Non-isomorphic Mappings in Synchronous Derivations

Anoop Sarkar

University of Pennsylvania
Department of Computer and Information Science, Philadelphia PA 19104
anoop@linc.cis.upenn.edu

1 Introduction

The formalism of synchronous tree adjoining grammars was introduced as a means for defining mappings between pairs of standard tree adjoining grammars (TAGs). Each mapping defines a (reversible) translation from the strings generated by one TAG in the pair to the other[1]. This mapping was first defined (in [2]) as an iterative rewriting process over derived trees by using links between nodes of the derived trees of a pair of TAGs. However, this definition of synchronous TAGs was modified in [1] and [3] for different reasons. [3] showed that the original definition had two major shortcomings: the rewriting definition could generate strings not in the set of tree adjoining languages (TALs) and the notion of rewriting derived trees was not well-defined in terms of derivation trees. To solve these problems with synchronous TAGs [3] proposes a definition in terms of pairs of isomorphic derivation trees for a pair of TAGs. [1] also redefines synchronous TAGs (for implementational reasons) as a mapping of a derivation tree from a source TAG to a derivation tree in the target TAG. Notably, the approach in [1] considers mappings between derivations that are not isomorphic. This paper looks at the formal nature of non-isomorphic mappings between TAGs.

First, we establish some notations and definitions used. Given a TAG, we can represent derivations by *derivation trees* (Definition 1). The adjunction of several trees $\gamma_1, \ldots, \gamma_k$ into a tree γ at distinct addresses $i_1, \ldots, i_k$, respectively is denoted by $\gamma[i_1, \gamma_1] \ldots [i_k, \gamma_k]$

Definition 1 (Derivation trees). Derivation trees have the following inductive definition:

- If $\gamma$is an elementary tree with no OA constraints the derivation tree consists of a single node labelled $\overline{\gamma}$.
- If Υ_j is a derivation tree with root labelled $\overline{\beta_j}$ for $1 \leq j \leq k$ then given the derivation

$$\gamma' = \gamma[i_1, \beta_1] \ldots [i_k, \beta_k]$$

then following (Fig 1) is also a derivation tree

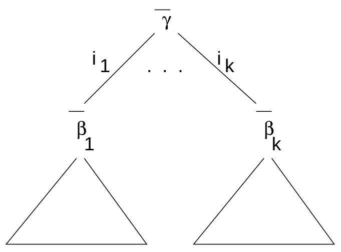

Fig. 1. Definition of derivation tree.

For any TAG we can give a CFG whose tree set is exactly the set of derivation trees for the TAG. [5] defines Generalized Context-Free Grammars (GCFGs) which generates a set of trees that can be interpreted as derivation trees in a TAG.

Definition 2. A **GCFG** is written as $G = (V, S, F, P)$, where

- V is a set of variables
- S is a distinguished member of V
- F is a finite set of function symbols
- P is a finite set of productions of the form

$$A \rightarrow f(A_1, \ldots, A_n)$$

where $n \geq 0, f \in F$, and $A, A_1, \ldots, A_n \in V$

The set of terms, $T(G)$ derived from a GCFG, G is the set of all t such that $S \overset{*}{\underset{G}{\Rightarrow}} t$ where the *derives* relation is defined as follows.

- $A \underset{G}{\Rightarrow} f()$ if $A \rightarrow f()$ is a production.
- $A \overset{*}{\underset{G}{\Rightarrow}} f(t_1, \ldots, t_n)$ if $A \rightarrow f(A_1, \ldots, A_n)$ is a production, and $A_i \overset{*}{\underset{G}{\Rightarrow}} t_i$ for $1 \leq i \leq n$.

For each zero arity function $f \in F$, we let $[\![f()]\!]_{TAG} = \gamma$, where γ is a complete elementary tree (i.e. with no OA nodes). For each function $f \in F$ of n arguments, where $n \geq 1$, we let

$$[\![f(t_1, \ldots, t_n)]\!]_{TAG} = \gamma[a_1, [\![t_1]\!]_{TAG}] \ldots [a_n, [\![t_n]\!]_{TAG}]$$

$\gamma$is a tree in which trees $[\![t_1]\!]_{TAG}, \ldots, [\![t_n]\!]_{TAG}$ are adjoined at addresses $a_1, \ldots, a_n$ to produce $[\![f(t_1, \ldots, t_n)]\!]_{TAG}$. Note that for any GCFG production,

[1] A familiarity with TAGs and synchronous TAGs as defined in, for instance, [4] and [2] is assumed.

say $A \rightarrow f(A_1, A_2)$ we can construct another production $A \rightarrow f'(A_2, A_1)$ without changing the interpretation of the GCFG. We shall use this fact heavily in later sections.

Definition 3 (Synchronous TAGs). A synchronous TAG G is defined as a set of triples $\{< L_i, R_i, \frown_i >\}$ where the L_i and R_i are elementary trees, both initial and auxiliary and $\frown_i$ is the set of links between tree addresses in L_i and R_i.

Thus, $G_L = \{L_i\}$ and $G_R = \{R_i\}$ form the two component TAGs, and in general for a tuple $< x_L, x_R, \ldots >$, x_L and x_R are taken to be its first and second components. Also, the tree associated with a node α in a derivation tree is referred to as $\overline{\alpha}$ and the parent of a node α in a tree is denoted as *parent*(α).

2 Isomorphic Synchronous TAG Derivations

For a synchronous TAG G a derivation is stated as a pair of derivation trees $D_S =< D_L, D_R >$. Parsing in a synchronous TAG is typically defined as

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

where $< w_L, w_R > \in L(G)$ and using TAG G_L to parse w_L. Using the set of links $\frown$ we obtain D_S and its right projection D_R yields the string w_R. However, as shown in [3] the mapping from D_L to D_S is not well defined unless the following conditions hold.

1. D_L and D_R are well formed derivation trees relative to G_L and G_R respectively.
2. D_L and D_R are isomorphic with respect to dominance relations in the trees, i.e. if f is such an isomorphic map then if $f(\eta_l) = \eta_r$ then $f(\mathit{parent}(\eta_l)) = \mathit{parent}(\eta_r)$.
3. They are also isomorphic with respect to the linking between tree pairs. So, if $f(\eta_l) = \eta_r$, then $< \overline{\eta_l}, \overline{\eta_r}, \frown > \in G$ *and* $< \overline{\mathit{parent}(\eta_l)}, \overline{\mathit{parent}(\eta_r)}, \frown' > \in G$ and $\langle a, b \rangle \in \frown'$ (the addresses where η_l and η_r adjoined into their parents).

We redefine this notion of a synchronous TAG derivation as a mapping from terms in a GCFG to derivation trees.

Definition 4 (Isomorphic Synchronous TAGs). For a synchronous TAG $G =< G_L, G_R, \frown >$ there is a GCFG $G_L{}'$ which defines the set of the derivation trees of G_L (from Definition 2). Then we define $[\![]\!]_{STAG}$ as follows,

1. $[\![f()]\!]_{STAG} = \gamma$ where $< [\![f()]\!]_{TAG}, \gamma, \frown > \in G$
2. $[\![f(t_1, \ldots, t_n)]\!]_{STAG} = \gamma[b_1, [\![t_1]\!]_{STAG}] \ldots [b_n, [\![t_n]\!]_{STAG}]$, where
   - $< \varphi, \gamma, \frown > \in G$
   - $\{a_1 \frown b_1, \ldots, a_n \frown b_n\} \subseteq \frown$ and
   - $[\![f(t_1, \ldots, t_n)]\!]_{TAG} = \varphi[a_1, [\![t_1]\!]_{TAG}] \ldots [a_n, [\![t_n]\!]_{TAG}]$

3. Let $\gamma[a, \phi] = \gamma$ and let $[\![f()]\!]_{STAG} = \phi$ whenever $< [\![f()]\!]_{TAG}, \gamma, \frown > \notin G$

Theorem 5. *Synchronous TAGs defined as a pair of isomorphic derivations (Definition 4) generate exactly the set of tree-adjoining languages.*

Proof. The definition given (Definition 4) always derives a derivation tree which satisfies the isomorphism requirement with respect to dominance relations and linking between nodes. Since the definition given is equivalent to that in [3] the proof in [3][page 8] suffices.

3 Non-isomorphic Mappings

There are several examples from the application of synchronous TAGs to areas such as the definition of translations between natural languages and the modelling of natural language semantics for which the isomorphism requirement is found to be too strong a restriction. The following examples are taken from a synchronous TAG mapping between English and French from the implementation by [1]. For more comprehensive arguments about the isomorphism requirement the reader is referred to [3] and [2].

As [2] points out, the English adverbial "hopefully" corresponds to the French phrase "on espére que". Fig 2 shows the derivation trees and their mapping (which is non-isomorphic). Such a case also occurs in building semantic structures for idioms where a complex syntactic phrase such as "kick the bucket" could be mapped to an atomic formula ("die") in the semantic representation.

(1) On espére qu'il ne va pas pleuvoir.
Hopefully it won't rain.

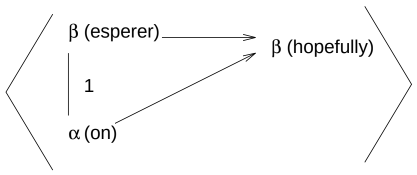

Fig. 2. Schematic derivation tree pair for example (1). The arrows show the required mapping between the derivations, which is not an isomorphism.

In (2) "de Jean" is substituted into the "dents" tree, whereas in (3) "lui" cliticizes in the tree for "soigne" while in the English sentence both the proper name and the pronoun are part of the object NP of the verb. Again the derivations are not isomorphic as shown by Fig 3.

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

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

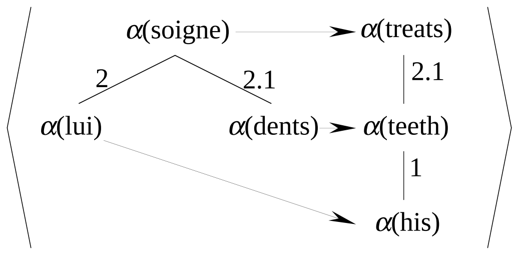

Fig. 3. Schematic derivation tree pair for example (3).

In (4) "en courant" adjoins as an adverbial modifer to the verb "monte". Under the natural mapping, "en courant" would be mapped with "runs" and "monte" with "up". However such a mapping causes an inversion in the dominance of the node in the derivation tree (see Fig 4).

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

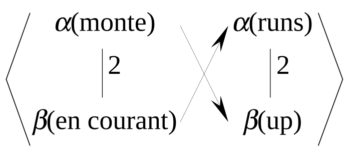

Fig. 4. Schematic derivation tree pair for example (4).

3.1 Bounded Non-isomorphic Mappings

In the above examples the non-isomorphic maps are bounded, and hence can be handled by allowing bounded subtrees of the derivation tree (or subderivations) to be treated as elementary constituents and stored in the lexicon as a unit. This is the alternative taken in [1] and also mentioned as a solution in [3]. A subderivation is defined as an m-incomplete derivation tree (see Definition 6).

Definition 6 (m-incomplete derivation tree, [4]). A m-incomplete derivation tree is defined by the following conditions:

1. Any derivation tree is a 0-incomplete derivation tree.
2. Let γ an elementary tree. Let P be the set of addresses of nodes in γ with OA constraints, and let Q be the set of addresses of nodes in γ not having OA constraints. Let $\{i_1, \ldots, i_k : k \geq 0\} \subseteq P$, $\{n_1, \ldots, n_q : q \geq 0\}$ be the rest of the addresses in P and $\{j_1, \ldots, j_l : l \geq 0\} \subseteq Q$. Let the auxiliary tree β_{j_p} be adjoinable at the node addressed j_p in γ and Γ_{j_p} be a m_{j_p}-incomplete derivation tree with root symbol $\overline{\beta_{j_p}}$ for $1 \leq p \leq l$. Similarly, let β_{n_p} be adjoinable at n_p in γ and Γ_{n_p} be a m_{n_p}-incomplete derivation tree with root symbol $\overline{\beta_{n_p}}$ for $1 \leq p \leq q$. Then the derivation tree in Fig. 5 is an $m = (k + m_{j_1} + \cdots + m_{j_l} + m_{n_1} + \cdots + m_{n_q})$-incomplete derivation tree.

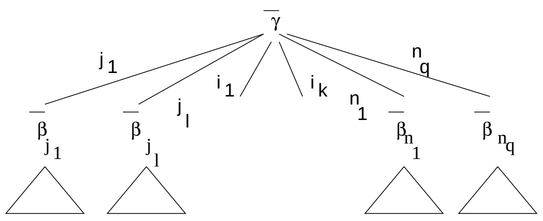

Fig. 5. m-incomplete derivation tree.

Theorem 7. *For every γ derived with k ($k \geq 0$) adjunctions such that there are m nodes in γ with* OA *constraints, there is a m-incomplete derivation tree Γ which describes γ.*

Proof. In [4][page 25].

A synchronous TAG G is now defined as a set of triples $\{< L_i, R_i, \frown_i >\}$ where the L_i and R_i are either elementary trees as before, or m-incomplete derivation trees. We add the following definition to the earlier definition (Definition 4) to allow the mapping of bounded (potentially) non-isomorphic derivation trees. We denote subderivations with the letters Γ, Δ, etc.

Definition 8 (Bounded Non-isomorphic Synchronous Derivations). The following is added to Definition 4:

4. $[\![f(t_1, \ldots, t_m, t_{m+1}, \ldots, t_n]\!]_{STAG} =$

   $\Delta[b_{m+1}, [\![t_{m+1}]\!]_{STAG}], \ldots, [b_n, [\![t_n]\!]_{STAG}], [c_p, [\![t_p]\!]_{STAG}], \ldots, [c_k, [\![t_k]\!]_{STAG}]$

   where,

- $A_i \overset{\Rightarrow}{_G} f_i(t_{i_1}, \ldots, t_{i_n}), 1 \leq i \leq m$ and $m \geq 0$
- $[\![f(f_1(), \ldots, f_m())]\!]_{TAG} = \Gamma$
- $< \Gamma, \Delta, \frown > \in G$, and
- for each i, $1 \leq i \leq m, [\![f_i(t_{i_1}, \ldots, t_{i_n})]\!]_{TAG} = \gamma[c_{i_1}, [\![t_{i_1}]\!]_{TAG}] \ldots [c_{i_n}, [\![t_{i_n}]\!]_{TAG}]$ and let $\{c_p, \ldots, c_k\} = \{c_{i_1}, \ldots, c_{i_n}\} \cap \frown$ and let $t_p \ldots t_k$ be the associated terms.

Note that the above definition is only for subderivations of height at most one. For a more general version of the above definition all we have to do is substitute for $A_i \overset{\Rightarrow}{_G} f_i(\ldots)$ the statement $A_i \overset{*}{\underset{G}{\Rightarrow}} f_{i_1}(\ldots(f_{i_n}(\ldots))\ldots)$ and then check all subsets of the set of leftmost (or rightmost) derivations to check for Γ as above.

3.2 Unbounded Non-isomorphic Mappings

Although most of the examples discussed earlier seem to require only cases of bounded non-isomorphic mappings, there are cases when an unbounded distance can occur between subderivations to be transferred. For example:

(5) Le bord de la couverture de ce livre est déchirée.
Le bord de la couverture en est déchirée.
The corner of the cover of the book is torn out.

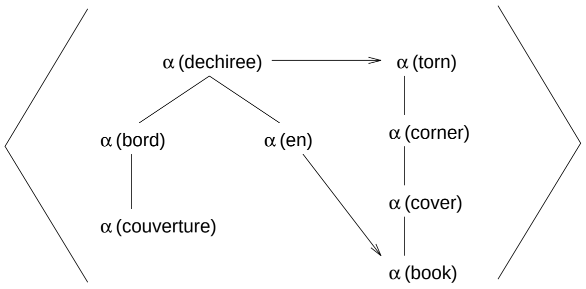

Fig. 6. Schematic derivation tree pair for example (5).

For (5) Fig 6 shows an instance of an unbounded non-isomorphic map between the derivation trees for French and English. To handle such cases we now define an unbounded mapping in a synchronous TAG which is nevertheless restricted enough to preserve the formal properties defined in previous sections.

Unbounded subderivations are stored in the lexicon as a pair $< P_L, P_R >$ where both P_L and P_R are a finite set of GCFG productions. Let $\Phi =$

$\{A_1, \ldots, A_n\}$ be the set of lhs's of the productions in P_L. Then P_L is defined as the finite set of productions of the form:

- $A_i \rightarrow f_i(\alpha_i)$, or
- $A_i \rightarrow f_i(A_p, \ldots, A_q, \alpha_i)$, where $1 \leq i \leq n$, and $\{A_p, \ldots, A_q\} \subseteq \Phi$ and each α_i is any sequence of labels, possibly empty.

P_R is then defined as:

- $A_i \rightarrow f_i(X, \alpha_i)$
- where, X is a sequence of labels, say $A_p, \ldots, A_q$ such that $\{A_p, \ldots, A_q\} \subseteq \Phi \cup \Phi'$ where Φ' is the finite set of lhs's of new productions added to P_R.

Such a definition allows the introduction of a bounded number of productions into P_R and also allows the removal of productions. Now, given D_L to obtain $D_S =< D_L, D_R >$ take the GCFG G that corresponds to D_L and for each pair $< P_L, P_R >$ find productions of P_L in P_{D_L} (renaming labels as necessary) and replace them with productions in P_R. $[\![]\!]_{STAG}$ can now be applied as before to obtain D_R.Note that this definition doesn't make clear that P_R will give us a well-formed TAG derivation tree. This requires a proof or further stipulation.

Example 1. For the unbounded example in (5) define $P_L = \{S \rightarrow f(A_2, \alpha), A_2 \rightarrow f_2(\beta), A_3 \rightarrow f_3(\delta)\}$ and $P_R = \{S \rightarrow f(\alpha), A_2 \rightarrow f_2(\beta), A_3 \rightarrow f_3(A_2, \delta)\}$. Then for the following set of GCFG productions for the French example in Fig 6

- α(déchirée) : $S \rightarrow f(A_1, A_2)$
- α(bord) : $A_1 \rightarrow f_1(A_3)$
- α(couverture) : $A_3 \rightarrow f_3()$
- α(en) : $A_2 \rightarrow f_2()$

conversion of the productions above that are matched by P_L to the productions in P_R gives us the English derivation tree in Fig 6.

References

1. Gilles Prigent. Overview of the TAG transfer module. ms. University of Pennsylvania, 1992.
2. Stuart Sheiber and Yves Schabes. Synchronous tree adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING '90)*, Helsinki, August 1990. Association for Computational Linguistics.
3. Stuart Shieber. Restricting the weak generative capacity of Synchronous Tree Adjoining Grammars. *Computational Intelligence*, 10(4):371–385, 1994.
4. K. Vijay-Shanker. *A Study of Tree Adjoining Grammars.* PhD thesis, Department of Computer and Information Science, University of Pennsylvania, 1987.
5. David J. Weir. *Characterizing Mildly Context-Sensitive Grammar Formalisms.* PhD thesis, Dept. of Computer and Info. Sc., University of Pennsylvania, Philadelphia, PA, 1988.

This article was processed using the LaTeX macro package with LLNCS style