition of TAG derivation, due to Vijay-Shanker (1987), is hierarchically structured in terms of *derivation trees*, trees that serve to characterize the operations required to construct a particular derived tree, and hence its yield.

TAG derivation trees are composed of nodes, conventionally notated as η, possibly in its subscripted variants. The parent of a node η in a derivation tree will be written $parent(\eta)$, and the tree that the node marks adjunction of will be notated $tree(\eta)$. The tree $tree(\eta)$ is to be adjoined into its parent $tree(parent(\eta))$ at an address specified on the arc in the tree linking the two; this address is notated $addr(\eta)$. (Of course the root node has no parent or address; the *parent* and *addr* functions are partial.)

A derivation tree is well-formed if for each arc in the derivation tree from η to $parent(\eta)$ labeled with $addr(\eta)$, the tree $tree(\eta)$ is an auxiliary tree that can be adjoined at the node $addr(\eta)$ in $tree(parent(\eta))$. (Alternatively, $tree(\eta)$ is an initial tree that can be substituted at the node $addr(\eta)$ in $tree(parent(\eta))$.) Furthermore, and without loss of expressivity, it is standard to exclude multiple sibling arcs specifying operations at the same tree address in the same tree. This exclusion makes the definition of the derived tree for a given derivation tree determinate.

A derivation tree specifies a derived tree by virtue of the normal definitions for adjunction and substitution. The language of a TAG G is then the set of strings that are the yields of derived trees specified by derivation trees that are well-formed according to G. We define the function $\mathcal{D}$ from derivation trees to the derived trees they specify, according to the following recursive definition:

$$\mathcal{D}(D) = \begin{cases} tree(\eta) \\ \qquad \text{if } D \text{ is a trivial tree of one node } \eta \\ tree(\eta)[\mathcal{D}(D_1)/t_1, \mathcal{D}(D_2)/t_2, \ldots, \mathcal{D}(D_k)/t_k] \\ \qquad \text{if } D \text{ is a tree with root node } \eta \\ \qquad\quad \text{and with } k \text{ child subtrees } D_1, \ldots, D_k \end{cases}$$

Here $I[A_1/t_1, \ldots, A_k/t_k]$ specifies the simultaneous adjunction (or substitution) of trees A_1 through A_k at t_1 through t_k, respectively, in I. Using the definitions of Vijay-Shanker (1987), this is well defined only as long as the t_i are disjoint, hence the need for the aforementioned exclusion.

A definition along these lines for synchronous TAGs would be quite natural. We would have derivation trees that specify at each node an elementary tree *pair*, with arcs labeled by *pairs* of tree addresses (such that the two addresses are linked in the parent elementary tree pair). A function from derivation trees to the derived tree pairs they specify — a generalization of the $\mathcal{D}$ function defined above — would then be used to generate the derived trees and the language of a synchronous grammar.

It should be obvious that such a synchronous derivation tree can be trivially restated as a pair of standard derivation trees, further simplifying the definition of synchronous TAG derivation. This leads to the following definition of synchronous TAG derivation. A derivation is a pair $\langle D_L, D_R \rangle$ where

(1) D_L is a well-formed derivation tree relative to G_L.
(2) D_R is a well-formed derivation tree relative to G_R.
(3) D_L and D_R are isomorphic. That is, there is a one-to-one onto mapping f from the nodes of D_L to the nodes of D_R that preserves dominance, i.e., if $f(\eta_l) = \eta_r$ then $f(parent(\eta_l)) = parent(\eta_r)$.

(4) The isomorphic operations are sanctioned by links in tree pairs. That is, if $f(\eta_l) = \eta_r$, then there is a tree pair $\langle tree(\eta_l), tree(\eta_r), \frown' \rangle$ in G. Furthermore, if η_l has a parent, then there is a tree pair

$$\langle tree(parent(\eta_l)), tree(parent(\eta_r)), \frown \rangle$$

in G and $addr(\eta_l) \frown addr(\eta_r)$.

This, then, is the most natural definition of synchronous tree-adjoining derivation, as it is the natural generalization of the definition of derivation for standard TAGs. It merely requires that there be two derivations that are separately well-formed and appropriately synchronized as specified by the links.

Several aspects of this definition are noteworthy. First, the derived tree pair for a derivation $\langle D_L, D_R \rangle$ is $\langle \mathcal{D}(D_L), \mathcal{D}(D_R) \rangle$. Second, the definition does not require extra linking information specifying whether the link impinges on the top or bottom of the linked nodes. It is completely declarative; no vestiges remain of the rewriting definition. Finally, it solves the two problems of expressivity and implementability mentioned above, as described in the next section.

4. Advantages of the Natural Definition

We show in this section that the natural definition of synchronous derivation solves the two problems described in Section 2.4.

4.1. **Expressivity.** Under the revised definition of synchronous derivation, only tree-adjoining languages can be expressed by a synchronous TAG. To see why, we look first at the problematic example of Figure 4, and then turn to a general argument.

Under the new definition, adjoining constraints are no longer inherited in an overall derived tree being generated incrementally in the flat rewriting process. Rather, they apply to the auxiliary trees that directly adjoin to the node. Thus, in the grammar of Figure 4, the links in the auxiliary trees can never be operated on. For instance, the link in β_1 requires β_2 to be adjoined there, but its corresponding left half cannot adjoin at the left end of the link. Similarly, the link in β_2 is useless as well. Thus, the only well-formed derivation is the one with no adjunctions whatsoever; the language of the grammar includes the single string pair $\langle \epsilon, \epsilon \rangle$ generated by its initial tree pair.

In general under the revised definition, the left-projection language, say, of a synchronous TAG is specifiable by a pure TAG by simply mapping any adjoining constraints on the right trees to corresponding ones on the linked nodes on the left and projecting the grammar on its left component. (The example of Figure 4, so projected, is the normal TAG given in Figure 7, which specifies the language containing only the empty string as expected.)

Alternatively, the TAL nature of synchronous TAGs under this definition can be easily shown by reduction to tree-set-local multicomponent TAGs (MCTAG), which are known to generate only tree-adjoining languages.[2] Each elementary tree pair in the synchronous TAG corresponds to an elementary tree set in the MCTAG. To ensure that left-hand trees are not adjoined into right-hand trees and vice versa, the node labels on the left- and right-hand trees are uniformly renamed apart. Each

[2]The observation that synchronous TAGs under the new definition should be reducible to MCTAG was brought to our attention by Owen Rambow.

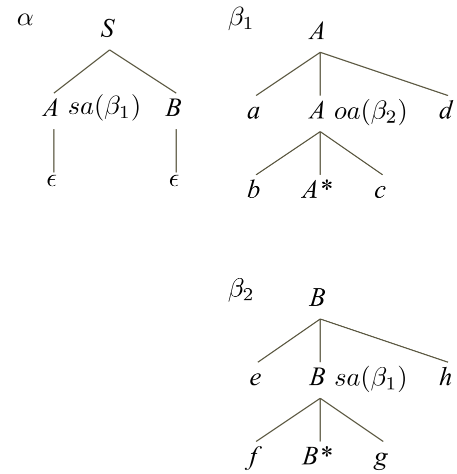


FIGURE 7. The left projection of the grammar of Figure 4.

node in a left-hand tree is marked with a selective adjoining constraint that allows adjunction only of certain elementary tree sets. For each link that impinges on the node, and each tree pair that can operate on that link, the corresponding tree set is allowed by the SA constraint. Similar constraints are added to each right-hand node. Finally, for each pair of nonterminals that root the trees in an initial tree pair, a new elementary tree is constructed rooted in a new nonterminal symbol not used elsewhere with two nonterminal children labeled by the left and right root nonterminals of the initial tree pair and which are to be filled by substitution.

Since any synchronous TAG can be reduced to a tree-set-local MCTAG, the languages generated by synchronous TAGs are at most the tree-adjoining languages. The converse inclusion is obvious.

4.2. **Implementability.** Another advantage of the new definition of synchronous derivation is in its utility for implementation of synchronous TAG transducers. Recall that under the rewriting definition, the structure of a synchronous derivation bears no uniform relationship to the kind of derivation postulated for standard TAGs and therefore recovered by standard TAG parsing algorithms. Thus, the second step in the schematic process

$$w_L \longrightarrow D_L \longrightarrow D_S \longrightarrow w_R$$

is not well defined. Under the natural definition, however, the synchronous derivation D_S is just $\langle D_L, D_R \rangle$. This close relation between a synchronous TAG derivation and derivations for the left and right projected grammars makes synchronous transduction straightforward. Any method for parsing that generates a standard derivation tree for a grammar can be applied to parse a string w_L relative to the left projection grammar. The resultant derivation is isomorphic to the derivation tree for the right projection grammar, where the mapping is given directly by the synchronous grammar. The right projection derivation is thus easily constructed, and the corresponding derivation tree can be computed directly. Schematically, the

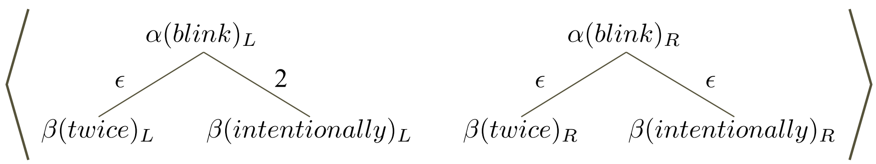


FIGURE 8. Derivation tree pair for the grammar of Figure 1.

process looks like this:

$$w_L \longrightarrow D_L \longrightarrow D_S(= \langle D_L, D_R \rangle) \longrightarrow D_R \longrightarrow w_R$$

This methodology applies even under the view of synchronous TAG derivations to be described in Section 5.1. For instance, Schabes and Shieber (1994) describe a parsing method for standard TAGs that can be used to construct derivation trees on the fly while parsing. A simple modification of the method can construct the isomorphic derivation tree for the object grammar of a transduction. In fact, this redefinition has allowed for the first implementation of synchronous TAG processing, due to Onnig Dombalagian. This implementation was based on the inference-based TAG parser that we have presented elsewhere (Schabes and Shieber, 1994).

5. Problems with the Natural Definition

Along with the advantages of the new definition of synchronous TAG derivation, new problems are introduced as well. First, the exclusion of multiple adjunctions at a single address is problematic for synchronous TAG derivations. Second, the isomorphism requirement between the derivation trees may be too strong as well. The former problem admits of a straightforward solution, which we describe below. The latter does not; we describe the symptoms of the problem but leave its resolution as an open issue for further research.

5.1. **Multiple Adjunction.** Consider the synchronous TAG analysis of the semantics of adverbs given in Figure 1. This grammar is intended to allow for the ambiguity of strings such as 'John intentionally blinked twice' as shown in Figure 3. As previously mentioned, the ambiguity arises from the ability to perform two rewriting steps at the same node, the root F node in the elementary tree $\alpha(blink)_R$ corresponding to the word 'blinked'. Under the natural definition, however, this would entail a derivation tree pair of the geometry given in Figure 8. But the right derivation tree is ill-formed, as it violates the prohibition against multiple adjunctions at a single address.

It was the desire to model semantic ambiguity through violations of the prohibition that led us originally to a rewriting — as opposed to a derivation tree — approach to defining synchronous TAG derivation. Thus, the deviation from the natural definition of synchronous derivation was necessary because we required the ability of two elementary trees to be adjoined at the same node. Unfortunately, the rewriting interpretation of TAGs is a very inelegant way in which to get this ability, leading as it does to the problems described in Section 2.4. Nonetheless, without this ability, the utility of synchronous TAGs is severely diminished.

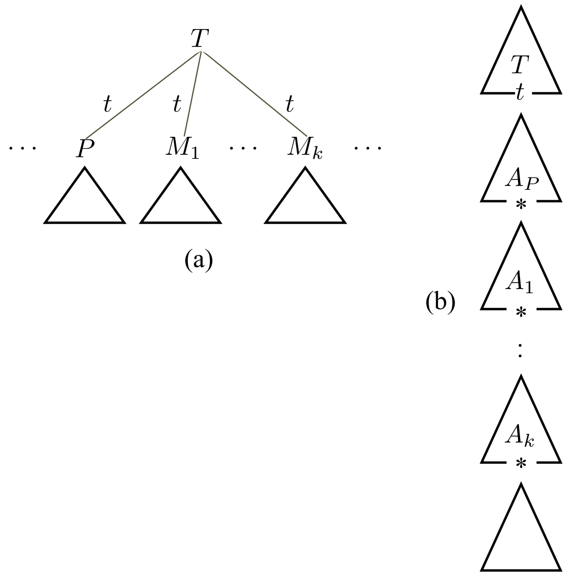


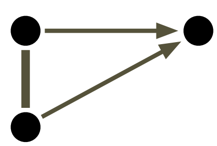
FIGURE 9. Interpretation of derivations with multiple adjunctions at a single node. In this case, several modifier trees M_1 through M_k have been adjoined at node t in tree T, along with a single predicative node P. The derived trees associated with P and M_1 through M_k, namely A_P and A_1 through A_k appear in the derived tree in that order.

For quite separate reasons, Schabes and I have been examining alternatives to Vijay-Shanker's definition of TAG derivation so as to allow for multiple adjunctions of certain auxiliary trees at the same node. Our solution (Schabes and Shieber, 1994) divides the class of auxiliary trees into two types, *modifier* trees and *predicative* trees, of which only the former allow such multiple adjunctions. In Vijay-Shanker's definition of derivation, a derivation tree is well-formed if no two auxiliary trees are adjoined at the same node in the same tree. In our revised definition, a derivation tree is well-formed if no two *predicative* auxiliary trees are adjoined at the same node in the same tree. Furthermore, so as to determinately specify a derived tree, all modifier trees that are adjoined at the same node in the same tree are ordered with respect to one another. Figure 9 shows the interpretation, in terms of derived tree (9b), of a derivation tree (9a) with multiple adjunctions at a single node. In essence, this diagram gives the interpretation of the operation $I[A_1/t_1, \ldots, A_k/t_k]$ when the t_k are not disjoint.

The existence of the revised definition of derivation vitiates the argument for the flat definition of synchronous TAG derivation. Rather, a direct definition is now possible along the previous lines. The only difference is that D_L and D_R are taken to be well-formed derivation trees of the new variety. Taking the trees $\beta(twice)_R$ and $\beta(intentionally)_R$ to be modifier trees, the synchronous derivation in Figure 8 is well-formed. The two possible orderings of the child nodes adjoining at address ϵ provide for the two readings of the ambiguous sentence.

5.2. **The Isomorphism Requirement.** A potentially more severe (and certainly more subtle) problem results from the requirement of isomorphism between D_L and D_R. There seem to be certain applications of synchronous TAGs for which this requirement is too strong. In this section, we present a taxonomy of potential counterexamples to isomorphism, organized by the "shape" of the nonisomorphic part of the mapping between the derivation trees. The examples are drawn from both technological application of synchronous TAG to the problem of defining translations between languages and application of synchronous TAG to the modeling of natural language semantics. It may turn out that different applications provide different amounts of pressure to loosen the isomorphism requirement in differing ways. Although we discuss several possible approaches to resolving this issue, we leave to further work whether a satisfactory solution for a given application can be found, and if so, what that solution might be.

Many-to-One Mappings.



The simplest examples are cases in which an atomic construction in one language is compound in another. For example, Abeillé et al. (1990) point out that the English adverbial 'hopefully' is translated by the French phrase 'on espère que'. Whereas the English corresponds to a single elementary tree, the French corresponds to a tree derived by substituting the elementary tree for 'on' as the NP argument of 'espère'. Such examples argue for the ability to allow the mapping between the left and right derivation trees to be relaxed from a strict isomorphism.

One might think (as indeed the present author did before penetrating discussions with Anthony Kroch) that a mismatch such as this shows that the isomorphism requirement must also be too strong for the purpose of modeling natural language semantics, for if these two constructions — 'hopefully' and 'on espère que' — have the same semantics, then at least one of the two (if not both) must exhibit a mismatch between the natural language derivation and a derivation of its logical form. The error in this reasoning follows from the assumption that the relationship of "corresponds as an appropriate translation" (in the sense in which bilingual dictionaries record such facts) is tantamount to "means the same as". This assumption is highly suspect. Bilingual dictionaries do not codify perfect translations in any sense, if such a notion is even coherent.

However, mismatches of this variety may also be found in applications to directly modeling natural-language semantics. For instance, the transduction relationship

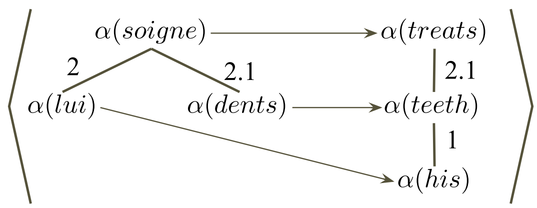
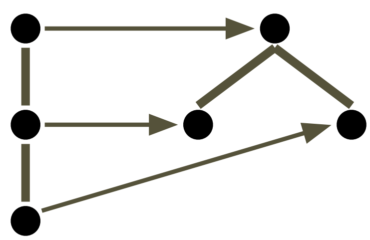


Figure 10. Schematic derivation tree pair for example (2). The arrows show the required mapping between the derivations, which is not an isomorphism.

between a compound idiom (such as 'kick the bucket') and its atomic semantics (given, e.g., by a simple predication of *die*) might be thought to be of this form.

Elimination of Dominance.



Even when the number of nodes in the paired derivation trees is the same, they may exhibit different structure. Nodes participating in a domination relationship in one tree may be mapped to nodes neither of which dominates the other.

Abeillé (personal communication) has noted a potential example of such a mismatch. For instance, in the sentence

(1) Le docteur soigne les dents de Jean.
The doctor treats Jean's teeth.

the subphrase 'de Jean' is substituted into the 'dents' tree syntactically, and arguably modifies the semantics of that tree as well. However, the cliticized version of the sentence

(2) Le docteur lui soigne les dents.
The doctor treats his teeth.

involves syntactic adjunction of the clitic 'lui' in the tree for 'soigne', although the translation into English, as before, places the pronoun within the object NP of the verb. Schematically, the derivation trees should show the geometry given in Figure 10. Note that the separate derivations are not isomorphic; a sibling relation in one tree corresponds to a domination relation in the other.

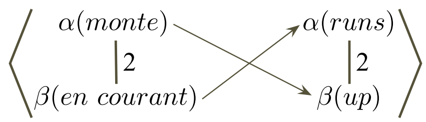
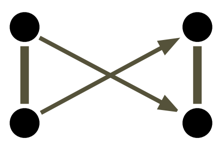


FIGURE 11. Schematic derivation tree pair for example (3). The arrows show the required mapping between the derivations, which is not an isomorphism.

Again, examples may be found in the arena of semantic interpretation. Although the argumentation is much more complex, and well beyond the scope of this paper, similar relationships arise in the context of modeling quantifier scope ambiguity.

Inversion of Dominance.



An even more extreme relationship, in which domination relationships are not only introduced but actually inverted, is exemplified by the French sentence and its English translation given in (3), and discussed by Whitelock (1992).

(3) Jean monte la rue en courant.
John runs up the street.

In this example, the phrase ‘en courant’ adjoins as an adverbial modifier to the verb ‘monte’. Presumably, ‘en courant’ would be paired with the English ‘runs’ and ‘monte’ with the English ‘up’. But the derivation tree for the English sentence would not then have the isomorphic structure in which ‘runs’ adjoins or substitutes into ‘up’, at least under the most natural analysis. Rather, the converse should hold; ‘up’ should be inserted into ‘runs’. Figure 11 shows the derivation tree pair schematically, including the nonisomorphism mapping between the trees.

We know of no example of inversion of dominance in applications to natural-language semantics.

5.3. **Relaxing Isomorphism.** In many of the above examples, although the mapping among derivation nodes is not an isomorphism, the deviation from isomorphism is nicely bounded, so that they could be well handled by allowing bounded subderivations to be considered elementary for the purpose of defining the relationship between the trees. In using synchronous TAGs as a model for language translation, that is, essentially to specify a bilingual lexicon, it is not surprising that bounded subderivations in one language are paired as a whole with bounded subderivations in another. Indeed, this is the modus operandi for traditional bilingual dictionaries. The Harper/Collins/Robert English-French dictionary provides an entry for ‘to [run] down/in/off’ with translation ‘descendre/entrer/partir en courant’ essentially providing the mapping between the pertinent subderivations. Similarly, the pertinent entry under ‘hopefully’ specifies the translation of ‘[hopefully] it won’t rain’ as ‘on espère qu’il ne va pas pleuvoir’, providing implicitly the subderivation mapping of 'hopefully' in its presentential position with 'on espère que'. For the most part, placing the isomorphism at the level of certain primitive and bounded subderivations is plausible, sufficiently expressive,[3] and retains the advantages described in Section 4.

If further relaxation of the isomorphism requirement is to be allowed, some method of controlling the relationship between the pair derivations will be needed. Owen Rambow and Giorgio Satta (personal communication) have conjectured that an approach along the lines of control grammars might be useful. This possibility, though tantalizing, remains to be explored.

Whitelock's method of "shake-and-bake" translation (Whitelock, 1992), under which translation involves reusing the same components but under different relationships, seems to correspond to a version of synchronous TAGs in which there is no constraint on the geometries of the derivation trees, the only requirement being that they are constructed from paired elements. This extreme version of relaxing the isomorphism requirement may in the end be necessary.

The exact nature of the relationship between paired derivation trees must remain for future work.

6. Conclusion

We have introduced a simple, natural definition of synchronous tree-adjoining derivation, based on isomorphisms between standard tree-adjoining derivations, that avoids the expressivity and implementability problems of the original rewriting definition. The decrease in expressivity, which would otherwise make the method unusable, is offset by the incorporation of an alternative definition of standard tree-adjoining derivation, previously proposed for completely separate reasons, that allows for multiple adjunctions at a single node in an elementary tree. The increased flexibility from the ability to perform such multiple adjunctions makes it conceivable to entertain using the natural definition of synchronous derivation. Nonetheless, some remaining problematic cases call for yet more flexibility in the definition; the isomorphism requirement may have to be relaxed. It remains for future research to tune the exact requirements on the allowable mappings.

Acknowledgements. The research described in this paper was made possible in part by a Presidential Young Investigator grant IRI-91-57996 from the National Science Foundation and matching funds from Xerox Corporation. An early version of this paper was presented at the Second Workshop on Tree-Adjoining Grammars, in Philadelphia, PA, in August 1992. The author is indebted to the following people for helpful discussions on the subject matter of this paper: Anne Abeillé, Aravind Joshi, Owen Rambow, Giorgio Satta, Yves Schabes, K. Vijay-Shanker, David Weir, and Peter Whitelock.

References

Abeillé, Anne, Yves Schabes, and Aravind K. Joshi. 1990. Using lexicalized tree adjoining grammars for machine translation. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING '90)*. Helsinki.

[3]The French clitic example, however, remains problematic. The relation between the clitic and the NP which it is semantically related to seems to be potentially unbounded.

Schabes, Yves, and Stuart M. Shieber. 1994. An alternative conception of tree-adjoining derivation. *Computational Linguistics* 20(1):91–124. Also available as cmp-lg/9404001.

Shieber, Stuart M., and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*. Helsinki.

Vijay-Shanker, K. 1987. A study of tree adjoining grammars. Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania.

Whitelock, Peter. 1992. Shake-and-bake translation. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING '92)*. Nantes, France.

Division of Applied Sciences, Harvard University, 33 Oxford Street, Cambridge, MA 02138, shieber@das.harvard.edu