A Decoder for Probabilistic Synchronous Tree Insertion Grammars

Steve DeNeefe * **and Kevin Knight** *
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292, USA
{sdeneefe,knight}@isi.edu

Heiko Vogler †
Department of Computer Science
Technische Universität Dresden
D-01062 Dresden
Heiko.Vogler@tu-dresden.de

Abstract

Synchronous tree insertion grammars (STIG) are formal models for syntax-based machine translation. We formalize a decoder for probabilistic STIG; the decoder transforms every source-language string into a target-language tree and calculates the probability of this transformation.

1 Introduction

Tree adjoining grammars (TAG) were invented in (Joshi et al. 1975) in order to better characterize the string sets of natural languages[1]. One of TAG's important features is the ability to introduce two related syntactic units in a single rule, then push those two units arbitrarily far apart in subsequent derivation steps. For machine translation (MT) between two natural languages, each being generated by a TAG, the derivations of the two TAG may be synchronized (Abeille et al., 1990; Shieber and Shabes, 1990) in the spirit of syntax-directed transductions (Lewis and Stearns, 1968); this results in *synchronous TAG* (STAG). Recently, in (Nesson et al., 2005, 2006) probabilistic synchronous tree insertion grammars (pSTIG) were discussed as model of MT; a tree insertion grammar is a particular TAG in which the parsing problem is solvable in cubic-time (Schabes and Waters, 1994). In (DeNeefe, 2009; DeNeefe and Knight 2009) a decoder for pSTIG has been proposed which transforms source-language strings into (modifications of) derivation trees of the pSTIG. Nowadays, large-scale linguistic STAG rule bases are available.

In an independent tradition, the automata-theoretic investigation of the translation of trees led to the rich theory of tree transducers (Gécseg and Steinby, 1984, 1997). Roughly speaking, a tree transducer is a finite term rewriting system. If each rewrite rule carries a probablity or, in general, a weight from some semiring, then they are weighted tree transducers (Maletti, 2006, 2006a; Fülöp and Vogler, 2009). Such weighted tree transducers have also been used for the specification of MT of natural languages (Yamada and Knight, 2001; Knight and Graehl, 2005; Graehl et al., 2008; Knight and May 2009).

Martin and Vere (1970) and Schreiber (1975) established the first connections between the two traditions; also Shieber (2004, 2006) and Maletti (2008, 2010) investigated their relationship.

The problem addressed in this paper is the decoding of source-language strings into target-language trees where the transformation is described by a pSTIG. Currently, this decoding requires two steps: first, every source string is translated into a derivation tree of the underlying pSTIG (DeNeefe, 2009; DeNeefe and Knight 2009), and second, the derivation tree is transformed into the target tree using an embedded tree transducer (Shieber, 2006). We propose a transducer model, called a *bottom-up tree adjoining transducer*, which performs this decoding in a single step and, simultaneously, computes the probabilities of its derivations. As a basis of our approach, we present a formal definition of pSTIG.

2 Preliminaries

For two sets Σ and A, we let $U_\Sigma(A)$ be the set of all (unranked) trees over Σ in which also elements of A may label leaves. We abbreviate $U_\Sigma(\emptyset)$ by U_Σ. We denote the set of *positions*, *leaves*, and *non-leaves* of $\xi \in U_\Sigma$ by $\mathrm{pos}(\xi) \subseteq \mathbb{N}^*$, $\mathrm{lv}(\xi)$, and $\mathrm{nlv}(\xi)$, resp., where ε denotes the root of ξ and $w.i$ denotes the ith child of position w; $\mathrm{nlv}(\xi) = \mathrm{pos}(\xi) \setminus \mathrm{lv}(\xi)$. For a position $w \in \mathrm{pos}(\xi)$, the *label of ξ at w* (resp., *subtree of ξ at w*) is denoted

* financially supported by NSF STAGES project, grant #IIS-0908532.

† financially supported by DFG VO 1011/5-1.

[1] see (Joshi and Shabes, 1997) for a survey

Proceedings of the 2010 Workshop on Applications of Tree Automata in Natural Language Processing, ACL 2010, pages 10–18,
Uppsala, Sweden, 16 July 2010.

by $\xi(w)$ (resp., $\xi|_w$). If additionally $\zeta \in U_\Sigma(A)$, then $\xi[\zeta]_w$ denotes the tree which is obtained from ξ by replacing its subtree at w by ζ. For every $\Delta \subseteq \Sigma \cup A$, the set $\mathrm{pos}_\Delta(\xi)$ is the set of all those positions $w \in \mathrm{pos}(\xi)$ such that $\xi(w) \in \Delta$. Similarly, we can define $\mathrm{lv}_\Delta(\xi)$ and $\mathrm{nlv}_\Delta(\xi)$. The *yield of* ξ is the sequence $\mathrm{yield}(\xi) \in (\Sigma \cup A)^*$ of symbols that label the leaves from left to right.

If we associate with $\sigma \in \Sigma$ a rank $k \in \mathbb{N}$, then we require that in every tree $\xi \in U_\Sigma(A)$ every σ-labeled position has exactly k children.

3 Probabilistic STAG and STIG

First we will define probabilistic STAG, and second, as a special case, probabilistic STIG.

Let N and T be two disjoint sets of, resp., nonterminals and terminals. A *substitution rule* r is a tuple $(\zeta_s, \zeta_t, V, W, P^r_{\mathrm{adj}})$ where

- $\zeta_s, \zeta_t \in U_N(T)$ (*source* and *target tree*) and $|\mathrm{lv}_N(\zeta_s)| = |\mathrm{lv}_N(\zeta_t)|$,
- $V \subseteq \mathrm{lv}_N(\zeta_s) \times \mathrm{lv}_N(\zeta_t)$ (*substitution sites*), V is a one-to-one relation, and $|V| = |\mathrm{lv}_N(\zeta_s)|$,
- $W \subseteq \mathrm{nlv}_N(\zeta_s) \times \mathrm{nlv}_N(\zeta_t)$ (*potential adjoining sites)*, and
- $P^r_{\mathrm{adj}} : W \to [0, 1]$ (*adjoining probability*).

An *auxiliary rule* r is a tuple $(\zeta_s, \zeta_t, V, W, *, P^r_{\mathrm{adj}})$ where ζ_s, ζ_t, W, and P^r_{adj} are defined as above and

- V is defined as above except that $|V| = |\mathrm{lv}_N(\zeta_s)| - 1$ and
- $* = (*_s, *_t) \in \mathrm{lv}_N(\zeta_s) \times \mathrm{lv}_N(\zeta_t)$ and neither $*_s$ nor $*_t$ occurs in any element of V; moreover, $\zeta_s(\varepsilon) = \zeta_s(*_s)$ and $\zeta_t(\varepsilon) = \zeta_t(*_t)$, and $*_s \neq \varepsilon \neq *_t$; the node $*_s$ (and $*_t$) is called the *foot-node of* ζ_s (resp., ζ_t).

An *(elementary) rule* is either a substitution rule or an auxiliary rule. The *root-category* of a rule r is the tuple $(\zeta_s(\varepsilon), \zeta_t(\varepsilon))$, denoted by $\mathrm{rc}(r)$.

A *probabilistic synchronous tree adjoining grammar* (pSTAG) is a tuple $G = (N, T, (S_s, S_t), \mathcal{S}, \mathcal{A}, P)$ such that N and T are two disjoint sets (resp., of nonterminals and terminals), $(S_s, S_t) \in N \times N$ (*start nonterminal*), $\mathcal{S}$ and $\mathcal{A}$ are finite sets of, resp., substitution rules and auxiliary rules, and $P : \mathcal{S} \cup \mathcal{A} \to [0, 1]$ such that for every $(A, B) \in N \times N$,

$$\sum_{\substack{r \in \mathcal{S} \\ \mathrm{rc}(r)=(A,B)}} P(r) = 1 \ \text{ and } \sum_{\substack{r \in \mathcal{A} \\ \mathrm{rc}(r)=(A,B)}} P(r) = 1$$

assuming that in each case the number of summands is not zero. In the following, let G always denote an arbitrary pSTAG.

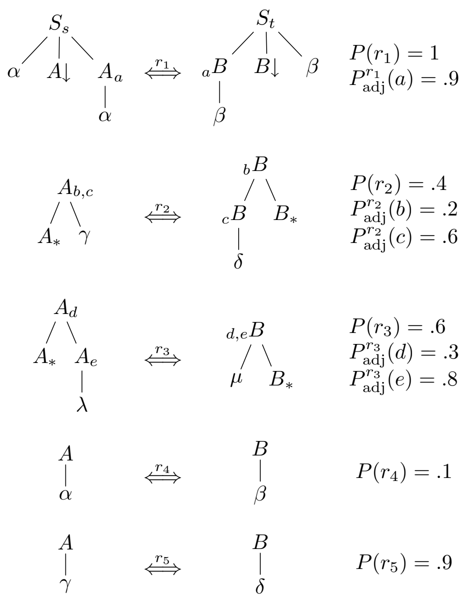

Figure 1: The running example pSTAG G.

In Fig. 1 we show the rules of our running example pSTAG, where the capital Roman letters are the nonterminals and the small Greek letters are the terminals. The substitution site (in rule r_1) is indicated by $\downarrow$, and the potential adjoining sites are denoted[2] by a, b, c, d, and e. For instance, in formal notation the rules r_1 and r_2 are written as follows:

$$r_1 = (S_s(\alpha, A, A(\alpha)),\ S_t(B(\beta), B, \beta),\ \{\downarrow\}, \{a\}, P^{r_1}_{\mathrm{adj}})$$

where $\downarrow = (2, 2)$ and $a = (3, 1)$, and

$$r_2 = (A(A, \gamma),\ B(B(\delta), B),\ \emptyset,\ \{b, c\}, *, P^{r_2}_{\mathrm{adj}})$$

where $b = (\varepsilon, \varepsilon)$, $c = (\varepsilon, 1)$, and $* = (1, 2)$.

In the derivation relation of G we will distinguish four types of steps:

1. substitution of a rule at a substitution site (substitution),
2. deciding to turn a potential adjoining site into an activated adjoining site (activation),
3. deciding to drop a potential adjoining site, i.e., not to adjoin, (non-adjoining) and
4. adjoining of a rule at an activated adjoining site (adjoining).

In the sentential forms (defined below) we will maintain for every adjoining site w a two-valued flag $g(w)$ indicating whether w is a potential ($g(w) = \mathrm{p}$) or an activated site ($g(w) = \mathrm{a}$).

The *set of sentential forms of* G is the set $\mathrm{SF}(G)$ of all tuples $\kappa = (\xi_s, \xi_t, V, W, g)$ with

[2] Their placement (as left or right index) does not play a role yet, but will later when we introduce pSTIG.

- $\xi_s, \xi_t \in U_N(T)$,
- $V \subseteq \mathrm{lv}_N(\xi_s) \times \mathrm{lv}_N(\xi_t)$ is a one-to-one relation, $|V| = |\mathrm{lv}_N(\xi_s)| = |\mathrm{lv}_N(\xi_t)|$,
- $W \subseteq \mathrm{nlv}_N(\xi_s) \times \mathrm{nlv}_N(\xi_t)$, and
- $g : W \to \{\mathrm{p}, \mathrm{a}\}$.

The *derivation relation (of G)* is the binary relation $\Rightarrow \;\subseteq\; \mathrm{SF}(G) \times \mathrm{SF}(G)$ such that for every $\kappa_1 = (\xi_s^1, \xi_t^1, V_1, W_1, g_1)$ and $\kappa_2 = (\xi_s^2, \xi_t^2, V_2, W_2, g_2)$ we have $\kappa_1 \Rightarrow \kappa_2$ iff one of the following is true:

1. (substitution) there are $w = (w_s, w_t) \in V_1$ and $r = (\zeta_s, \zeta_t, V, W, P_{\mathrm{adj}}^r) \in \mathcal{S}$ such that
   – $(\xi_s^1(w_s), \xi_t^1(w_t)) = \mathrm{rc}(r)$,
   – $\xi_s^2 = \xi_s^1[\zeta_s]_{w_s}$ and $\xi_t^2 = \xi_t^1[\zeta_t]_{w_t}$,
   – $V_2 = (V_1 \setminus \{w\}) \cup w.V$,[3]
   – $W_2 = W_1 \cup w.W$, and
   – g_2 is the union of g_1 and the set of pairs $(w.u, \mathrm{p})$ for every $u \in W$;
   this step is denoted by $\kappa_1 \overset{w,r}{\Longrightarrow} \kappa_2$;

2. (activation) there is a $w \in W_1$ with $g_1(w) = \mathrm{p}$ and $(\xi_s^1, \xi_t^1, V_1, W_1) = (\xi_s^2, \xi_t^2, V_2, W_2)$, and g_2 is the same as g_1 except that $g_2(w) = \mathrm{a}$; this step is denoted by $\kappa_1 \overset{w}{\Longrightarrow} \kappa_2$;

3. (non-adjoining) there is $w \in W_1$ with $g_1(w) = \mathrm{p}$ and $(\xi_s^1, \xi_t^1, V_1) = (\xi_s^2, \xi_t^2, V_2)$, $W_2 = W_1 \setminus \{w\}$, and g_2 is g_1 restricted to W_2; this step is denoted by $\kappa_1 \overset{\neg w}{\Longrightarrow} \kappa_2$;

4. (adjoining) there are $w \in W_1$ with $g_1(w) = \mathrm{a}$, and $r = (\zeta_s, \zeta_t, V, W, *, P_{\mathrm{adj}}^r) \in \mathcal{A}$ such that, for $w = (w_s, w_t)$,
   – $(\xi_s^1(w_s), \xi_t^1(w_t)) = \mathrm{rc}(r)$,
   – $\xi_s^2 = \xi_s^1[\zeta_s']_{w_s}$ where $\zeta_s' = \zeta_s[\xi_s^1|_{w_s}]_{*_s}$, $\xi_t^2 = \xi_t^1[\zeta_t']_{w_t}$ where $\zeta_t' = \zeta_t[\xi_t^1|_{w_t}]_{*_t}$,
   – V_2 is the smallest set such that (i) for every $(u_s, u_t) \in V_1$ we have $(u_s', u_t') \in V_2$ where
   $$u_s' = \begin{cases} u_s & \text{if } w_s \text{ is not a prefix of } u_s, \\ w_s.*_s.u & \text{if } u_s = w_s.u \text{ for some } u; \end{cases}$$
   and u_t' is obtained in the same way from u_t, w_t, and $*_t$, and (ii) V_2 contains $w.V$;
   – W_2 is the smallest set such that (i) for every $(u_s, u_t) \in W_1 \setminus \{w\}$ we have $(u_s', u_t') \in W_2$ where u_s' and u_t' are obtained in the same way as for V_2, and $g_2(u_s', u_t') = g_1(u_s, u_t)$ and (ii) W_2 contains $w.W$ and $g_2(w.u) = \mathrm{p}$ for every $u \in W$;
   this step is denoted by $\kappa_1 \overset{w,r}{\Longrightarrow} \kappa_2$.

[3] $w.V = \{(w_s.v_s, w_t.v_t) \mid (v_s, v_t) \in V\}$

In Fig. 2 we show a derivation of our running example pSTAG where activated adjoining sites are indicated by surrounding circles, the other adjoining sites are potential.

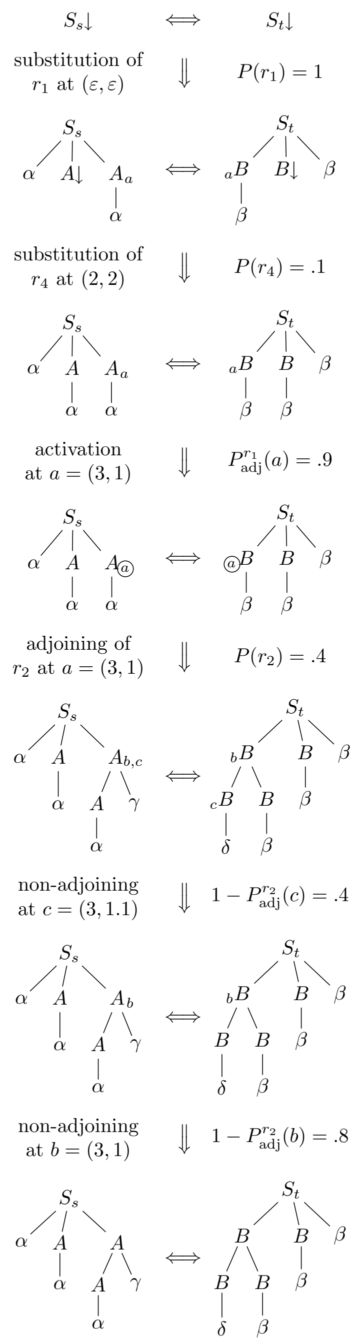

Figure 2: An example derivation with total probability $1 \times .1 \times .9 \times .4 \times .4 \times .8 = .01152$.

The only *initial* sentential form is $\kappa_{\mathrm{in}} = (S_s, S_t, \{(\varepsilon, \varepsilon)\}, \emptyset, \emptyset)$. A sentential form κ is *final* if it has the form $(\xi_s, \xi_t, \emptyset, \emptyset, \emptyset)$. Let $\kappa \in \mathrm{SF}(G)$. A *derivation (of κ)* is a sequence d of the form $\kappa_0 u_1 \kappa_1 \ldots u_n \kappa_n$ with $\kappa_0 = \kappa_{\mathrm{in}}$ and $n \geq 0$, $\kappa_{i-1} \overset{u_i}{\Rightarrow} \kappa_i$ for every $1 \leq i \leq n$ (and $\kappa_n = \kappa$). We denote κ_n also by $\mathrm{last}(d)$, and the set of all derivations of κ (resp., derivations) by $D(\kappa)$ (resp., D). We call $d \in D$ *successful* if $\mathrm{last}(d)$ is final.

The *tree transformation computed by* G is the relation $\tau_G \subseteq U_N(T) \times U_N(T)$ with $(\xi_s, \xi_t) \in \tau_G$ iff there is a successful derivation of $(\xi_s, \xi_t, \emptyset, \emptyset, \emptyset)$.

Our definition of the probability of a derivation is based on the following observation.[4] Let $d \in D(\kappa)$ for some $\kappa = (\xi_s, \xi_t, V, W, g)$. Then, for every $w \in W$, the rule which created w and the corresponding local position in that rule can be retrieved from d. Let us denote this rule by $r(d, \kappa, w)$ and the local position by $l(d, \kappa, w)$.

Now let d be the derivation $\kappa_0 u_1 \kappa_1 \ldots u_n \kappa_n$. Then the *probability of* d is defined by

$$P(d) = \prod_{1 \le i \le n} P_d(\kappa_{i-1} \overset{u_i}{\Rightarrow} \kappa_i)$$

where

1. (substitution) $P_d(\kappa_{i-1} \overset{w,r}{\Longrightarrow} \kappa_i) = P(r)$
2. (activation) $P_d(\kappa_{i-1} \overset{w}{\Longrightarrow} \kappa_i) = P^{r'}_{\mathrm{adj}}(w')$ where $r' = r(d, \kappa_{i-1}, w)$ and $w' = l(d, \kappa_{i-1}, w)$
3. (non-adjoining) $P_d(\kappa_{i-1} \overset{\neg w}{\Longrightarrow} \kappa_i) = 1 - P^{r'}_{\mathrm{adj}}(w')$ where r' and w' are defined as in the activation case
4. (adjoining) $P_d(\kappa_{i-1} \overset{w,r}{\Longrightarrow} \kappa_i) = P(r)$.

In order to describe the generative model of G, we impose a deterministic strategy sel on the derivation relation in order to obtain, for every sentential form, a probability distribution among the follow-up sentential forms. A *deterministic derivation strategy* is a mapping $\mathrm{sel} : \mathrm{SF}(G) \to (\mathbb{N}^* \times \mathbb{N}^*) \cup \{\bot\}$ such that for every $\kappa = (\xi_s, \xi_t, V, W, g) \in \mathrm{SF}(G)$, we have that $\mathrm{sel}(\kappa) \in V \cup W$ if $V \cup W \neq \emptyset$, and $\mathrm{sel}(\kappa) = \bot$ otherwise. In other words, sel chooses the next site to operate on. Then we define $\Rightarrow_{\mathrm{sel}}$ in the same way as $\Rightarrow$ but in each of the cases we require that $w = \mathrm{sel}(\kappa_1)$. Moreover, for every derivation $d \in D$, we denote by $\mathrm{next}(d)$ the set of all derivations of the form $du\kappa$ where $\mathrm{last}(d) \overset{u}{\Rightarrow}_{\mathrm{sel}} \kappa$.

The generative model of G comprises all the generative stories of G. A *generative story* is a tree $t \in U_D$; the root of t is labeled by κ_{in}. Let $w \in \mathrm{pos}(t)$ and $t(w) = d$. Then either w is a leaf, because we have stopped the generative story at w, or w has $|\mathrm{next}(d)|$ children, each one represents exactly one possible decision about how to extend d by a single derivation step (where their order does not matter). Then, for every generative story t, we have that

$$\sum_{w \in \mathrm{lv}(t)} P(t(w)) = 1 \ .$$

We note that (D, next, μ) can be considered as a discrete Markov chain (cf., e.g. (Baier et al., 2009)) where the initial probability distribution $\mu : D \to [0, 1]$ maps $d = \kappa_{\mathrm{in}}$ to 1, and all the other derivations to 0.

A *probabilistic synchronous tree insertion grammar* (pSTIG) G is a pSTAG except that for every rule $r = (\zeta_s, \zeta_t, V, W, P^r_{\mathrm{adj}})$ or $r = (\zeta_s, \zeta_t, V, W, *, P^r_{\mathrm{adj}})$ we have that

- if $r \in \mathcal{A}$, then $|\mathrm{lv}(\zeta_s)| \ge 2$ and $|\mathrm{lv}(\zeta_t)| \ge 2$,
- for $* = (*_s, *_t)$ we have that $*_s$ is either the rightmost leaf of ζ_s or its leftmost one; then we call r, resp., *L-auxiliary in the source* and *R-auxiliary in the source*; similarly, we restrict $*_t$; the *source-spine of* r (*target-spine of* r) is the set of prefixes of $*_s$ (resp., of $*_t$)
- $W \subseteq \mathrm{nlv}_N(\zeta_s) \times \{\mathrm{L}, \mathrm{R}\} \times \mathrm{nlv}_N(\zeta_t) \times \{\mathrm{L}, \mathrm{R}\}$ where the new components are the *direction-type* of the potential adjoining site, and
- for every $(w_s, \delta_s, w_t, \delta_t) \in W$, if w_s lies on the source-spine of r and r is L-auxiliary (R-auxiliary) in the source, then $\delta_s = \mathrm{L}$ (resp., $\delta_s = \mathrm{R}$), and corresponding restrictions hold for the target component.

According to the four possibilities for the foot-node $*$ we call r LL-, LR-, RL-, or RR-auxiliary. The restriction for the probability distribution P of G is modified such that for every $(A, B) \in N \times N$ and $x, y \in \{\mathrm{L}, \mathrm{R}\}$:

$$\sum_{\substack{r \in \mathcal{A},\ \mathrm{rc}(r) = (A,B) \\ r \text{ is } xy-\text{auxiliary}}} P(r) = 1 \ .$$

In the derivation relation of the pSTIG G we will have to make sure that the direction-type of the chosen adjoining site w matches with the type of auxiliarity of the auxiliary rule. Again we assume that the data structure $\mathrm{SF}(G)$ is enriched such that for every potential adjoining site w of $\kappa \in \mathrm{SF}(G)$ we know its direction-type $\mathrm{dir}(w)$.

We define the derivation relation of the pSTIG G to be the binary relation $\Rightarrow_I \subseteq \mathrm{SF}(G) \times \mathrm{SF}(G)$ such that we have $\kappa_1 \Rightarrow_I \kappa_2$ iff (i) $\kappa_1 \Rightarrow \kappa_2$ and

[4] We note that a different definition occurs in (Nesson et al., 2005, 2006).

(ii) if adjoining takes place at w, then the used auxiliary rule must be $\mathrm{dir}(w)$-auxiliary. Since $\Rightarrow_I$ is a subset of $\Rightarrow$, the concepts of derivation, successful derivation, and tree transformation are defined also for a pSTIG.

In fact, our running example pSTAG in Fig. 1 is a pSTIG, where r_2 and r_3 are RL-auxiliary and every potential adjoining site has direction-type RL; the derivation shown in Fig. 2 is a pSTIG-derivation.

4 Bottom-up tree adjoining transducer

Here we introduce the concept of a bottom-up tree adjoining transducer (BUTAT) which will be used to formalize a decoder for a pSTIG.

A BUTAT is a finite-state machine which translates strings into trees. The left-hand side of each rule is a string over terminal symbols and state-variable combinations. A variable is either a substitution variable or an adjoining variable; a substitution variable (resp., adjoining variable) can have an output tree (resp., output tree with foot node) as value. Intuitively, each variable value is a translation of the string that has been reduced to the corresponding state. The right-hand side of a rule has the form $q(\zeta)$ where q is a state and ζ is an output tree (with or without foot-node); ζ may contain the variables from the left-hand side of the rule. Each rule has a probability $p \in [0, 1]$.

In fact, BUTAT can be viewed as the string-to-tree version of bottom-up tree transducers (Engelfriet, 1975; Gecseg and Steinby, 1984,1997) in which, in addition to substitution, adjoining is allowed.

Formally, we let $X = \{x_1, x_2, \ldots\}$ and $F = \{f_1, f_2, \ldots\}$ be the sets of *substitution variables* and *adjoining variables*, resp. Each substitution variable (resp., adjoining variable) has rank 0 (resp., 1). Thus when used in a tree, substitution variables are leaves, while adjoining variables have a single child.

A *bottom-up tree adjoining transducer* (BUTAT) is a tuple $M = (Q, \Gamma, \Delta, Q_f, R)$ where

- Q is a finite set (of *states*),
- Γ is an alphabet (of *input symbols*), assuming that $Q \cap \Gamma = \emptyset$,
- Δ is an alphabet (of *output symbols*),
- $Q_f \subseteq Q$ (set of *final states*), and
- R is a finite set of rules of the form

$$\gamma_0\, q_1(z_1)\, \gamma_1\, \ldots\, q_k(z_k)\, \gamma_k \xrightarrow{p} q(\zeta) \quad (\dagger)$$

where $p \in [0, 1]$ (*probability* of $(\dagger)$), $k \geq 0$, $\gamma_0, \gamma_1, \ldots, \gamma_k \in \Gamma^*$, $q, q_1, \ldots, q_k \in Q$, $z_1, \ldots, z_k \in X \cup F$, and $\zeta \in \mathrm{RHS}(k)$ where $\mathrm{RHS}(k)$ is the set of all trees over $\Delta \cup \{z_1, \ldots, z_k\} \cup \{*\}$ in which the nullary $*$ occurs at most once.

The set of *intermediate results of* M is the set $\mathrm{IR}(M) = \{\iota \mid \iota \in U_\Delta(\{*\}), |\mathrm{pos}_{\{*\}}(\iota)| \leq 1\}$ and the set of *sentential forms of* M is the set $\mathrm{SF}(M) = (\Gamma \cup \{q(\iota) \mid q \in Q, \iota \in \mathrm{IR}(M)\})^*$. The *derivation relation induced by* M is the binary relation $\Rightarrow \subseteq \mathrm{SF}(M) \times \mathrm{SF}(M)$ such that for every $\xi_1, \xi_2 \in \mathrm{SF}(M)$ we define $\xi_1 \Rightarrow \xi_2$ iff there are $\xi, \xi' \in \mathrm{SF}(M)$, there is a rule of the form $(\dagger)$ in R, and there are $\zeta_1, \ldots, \zeta_k \in \mathrm{IR}(M)$ such that:

- for every $1 \leq i \leq k$: if $z_i \in X$, then ζ_i does not contain $*$; if $z_i \in F$, then ζ_i contains $*$ exactly once,
- $\xi_1 = \xi\, \gamma_0\, q_1(\zeta_1)\, \gamma_1\, \ldots\, q_k(\zeta_k)\, \gamma_k\, \xi'$, and
- $\xi_2 = \xi\, q(\theta(\zeta))\, \xi'$

where θ is a function that replaces variables in a right-hand side with their values (subtrees) from the left-hand side of the rule. Formally, $\theta : \mathrm{RHS}(k) \rightarrow \mathrm{IR}(M)$ is defined as follows:

(i) for every $\xi = \delta(\xi_1, \ldots, \xi_n) \in \mathrm{RHS}(k)$, $\delta \in \Delta$, we have $\theta(\xi) = \delta(\theta(\xi_1), \ldots, \theta(\xi_n))$,
(ii) (substitution) for every $z_i \in X$, we have $\theta(z_i) = \zeta_i$,
(iii) (adjoining) for every $z_i \in F$ and $\xi \in \mathrm{RHS}(k)$, we have $\theta(z_i(\xi)) = \zeta_i[\theta(\xi)]_v$ where v is the uniquely determined position of $*$ in ζ_i, and
(iv) $\theta(*) = *$.

Clearly, the probablity of a rule carries over to derivation steps that employ this rule. Since, as usual, a derivation d is a sequence of derivation steps, we let the *probability of* d be the product of the probabilities of its steps.

The *string-to-tree transformation computed by* M is the set τ_M of all tuples $(\gamma, \xi) \in \Gamma^* \times U_\Delta$ such that there is a derivation of the form $\gamma \Rightarrow^* q(\xi)$ for some $q \in Q_f$.

5 Decoder for pSTIG

Now we construct the decoder $\mathrm{dec}(G)$ for a pSTIG G that transforms source strings directly into target trees and simultaneously computes the probability of the corresponding derivation of G. This decoder is formalized as a BUTAT.

Since $\mathrm{dec}(G)$ is a string-to-tree transducer, we have to transform the source tree ζ_s of a rule r into a left-hand side ρ of a $\mathrm{dec}(G)$-rule. This is done similarly to (DeNeefe and Knight, 2009) by traversing ζ_s via recursive descent using a mapping φ (see an example after Theorem 1); this creates appropriate state-variable combinations for all substitution sites and potential adjoining sites of r. In particular, the source component of the direction-type of a potential adjoining site determines the position of the corresponding combination in ρ. If there are several potential adjoining sites with the same source component, then we create a ρ for every permutation of these sites. The right-hand side of a $\mathrm{dec}(G)$-rule is obtained by traversing the target tree ζ_t via recursive descent using a mapping ψ_ρ and, whenever a nonterminal with a potential adjoining site w is met, a new position labeled by f_w is inserted.[5] If there is more than one potential adjoining site, then the set of all those sites is ordered as in the left-hand side ρ from top to bottom.

Apart from these main rules we will employ rules which implement the decision of whether or not to turn a potential adjoining site w into an activated adjoining site. Rules for the first purpose just pass the already computed output tree through from left to right, whereas rules for the second purpose create for an empty left-hand side the output tree $*$.

We will use the state behavior of $\mathrm{dec}(G)$ in order to check that (i) the nonterminals of a substitution or potential adjoining site match the root-category of the used rule, (ii) the direction-type of an adjoining site matches the auxiliarity of the chosen auxiliary rule, and (iii) the decisions of whether or not to adjoin for each rule r of G are kept separate.

Whereas each pair (ξ_s, ξ_t) in the translation of G is computed in a top-down way, starting at the initial sentential form and substituting and adjoining to the present sentential form, $\mathrm{dec}(G)$ builds ξ_t in a bottom-up way. This change of direction is legitimate, because adjoining is associative (Vijay-Shanker and Weir, 1994), i.e., it leads to the same result whether we first adjoin r_2 to r_1, and then align r_3 to the resulting tree, or first adjoin r_3 to r_2, and then adjoin the resulting tree to r_1.

In Fig. 3 we show some rules of the decoder of our running example pSTIG and in Fig. 4 the derivation of this decoder which correponds to the derivation in Fig. 2.

[5] We will allow variables to have structured indices that are not elements of $\mathbb{N}$. However, by applying a bijective renaming, we can always obtain rules of the form (†).

Theorem 1. Let G be a pSTIG over N and T. Then there is a BUTAT $\mathrm{dec}(G)$ such that for every $(\xi_s, \xi_t) \in U_N(T) \times U_N(T)$ and $p \in [0,1]$ the following two statements are equivalent:

1. there is a successful derivation of $(\xi_s, \xi_t, \emptyset, \emptyset, \emptyset)$ by G with probability p,
2. there is a derivation from $\mathrm{yield}(\xi_s)$ to $[S_s, S_t](\xi_t)$ by $\mathrm{dec}(G)$ with probability p.

PROOF. Let $G = (N, T, [S_s, S_t], \mathcal{S}, \mathcal{A}, P)$ be a pSTIG. We will construct the BUTAT $\mathrm{dec}(G) = (Q, T, N \cup T, \{[S_s, S_t]\}, R)$ as follows (where the mappings φ and ψ_ρ will be defined below):

- $Q = [N \times N] \cup [N \times \{\mathrm{L}, \mathrm{R}\} \times N \times \{\mathrm{L}, \mathrm{R}\}]$ $\cup \{[r, w] \mid r \in \mathcal{A}, w$ is an adjoining site of $r\}$,
- R is the smallest set R' of rules such that for every $r \in \mathcal{S} \cup \mathcal{A}$ of the form $(\zeta_s, \zeta_t, V, W, P^r_{\mathrm{adj}})$ or $(\zeta_s, \zeta_t, V, W, *, P^r_{\mathrm{adj}})$:
  - for every $\rho \in \varphi(\varepsilon)$, if $r \in \mathcal{S}$, then the main rule

$$\rho \xrightarrow{P(r)} [\zeta_s(\varepsilon), \zeta_t(\varepsilon)]\big(\psi_\rho(\varepsilon)\big)$$

    is in R', and if $r \in \mathcal{A}$ and r is $\delta_s\delta_t$-auxiliary, then the main rule

$$\rho \xrightarrow{P(r)} [\zeta_s(\varepsilon), \delta_s, \zeta_t(\varepsilon), \delta_t]\big(\psi_\rho(\varepsilon)\big)$$

    is in R', and
  - for every $w = (w_s, \delta_s, w_t, \delta_t) \in W$ the rules

$$q_w\big(f_w\big) \xrightarrow{P^r_{\mathrm{adj}}(w)} [r, w]\big(f_w(*)\big)$$

    with $q_w = [\zeta(w_s), \delta_s, \zeta_t(w_t), \delta_t]$ for activation at w, and the rule

$$\varepsilon \xrightarrow{1-P^r_{\mathrm{adj}}(w)} [r, w](*)$$

    for non-adjoining at w are in R'.

We define the mapping

$$\varphi : \mathrm{pos}(\zeta_s) \to \mathcal{P}((T \cup Q(X \cup F))^*)$$

with $Q(X \cup F) = \{q(z) \mid q \in Q, z \in X \cup F\}$ inductively on its argument as follows. Let $w \in \mathrm{pos}(\zeta_s)$ and let w have n children.

(a) Let $\zeta_s(w) \in T$. Then $\varphi(w) = \{\zeta_s(w)\}$.

(b) (substitution site) Let $\zeta_s(w) \in N$ and let $w' \in \mathrm{pos}(\zeta_t)$ such that $(w, w') \in V$. Then

$$\varphi(w) = \{[\zeta_s(w), \zeta_t(w')]\big(x_{(w,w')}\big)\}.$$

(c) (adjoining site) Let $\zeta_s(w) \in N$ and let there be an adjoining site in W with w as first component. Then, we define $\varphi(w)$ to be the smallest set such that for every permutation $(u_1, \ldots, u_l)$ (resp., $(v_1, \ldots, v_m)$) of all the L-adjoining (resp., R-adjoining) sites in W with w as first component, the set[6]

$$J \circ \varphi(w.1) \circ \ldots \circ \varphi(w.n) \circ K$$

is a subset of $\varphi(w)$, where $J = \{u'_1 \ldots u'_l\}$ and $K = \{v'_m \ldots v'_1\}$ and

$$u'_i = [r, u_i]\big(f_{u_i}\big) \text{ and } v'_j = [r, v_j]\big(f_{v_j}\big)$$

for $1 \leq i \leq l$ and $1 \leq j \leq m$.

(d) Let $\zeta_s(w) \in N$, $w \neq *$, and let w be neither the first component of a substitution site in V nor the first component of an adjoining site in W. Then

$$\varphi(w) = \varphi(w.1) \circ \ldots \circ \varphi(w.n) \ .$$

(e) Let $w = *$. Then we define $\varphi(w) = \{\varepsilon\}$.

For every $\rho \in \varphi(\varepsilon)$, we define the mapping

$$\psi_\rho : \mathrm{pos}(\zeta_t) \to U_{N \cup F \cup X}(T \cup \{*\})$$

inductively on its argument as follows. Let $w \in \mathrm{pos}(\zeta_t)$ and let w have n children.

(a) Let $\zeta_t(w) \in T$. Then $\psi_\rho(w) = \zeta_t(w)$.

(b) (substitution site) Let $\zeta_t(w) \in N$ and let $w' \in \mathrm{pos}(\zeta_s)$ such that $(w', w) \in V$. Then $\psi_\rho(w) = x_{(w',w)}$.

(c) (adjoining site) Let $\zeta_t(w) \in N$ and let there be an adjoining site in W with w as third component. Then let $\{u_1, \ldots, u_l\} \subseteq W$ be the set of all potential adjoining sites with w as third component, and we define

$$\psi_\rho(w) = f_{u_1}(\ldots f_{u_l}(\zeta) \ldots)$$

where $\zeta = \zeta_t(w)(\psi_\rho(w.1), \ldots, \psi_\rho(w.n))$ and the u_i's occur in $\psi_\rho(w)$ (from the root towards the leaves) in exactly the same order as they occur in ρ (from left to right).

(d) Let $\zeta_t(w) \in N$, $w \neq *$, and let w be neither the second component of a substitution site in V nor the third component of an adjoining site in W. Then

$$\psi_\rho(w) = \zeta_t(w)(\, \psi_\rho(w.1), \ldots, \psi_\rho(w.n)).$$

(e) Let $w = *$. Then $\psi_\rho(w) = *$.

With $\mathrm{dec}(G)$ constructed as shown, for each derivation of G there is a corresponding derivation of $\mathrm{dec}(G)$, with the same probability, and vice versa. The derivations proceed in opposite directions. Each sentential form in one has an equivalent sentential form in the other, and each step of the derivations correspond. There is no space to present the full proof, but let us give a slightly more precise idea about the formal relationship between the derivations of G and $\mathrm{dec}(G)$.

In the usual way we can associate a derivation tree d^t with every successful derivation d of G. Assume that $\mathrm{last}(d) = (\xi_s, \xi_t, \emptyset, \emptyset, \emptyset)$, and let E_s and E_t be the embedded tree transducers (Shieber, 2006) associated with, respectively, the source component and the target component of G. Then it was shown in (Shieber, 2006) that $\tau_{E_s}(d^t) = \xi_s$ and $\tau_{E_t}(d^t) = \xi_t$ where τ_E denotes the tree-to-tree transduction computed by an embedded tree transducer E. Roughly speaking, E_s and E_t reproduce the derivations of, respectively, the source component and the target component of G that are prescribed by d^t. Thus, for $\kappa = (\xi'_s, \xi'_t, V, W, g)$, if $\kappa_{in} \Rightarrow^*_G \kappa$ and κ is a prefix of d, then there is exactly one subtree $d^t[(w, w')]$ of d^t associated with every $(w, w') \in V \cup W$, which prescribes how to continue at (w, w') with the reproduction of d. Having this in mind, we obtain the sentential form of the $\mathrm{dec}(G)$-derivation which corresponds to κ by applying a modification of φ to κ where the modification amounts to replacing $x_{(w,w')}$ and $f_{(w,w')}$ by $\tau_{E_t}(d^t[(w, w')])$; note that $\tau_{E_t}(d^t[(w, w')])$ might contain $*$. ■

As illustration of the construction in Theorem 1 let us apply the mappings φ and ψ_ρ to rule r_2 of Fig. 1, i.e., to $r_2 = (\zeta_s, \zeta_t, \emptyset, \{b, c\}, *, P^{r_2}_{\mathrm{adj}})$ with $\zeta_s = A(A, \gamma)$, $\zeta_t = B(B(\delta), B)$, $b = (\varepsilon, \mathrm{R}, \varepsilon, \mathrm{L})$, $c = (\varepsilon, \mathrm{R}, 1, \mathrm{L})$, and $* = (1, 2)$.

Let us calculate $\varphi(\varepsilon)$ on ζ_s. Due to (c),

$$\varphi(\varepsilon) = J \circ \varphi(1) \circ \varphi(2) \circ K.$$

Since there are no L-adjoinings at ε, we have that $J = \{\varepsilon\}$. Since there are the R-adjoinings b and c at ε, we have the two permutations (b, c) and (c, b).

$\underline{(v_1, v_2) = (b, c)}$: $K = \{[r_2, c](f_c)[r_2, b](f_b)\}$

$\underline{(v_1, v_2) = (c, b)}$: $K = \{[r_2, b](f_b)[r_2, c](f_c)\}$

Due to (e) and (a), we have that $\varphi(1) = \{\varepsilon\}$ and $\varphi(2) = \{\gamma\}$, resp. Thus, $\varphi(\varepsilon)$ is the set:

$$\{\gamma\, [r_2, c](f_c)\, [r_2, b](f_b),\ \gamma\, [r_2, b](f_b)\, [r_2, c](f_c)\}.$$

[6] using the usual concatenation $\circ$ of formal languages

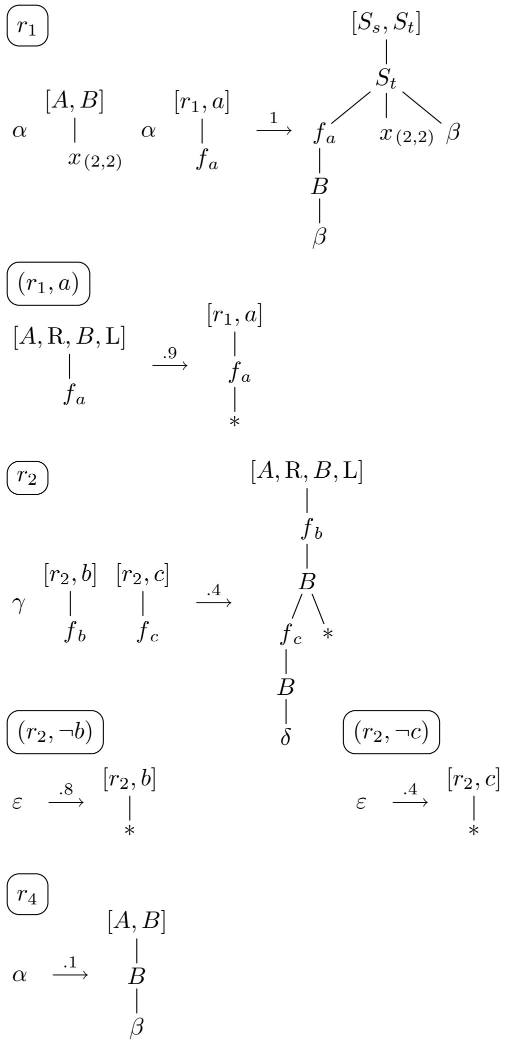

Figure 3: Some rules of the running example decoder.

Now let $\rho = \gamma\,[r_2, b](f_b)\,[r_2, c](f_c)$. Let us calculate $\psi_\rho(\varepsilon)$ on ζ_t.

$$\begin{aligned}
\psi_\rho(\varepsilon) &\overset{(c)}{=} f_b(B(\psi_\rho(1), \psi_\rho(2))) \\
&\overset{(c)}{=} f_b(B(f_c(B(\psi_\rho(1.1))), \psi_\rho(2))) \\
&\overset{(a)}{=} f_b(B(f_c(B(\delta)), \psi_\rho(2))) \\
&\overset{(e)}{=} f_b(B(f_c(B(\delta)), *))
\end{aligned}$$

Hence we obtain the rule

$$\gamma\,[r_2, b](f_b)\,[r_2, c](f_c) \to$$

$$[A, \mathrm{R}, B, \mathrm{L}](f_b(B(f_c(B(\delta)), *)))$$

which is also shown in Fig. 3.

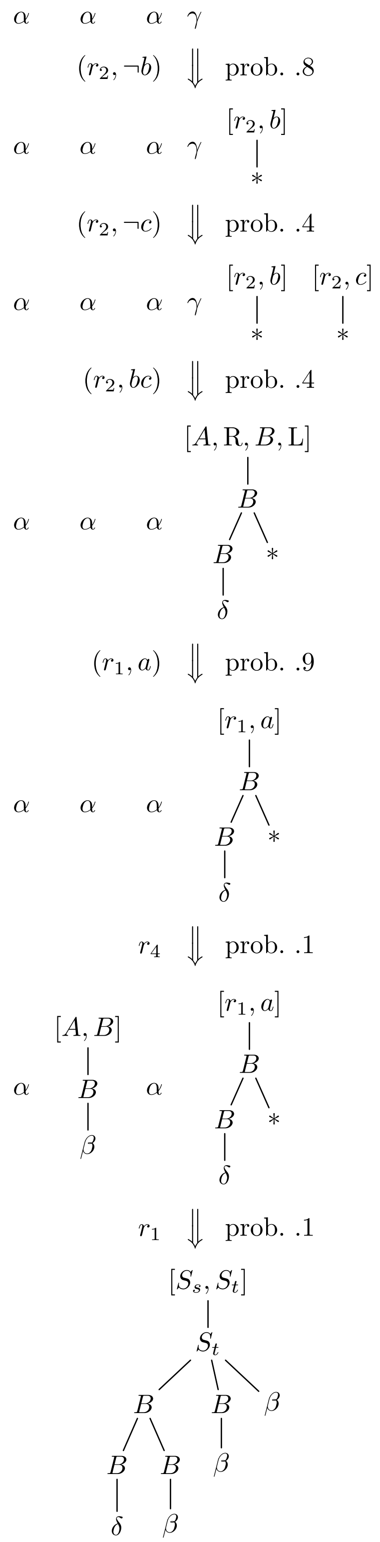

Figure 4: Derivation of the decoder corresponding to the derivation in Fig. 2.

References

A. Abeille, Y. Schabes, A.K. Joshi. Using lexicalized TAGs for machine translation. In *Proceedings of the 13th International Conference on Computational Linguistics*, volume 3, pp. 1–6, Helsinki, Finland, 1990.

C. Baier, M. Größer, F. Ciesinski. Model checking linear-time properties of probabilistic systems. In *Handbook of Weighted Automata*, Chapter 13, pp. 519–570, Springer, 2009.

S. DeNeefe. Tree adjoining machine translation. Ph.D. thesis proposal, Univ. of Southern California, 2009.

S. DeNeefe, K. Knight. Synchronous tree adjoining machine translation. In *Proc. of Conf. Empirical Methods in NLP*, pp. 727–736, 2009.

J. Engelfriet. Bottom-up and top-down tree transformations — a comparison. *Math. Systems Theory*, 9(3):198–231, 1975.

J. Engelfriet. Tree transducers and syntax-directed semantics. In *CAAP 1982: Lille, France*, 1982.

A. Fujiyoshi, T. Kasai. Spinal-formed context-free tree grammars. *Theory of Computing Systems*, 33:59–83, 2000.

Z. Fülöp, H. Vogler. Weighted tree automata and tree transducers. In *Handbook of Weighted Automata*, Chapter 9, pp. 313–403, Spinger, 2009.

F. Gécseg, M. Steinby. *Tree Automata*. Akadémiai Kiadó, Budapest, 1984.

F. Gécseg, M. Steinby. Tree languages. In *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer-Verlag, 1997.

J. Graehl, K. Knight, J. May. Training tree transducers. *Computational Linguistics*, 34(3):391–427, 2008

A.K. Joshi, L.S. Levy, M. Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163, 1975.

A.K. Joshi, Y. Schabes. Tree-adjoining grammars. In *Handbook of Formal Languages*. Chapter 2, pp. 69–123, Springer-Verlag, 1997.

K. Knight, J. Graehl. An overview of probabilistic tree transducers for natural language processing. In *Computational Linguistics and Intelligent Text Processing, CICLing 2005*, LNCS 3406, pp. 1–24, Springer, 2005.

K. Knight, J. May. Applications of Weighted Automata in Natural Language Processing. In *Handbook of Weighted Automata*, Chapter 14, pp. 571–596, Springer, 2009.

P.M. Lewis, R.E. Stearns. Syntax-directed transductions. *Journal of the ACM*, 15:465–488, 1968.

A. Maletti. Compositions of tree series transformations. *Theoret. Comput. Sci.*, 366:248–271, 2006.

A. Maletti. The Power of Tree Series Transducers. Ph.D. thesis, TU Dresden, Germany, 2006.

A. Maletti. Compositions of extended top-down tree transducers. *Information and Computation*, 206:1187–1196, 2008.

A. Maletti. Why synchronous tree substitution grammars? in *Proc. 11th Conf. North American Chapter of the Association of Computational Linguistics*. 2010.

D.F. Martin and S.A. Vere. On syntax-directed transductions and tree transducers. In *Ann. ACM Symposium on Theory of Computing*, pp. 129–135, 1970.

R. Nesson, S.M. Shieber, and A. Rush. Induction of probabilistic synchronous tree-insertion grammars. Technical Report TR-20-05, Computer Science Group, Harvard Univeristy, Cambridge, Massachusetts, 2005.

R. Nesson, S.M. Shieber, and A. Rush. Induction of probabilistic synchronous tree-inserting grammars for machine translation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA 2006)*, 2006.

Y. Schabes, R.C. Waters. Tree insertion grammars: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21:479–513, 1994.

P.P. Schreiber. Tree-transducers and syntax-connected transductions. In *Automata Theory and Formal Languages*, Lecture Notes in Computer Science 33, pp. 202–208, Springer, 1975.

S.M. Shieber. Synchronous grammars and tree transducers. In *Proc. 7th Workshop on Tree Adjoining Grammars and Related Formalisms*, pp. 88–95, 2004.

S.M. Shieber. Unifying synchronous tree-adjoining grammars and tree transducers via bimorphisms. In *Proc. 11th Conf. European Chapter of ACL, EACL 06*, pp. 377–384, 2006.

S.M. Shieber, Y. Schabes. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, volume 3, pp. 253–258, Helsinki, Finland, 1990.

K. Vijay-Shanker, D.J. Weir. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:511–546, 1994.

K. Yamada and K. Knight. A syntax-based statistical translation model. In *Proc. of 39th Annual Meeting of the Assoc. Computational Linguistics*, pp. 523–530, 2001.