Compositions of Tree-to-Tree Statistical Machine Translation Models

Andreas Maletti *

Universität Stuttgart, Institute for Natural Language Processing
Pfaffenwaldring 5b, 70569 Stuttgart, Germany
maletti@ims.uni-stuttgart.de

Abstract. Compositions of well-known tree-to-tree translation models used in statistical machine translation are investigated. Synchronous context-free grammars are closed under composition in both the unweighted as well as the weighted case. In addition, it is demonstrated that there is a close connection between compositions of synchronous tree-substitution grammars and compositions of certain tree transducers because the intermediate trees can encode finite-state information. Utilizing these close ties, the composition closure of synchronous tree-substitution grammars is identified in the unweighted and weighted case. In particular, in the weighted case, these results build on a novel lifting strategy that will prove useful also in other setups.

1 Introduction

Several different translation models are nowadays used in syntax-based statistical machine translation [17]. The translation model is the main component responsible for the transformation of the input into the translated output, and thus the expressive power of the translation model limits the possible translations. For example, the framework 'Moses' [18] provides implementations of synchronous context-free grammars (SCFGs) [1] and several variants of synchronous tree-substitution grammars (STSGs) [6]. The expressive power of SCFGs and STSGs is reasonably well-understood, and in particular, knowledge of the limitations of the models has helped many authors to pre-process [26,5,20] or post-process [4,25] their data and to achieve better translation results. Together with pre- or post-processing steps, the translation model is no longer solely responsible for the transformation process, but we rather obtain a composition of several models or simply a composition chain [23]. Occasionally, composition chains also appear because they ideally support a modular development of components for specific translation tasks [3] (e.g., translating numerals or geographic locations). However, it is often difficult to evaluate such composition chains efficiently especially when the pre- or post-processing steps are nondeterministic.

In the string-to-string setting the phrase-based models are essentially finite-state transducers and chains of them can be collapsed into a single transducer [24]

* Supported by the German Research Foundation (DFG) grant MA / 4959 / 1-1.

because they are closed under composition. However, this is not true for several tree-to-tree models. Efficient on-the-fly evaluations for composition chains are presented in [23] along with the observation that the straightforward sequential evaluation of composition chains is terribly inefficient. Even in the on-the-fly evaluation the chains should be as short as possible. In this contribution, we will investigate the expressive power of composition chains of the established tree-to-tree translation models. The symmetric tree-to-tree setting, although typically worse in terms of translation quality than the string-to-tree or tree-to-string setting, is particularly convenient since it allows a clean notion of composition.

We first demonstrate that (unweighted and weighted) composition chains of SCFGs can always be reduced to just a single SCFG. In addition, we demonstrate how to utilize results for unweighted extended tree transducers [22] to obtain results for STSGs. The main insight in this part is that even local models like STSGs obtain a finite-state behavior in composition chains. Thus, a composition of two STSGs is as powerful as a composition of two corresponding tree transducers. This close connection allows us to show that two STSGs are necessary and sufficient for arbitrary composition chains of certain simple, yet commonly used STSGs. These results hold in the absence of weights. However, all translation models used in statistical machine translation are weighted, so as a second contribution we demonstrate how to lift the unweighted results into the weighted setting. Our novel lifting procedure, which we believe will be useful also in other setups, relies on a separation of the weights and several normalization procedures. Overall, we achieve the same results also in the weighted setting, which essentially shows that short chains of certain STSGs suffice.

2 Preliminaries

Let us start with some basic notions for trees, which we depict graphically whenever possible. Formally, our trees use a finite set N of internal labels and a finite set L of leaf labels. The internal labels can label any non-leaf node of the tree and such labeled nodes can have any positive number of children, whereas leaf labels only label leaves; i.e., nodes without children. Thus, our trees are inductively defined to be the smallest set $T_N(L)$ such that (i) every leaf node labeled $\ell \in L$ is a tree $\ell \in T_N(L)$ and (ii) $n(t_1, \ldots, t_k) \in T_N(L)$ is a tree consisting of a root node labeled n and k direct subtrees for any given positive integer k, internal label $n \in N$, and trees $t_1, \ldots, t_k \in T_N(L)$. A tree t that consists only of a (non-leaf) root node and leaf nodes is shallow, so a shallow tree is of the form $n(\ell_1, \ldots, \ell_k)$ for some $n \in N$, $k \geq 1$, and $\ell_1, \ldots, \ell_k \in L$. To easily access information in a tree, we use the following notation. For each node ν in a tree $t \in T_N(L)$ we write $t(\nu)$ for the label of the node ν. Occasionally, we are interested in the leaf nodes that are labeled by certain leaf labels $Q \subseteq L$. Consequently, the set of all nodes that are leaves and labeled by an element of Q is denoted by $\mathrm{leaves}_Q(t)$, and the elements of $\mathrm{leaves}_Q(t)$ are called anchors for substitution. Moreover, given another tree $u \in T_N(L)$ and a leaf $\nu \in \mathrm{leaves}_Q(t)$, we write $t[\nu \leftarrow u]$ for the

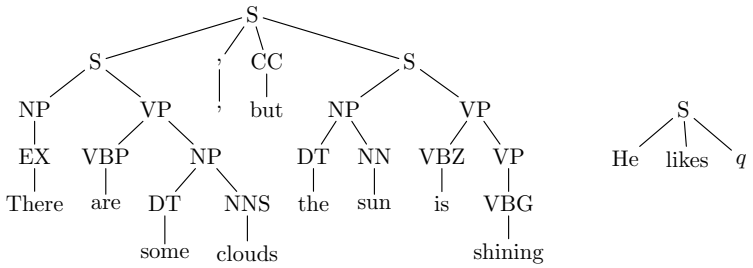

Fig. 1. The left tree t is not shallow, whereas the right tree u is. If $Q = \{q\}$, then $\mathrm{leaves}_Q(t) = \emptyset$ and $\mathrm{leaves}_Q(u) = \{\nu\}$, where ν is the q-labeled node in u. Obviously, its label is $u(\nu) = q$. Moreover, $u[\nu \leftarrow \mathrm{her}] = \mathrm{S}(\mathrm{He}, \mathrm{likes}, \mathrm{her})$. Note that the q-labeled node vanishes in the substitution.

tree obtained from t by replacing the leaf ν by the tree u. These notations are illustrated in Fig. 1, and we refer to [12,13] for an in-depth exposition.

Our weights will be taken from commutative semirings [16,14], which are algebraic structures $(A, +, \cdot, 0, 1)$ such that $(A, +, 0)$ and $(A, \cdot, 1)$ are commutative monoids and $(\sum_{i=1}^{k} a_i) \cdot a = \sum_{i=1}^{k} (a_i \cdot a)$ for all non-negative integers k and $a, a_1, \ldots, a_k \in A$. Typical examples of such semirings include the Boolean semiring $(\{0,1\}, \max, \min, 0, 1)$, the Viterbi semiring $([0,1], \max, \cdot, 0, 1)$ on the unit interval $[0,1]$, and the semiring $(\mathbb{Q}, +, \cdot, 0, 1)$ of rational numbers. In the following, let $(A, +, \cdot, 0, 1)$ be an arbitrary commutative semiring. Similarly, we fix the finite sets N and L of default internal labels and leaf labels, respectively.

A weighted (linear, nondeleting extended top-down) tree transducer [15,9] is a tuple $T = (Q, \Sigma, (q_1, q_2), R, \mathrm{wt})$ consisting of (i) a finite set Q of states, (ii) a finite set Σ of internal labels for the trees generated, (iii) designated initial states $q_1, q_2 \in Q$, (iv) a finite set R of rules of the form $(q, t) \xrightarrow{\varphi} (q', t')$ consisting of states $q, q' \in Q$, input and output tree fragments $t, t' \in T_\Sigma(L \cup Q)$,[1] and a bijective alignment $\varphi\colon \mathrm{leaves}_Q(t) \to \mathrm{leaves}_Q(t')$, and (v) a rule weight assignment $\mathrm{wt}\colon R \to A$. The transducer T is a synchronous tree-substitution grammar (STSG) [6] if $Q = \Sigma$ and in each rule $(q, t) \xrightarrow{\varphi} (q', t') \in R$ the root labels of t and t' are q and q', respectively.[2] Roughly speaking, an STSG replaces the "hidden" finite-state behavior by locality tests because the root labels (i.e., the states of a rule) are visible in the input and output tree fragments. Finally, T is a synchronous context-free grammar (SCFG) [1] if it is an STSG and in each rule $(q, t) \xrightarrow{\varphi} (q', t') \in R$ the trees t and t' are shallow. In an SCFG, the input and the output tree are assembled like derivation trees of a context-free grammar (i.e., one level at a time). We recall two restrictions on tree transducer rules. A rule $(q, t) \xrightarrow{\varphi} (q', t')$ is an ε-rule if $t \in Q$. Similarly, it is a non-strict rule if $t' \in Q$. The tree transducer T is ε-free if it does not contain any ε-rules in R, and it is strict provided that it has no non-strict rules in R. Finally, simple tree

[1] For technical reasons we disallow that $\{t, t'\} \subseteq Q$.

[2] Note that in an STSG the elements of Σ can label internal nodes and leaves.

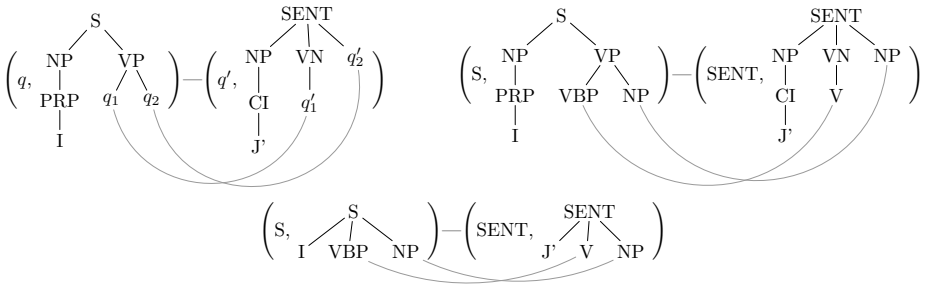

Fig. 2. Example rules of a tree transducer [top left], an STSG [top right], and an SCFG [bottom].

transducers are both ε-free and strict. Note that an SCFG is always simple. We show a few example rules of each type in Fig. 2.

Let us recall the derivation semantics [10] of a weighted tree transducer $T = (Q, \Sigma, (q_1, q_2), R, \mathrm{wt})$. The derivations are defined over rule-like triples of the form (u, ψ, u') consisting of a (partial) input tree $u \in T_\Sigma(L \cup Q)$, a bijective alignment $\psi\colon \mathrm{leaves}_Q(u) \to \mathrm{leaves}_Q(u')$ linking synchronous states, and a (partial) output tree $u' \in T_\Sigma(L \cup Q)$. Note that the derivation forms are thus essentially rules without the states. Given two such forms (u, ψ, u') and (s, ψ', s') and a rule $\rho = (q, t) \xrightarrow{\varphi} (q', t') \in R$, we say that (u, ψ, u') derives (s, ψ', s') via ρ, written $(u, \psi, u') \Rightarrow_T^\rho (s, \psi', s')$ if the least element $\nu \in \mathrm{leaves}_Q(u)$ with respect to some arbitrary linear order on nodes is such that (i) the node ν is labeled $u(\nu) = q$, (ii) its synchronized node $\psi(\nu)$ in u' has label $u'(\psi(\nu)) = q'$, (iii) $s = u[\nu \leftarrow t]$ is obtained from u by replacing ν by t, (iv) $s' = u'[\psi(\nu) \leftarrow t']$ is obtained from u' by replacing $\psi(\nu)$ by t', and (v) the synchronization ψ' is given for every $\nu' \in \mathrm{leaves}_Q(s)$ by $\psi'(\nu') = \varphi(\nu')$ if $\nu' \in \mathrm{leaves}_Q(t)$ and $\psi'(\nu') = \psi(\nu')$ otherwise. In other words, we keep the old synchronized states (except the replaced ones) and add the synchronized states of the rule ρ.[3] We illustrate a derivation step in Fig. 3. The derivation process starts with the initial form $\xi_0 = (q_1, \psi_0, q_2)$, in which the root nodes ν_1 and ν_2 of the trees q_1 and q_2, respectively, are synchronized (i.e., $\psi_0(\nu_1) = \nu_2$). Given trees $t, t' \in T_\Sigma(L)$, the transducer T assigns the weight

$$T(t, t') = \sum_{\xi_0 \Rightarrow_T^{\rho_1} \xi_1 \Rightarrow_T^{\rho_2} \cdots \Rightarrow_T^{\rho_n} (t, \emptyset, t')} \left(\prod_{i=1}^{n} \mathrm{wt}(\rho_i) \right)$$

to the pair (t, t'). We note that this sum always remains finite. Intuitively, we sum up the weights of all derivations of the tree pair (t, t'), where the weight of the derivation is obtained by multiplying the rule weights used in the derivation. In this manner, the transducer T computes a mapping $T\colon T_\Sigma(L) \times T_\Sigma(L) \to A$.

Finally, let us formally introduce compositions of weighted tree-to-tree translations. For all alphabets Σ and Δ, a mapping $\tau\colon T_\Sigma(L) \times T_\Delta(L) \to A$ is fini-

[3] For simplicity, we assume that nodes in different trees are disjoint.

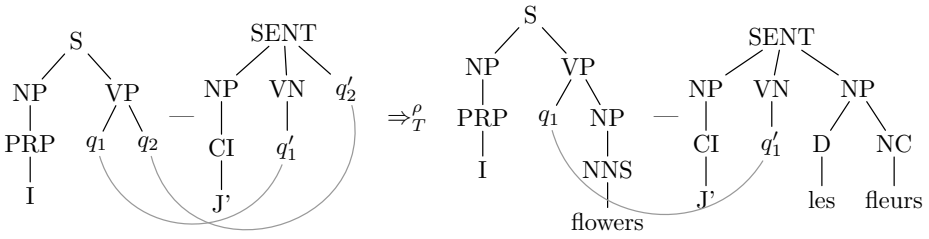

Fig. 3. The rule ρ: $(q_2, \mathrm{NP}(\mathrm{NNS}(\mathrm{flowers}))) \xrightarrow{\emptyset} (q_2', \mathrm{NP}(\mathrm{D}(\mathrm{les}), \mathrm{NC}(\mathrm{fleurs})))$ used in a derivation step. Note that the states q_2 and q_2' completely disappear.

tary, if for every $t \in T_\Sigma(L)$ there exist only finitely many $u \in T_\Delta(L)$ such that $\tau(t,u) \neq 0$. Similarly, it is co-finitary, if for every $u \in T_\Delta(L)$ there exist only finitely many $t \in T_\Sigma(L)$ such that $\tau(t,u) \neq 0$. Now let $\tau\colon T_\Sigma(L) \times T_\Delta(L) \to A$ and $\tau'\colon T_\Delta(L) \times T_\Gamma(L) \to A$ be such that τ is finitary or τ' is co-finitary. Then the composition τ followed by τ', written $\tau\,;\tau'$, is defined for every $t \in T_\Sigma(L)$ and $s \in T_\Gamma(L)$ by

$$(\tau\,;\tau')(t,s) = \sum_{u \in T_\Delta(L)} \tau(t,u) \cdot \tau'(u,s) \ .$$

Note that this sum is well-defined because of the finitary or co-finitary restriction, which yields that only finitely many choices of u yield non-zero products. Roughly speaking, we sum over all potential intermediate trees u and take the product of the weights for the translation from t to u and the translation from u to s, which shows that composition corresponds to executing the second transducer on the output of the first transducer. Composition extends to classes $\mathcal{C}$ of weighted translations in the usual manner, and we use $\mathcal{C}^n$ for the composition $\mathcal{C}\,;\cdots;\,\mathcal{C}$ containing the class $\mathcal{C}$ exactly n times.

3 Unweighted compositions

Let us first collect what is known about the unweighted case, which is obtained using the Boolean semiring $(\{0,1\}, \max, \min, 0, 1)$ as weight structure. In this setting, tree-to-tree translations are essentially relations on trees. It is evident from the formal definitions that each SCFG is a special STSG, which in turn is a special tree transducer, so the expressive power increases from SCFGs to STSGs to tree transducers (TTs). Using the abbreviations as denotations for the classes of tree relations that can be generated by the corresponding translation model, we thus have SCFG $\subseteq$ STSG $\subseteq$ TT. Moreover, the key property that separates SCFGs and STSGs was identified in [6]. The relations computed by SCFGs only contain pairs of isomorphic trees (disregarding the labels and the order of the children). It is also easy to show that STSGs and TTs can be separated, so we obtain the strict hierarchy

$$\mathrm{SCFG} \subset \mathrm{STSG} \subset \mathrm{TT} \ . \tag{1}$$

Table 1. Known results on composition closures.

Model	Composition closure	Reference
top-down tree transducer	1	[7]
simple tree transducer	2	[2]
other tree transducer	∞	[8]

Composition essentially corresponds to running two translations consecutively, where the first translation translates the input into intermediate results and the second translation translates those intermediate results into the final results. Compositions of tree translations have been extensively investigated (see [11] for a survey). To avoid a careful distinction between transducers and their translations, we will conflate the class of models with the class of translations computable by it. Since all classes discussed here contain the identity relation, compositions of our classes $\mathcal{C}$ form a natural hierarchy; i.e., $\mathcal{C} \subseteq \mathcal{C}^2 \subseteq \mathcal{C}^3 \subseteq \cdots$. This hierarchy collapses at level n if $\mathcal{C}^n = \mathcal{C}^{n+1}$. We also say that the composition closure is obtained at level n provided that n is the least integer, for which the hierarchy collapses. Intuitively, if the closure is obtained at level n, then compositions of n translations of $\mathcal{C}$ are necessary and sufficient to generate any translation computable by any composition of $\mathcal{C}$. Provided that the composition closure for $\mathcal{C}$ is n, we thus have $\mathcal{C} \subset \cdots \subset \mathcal{C}^n = \mathcal{C}^{n+1} = \cdots$. We use ∞ to indicate that the hierarchy never collapses. We summarize the known results [7,2,8] on the composition closure in Table 1.

We start our investigation with SCFGs. Given two SCFGs T_1 and T_2 we can simply "join" rules of T_1 and T_2 that coincide on the intermediate tree. We illustrate this approach in Fig. 4. Such rules can certainly be executed consecutively in the on-the-fly approach [23]. A refined version of this approach taking the finite-state information and the non-shallow output into account is used to prove that (our linear and nondeleting) top-down tree transducers are closed under composition [7].

Theorem 1. *The composition closure of unweighted and weighted SCFGs is achieved at the first level.*

Proof. We prove the statement for arbitrary weighted SCFGs. Let

$$T = (\Sigma, \Sigma, (S_1, S_2), R, \mathrm{wt}) \qquad \text{and} \qquad T' = (\Sigma', \Sigma', (S_1', S_2'), R', \mathrm{wt}')$$

be weighted SCFGs. If $S_2 \neq S_1'$, then the composition $T\,;T'$ is the constant 0 mapping, which can easily be computed by a single SCFG. Now suppose that $S_2 = S_1'$. We construct the weighted SCFG $T'' = (\Sigma'', \Sigma'', (S_1, S_2'), R'', \mathrm{wt}'')$, where $\Sigma'' = \Sigma \cup \Sigma'$ and the rules R'' and their weights wt'' are obtained as follows: For every rule $\rho = (\sigma, s) \xrightarrow{\varphi} (\delta, t)$ of R and rule $\rho' = (\delta, t) \xrightarrow{\psi} (\gamma, u)$ of R' we construct the rule $\rho'' = (\sigma, s) \xrightarrow{\varphi;\psi} (\gamma, u)$ of R'' and set $\mathrm{wt}''(\rho'') = \mathrm{wt}(\rho) \cdot \mathrm{wt}'(\rho')$. No other rules are in R''. The correctness of this construction is straightforward. □

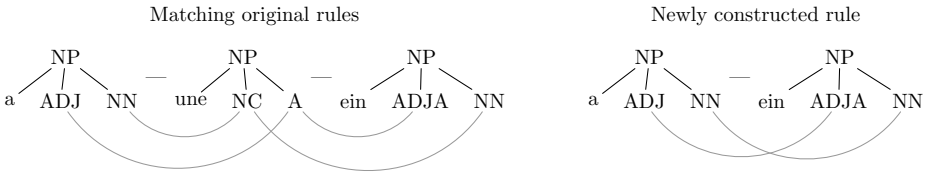

Fig. 4. Rule matching and joining in the composition of SCFGs. The left and middle part form a rule of T_1 and the middle and right part form a rule of T_2. The newly constructed rule will simply avoid the intermediate tree fragment.

Next, we will show that the composition closure for simple STSGs can be obtained from the known results via a small insight. Recall that SCFGs and STSGs are both local, so they are missing the hidden finite-state behavior of general tree transducers. However, we can simulate the hidden finite-state behavior for both models in compositions with the help of the unknown (hidden) intermediate trees. Namely, we can annotate the desired finite-state information on the intermediate trees in the spirit of the representation of a regular tree language as the image of a local tree language under a relabeling [13]. We illustrate the approach in Fig. 5. Note that the first STSG encodes the states in its output (i.e., the intermediate tree), whereas the second STSG encodes them in its input (i.e., also the intermediate tree).

Lemma 1. *For all $n \geq 2$, compositions of n simple STSGs are as expressive as compositions of n simple tree transducers.*

Proof. We only provide the argument for compositions $T\,;T'$ of 2 simple STSGs T and T'. Assume that $(q,t) \xrightarrow{\varphi} (q',t')$ is a rule of the first simple tree transducer $T = (Q, \Sigma, (q_1, q_2), R, \mathrm{wt})$. Since T is strict, we have $t' = \delta(t'_1, \ldots, t'_k)$ for some internal symbol $\delta \in \Sigma$ and subtrees $t'_1, \ldots, t'_k$. We will adjust the internal symbols to $\Sigma' = \Sigma \cup (\Sigma \times Q \times Q)$, which allows us to use combinations of internal symbols together with two states. For every state-labeled node $\nu \in \mathrm{leaves}_Q(t')$ in t' we additionally guess two internal symbols $\sigma_\nu, \delta_\nu \in \Sigma$. Then we construct the rule $\big(\sigma, u\big) \xrightarrow{\varphi} \big((\delta, q, q')(\delta, q, q')(u'_1, \ldots, u'_k)\big)$, where σ is the root label of t, $u = t[\varphi^{-1}(\nu) \leftarrow \sigma_\nu \mid \nu \in \mathrm{leaves}_Q(t')]$, and the subtrees $u'_1, \ldots, u'_k$ are obtained from the subtrees $t'_1, \ldots, t'_k$ by replacing each leaf node $\nu \in \mathrm{leaves}_Q(t')$ by the state-annotated variant $(\delta_\nu, t(\varphi^{-1}(\nu)), t'(\nu))$. In other words, we guess the internal symbols that will replace a state leaf in the input and output fragment t and t' and replace the state leaf by the guessed internal symbol in the input fragment and the triple containing the guessed internal symbol and the two synchronized states. We construct a new rule for each original rule and all possible guesses. Similarly, we need to annotate the finite-state information of the second tree transducer T' in its input fragments, which works in essentially the same manner using the input fragments instead of the output fragments. This also shows that we actually need to annotate up to 4 states to each symbol in the intermediate tree, and we additionally need to guess the finite-state information (that

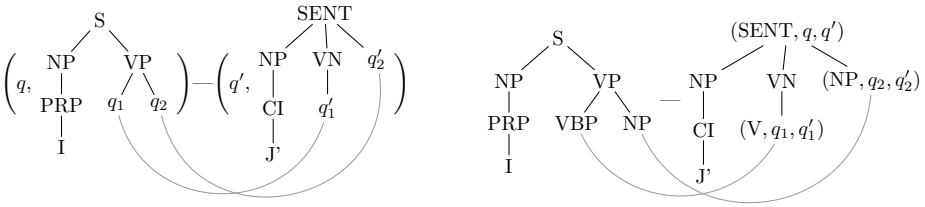

Fig. 5. Illustration of the state annotation on the intermediate tree. The left part shows the original tree transducer rule and the right STSG rule shows how the state annotation is performed on the output tree using the guessed nonterminal pairs $(\mathrm{VBP}, \mathrm{V})$ and $(\mathrm{NP}, \mathrm{NP})$ for (q_1, q_1') and (q_2, q_2'), respectively.

Table 2. Composition closure results for unweighted and weighted SCFGs and STSGs. They mirror the corresponding results for tree transducers.

Model	Unweighted / weighted composition closure	Results
SCFGs	1	Theorem 1
simple STSGs	2	Theorems 2 and 4
other STSGs	∞	Theorems 3 and 5

can also occur in internal symbols) of the other tree transducer. We omit the technical details. □

Theorem 2. *The composition closure of simple STSGs is obtained at the second level.*

Proof. Simple tree transducers achieve the composition closure at level 2 [2]. Since the second levels of the composition hierarchy for simple tree transducers and simple STSGs coincide by Lemma 1 and simple STSGs are less expressive by (1), the composition closure of simple STSGs is achieved at level 2 as well. □

Finally, we examine the composition hierarchy of the remaining cases (non-strict STSGs and STSGs with ε-rules). In both cases, the corresponding hierarchy for tree transducers is infinite. Moreover, re-examining the counterexample translation τ provided in [8, Example 43], we can easily see that it does not utilize its finitely many states and can be generated by a non-strict STSG as well. Hence for every $n \geq 1$ we also obtain a translation τ^{n+1} that can be computed by $(n+1)$ STSGs, but not by n tree transducers according to [8, Lemma 44]. Since by (1) we have $\mathrm{STSG} \subseteq \mathrm{TT}$, it follows that $\mathrm{STSG}^n \subseteq \mathrm{TT}^n$ and thus n STSGs also cannot implement τ^{n+1}. The analogous arguments using the inverse translation τ^{-1} can be used to prove the infiniteness of the composition hierarchy for STSGs with ε-rules. We summarize the results in Table 2.

Theorem 3. *The composition hierarchy of strict STSGs, ε-free STSGs, and general STSGs is infinite.*

4 Weighted compositions

In the weighted setting, which is more relevant in statistical machine translation, the models assign a weight to each rule. During derivations the weights of the participating rules are multiplied, and if there are several ways to achieve the same input- and output-tree pair, then the derivation weights are summed up. To avoid infinite summations, we restrict ourselves to ε-free or strict models.

The goal of this section is to lift the unweighted results of the previous section into the weighted setting. In Theorem 1 we already proved that SCFGs are closed under composition also in the weighted case. Moreover, the result of Lemma 1 also holds in the weighted case, so the composition closure of simple weighted STSGs and that of simple weighted tree transducers again coincide. It only remains to establish the composition closure for simple weighted tree transducers. Roughly speaking, we will reduce the weighted problem to the unweighted setting by removing the weights from the tree transducer and moving them into a particularly simple type of translation, called weighted relabeling. For the ease of presentation we assume that no rule consists only of a leaf in the input or output tree fragment (i.e., $t \notin L$ and $t' \notin L$ for all considered rules $(q,t) \xrightarrow{\varphi} (q',t')$). This is realistic in statistical machine translation since the parsers usually attach at least a part-of-speech tag to each lexical item. Moreover, we can easily adjust our approach and relabel leaf symbols as well.

For a given alphabet Σ, a *weighted relabeling* is a mapping $\kappa\colon \Sigma \times \Sigma \to A$. In other words, it is a weighted association between symbols. It extends to pairs of trees such that it assigns weight 0 to all pairs of trees of different shape. For trees of the same shape, it simply takes the product of the symbol-to-symbol weights given by κ for all corresponding nodes in the two trees. Formally, each such relabeling κ extends to a weighted tree translation $\overline{\kappa}\colon T_\Sigma(L) \times T_\Sigma(L) \to A$ inductively by (i) $\overline{\kappa}(\ell,\ell) = 1$ for every $\ell \in L$; i.e., we do not relabel leaf symbols, and (ii) for every $k \geq 1$, symbols $\sigma, \delta \in \Sigma$, subtrees $t_1, \ldots, t_k, u_1, \ldots, u_k \in T_\Sigma(L)$

$$\overline{\kappa}\big(\sigma(t_1,\ldots,t_k), \delta(u_1,\ldots,u_k)\big) = \kappa(\sigma,\delta) \cdot \prod_{i=1}^{k} \overline{\kappa}(t_i,u_i) \ .$$

We relabel trees with an internal symbol as root by charging the weight for relabeling the root symbol to another symbol and then multiply the product of the weights of recursively relabeling the subtrees. In all remaining cases, $\overline{\kappa}(\ldots,\ldots) = 0$. We use wREL for the class of all weighted translations computable by weighted relabelings and s-wTT for the corresponding class computed by simple weighted tree transducers. Since most devices in this section are weighted, we will drop the explicit mention that they are weighted and simply say 'relabeling' or 'simple tree transducer'.

Lemma 2. *For every composition of a simple tree transducer and a relabeling in either order, we can present an equivalent simple tree transducer.*

$$\text{s-wTT} \,;\, \text{wREL} \subseteq \text{s-wTT} \qquad \textit{and} \qquad \text{wREL} \,;\, \text{s-wTT} \subseteq \text{s-wTT}$$

Proof. The first statement is obtained by combining the decomposition of [9, Lemma 4.1] and the composition results of [19, Theorem 2.4]. Moreover, since all the involved models are symmetric, we also immediately obtain the second statement. □

The next lemma shows that we can separate the weights from a simple tree transducer leaving a composition of an essentially unweighted (i.e., unambiguous and Boolean[4]) simple tree transducer T' and a relabeling. Moreover, the tree relation computed by T' will be injective.[5] *Unambiguous* means that for each (successful) translation (t, u) containing an input and an output tree there exists exactly one derivation yielding (t, u). We use su-$\mathrm{TT}_{\mathrm{inj}}$ for the injective translations computed by simple unambiguous tree transducers. Note that these weighted translations are essentially the characteristic functions of the translations of the corresponding unweighted tree transducers, which motivates the chosen abbreviation.

Lemma 3. *Every simple tree transducer T can be equivalently represented by a composition of a simple unambiguous Boolean tree transducer T' computing an injective translation followed by a relabeling κ.*

$$\text{s-wTT} \subseteq \text{su-TT}_{\text{inj}} \,;\, \text{wREL}$$

Proof. Let $T = (Q, \Sigma, (q_1, q_2), R, \mathrm{wt})$. For every rule $\rho = (q, t) \xrightarrow{\varphi} (q', t')$ of R, we have $t' = \sigma(t'_1, \ldots, t'_k)$ for some integer k, symbol $\sigma \in \Sigma$, and subtrees $t'_1, \ldots, t'_k \in T_\Sigma(L \cup Q)$ because T is strict. For this rule ρ, we construct the rule $(q, t) \xrightarrow{\varphi} (q', \langle \sigma, \rho \rangle(t'_1, \ldots, t'_k))$ of T', which essentially records the rule application in the root of the output tree fragment. The weight of this new rule is 1 in T'. Finally, the relabeling κ is such that $\kappa(\sigma, \sigma) = 1$ and $\kappa(\langle \sigma, \rho \rangle, \sigma) = \mathrm{wt}(\rho)$ for all $\sigma \in \Sigma$ and $\rho \in R$, and 0 otherwise. In other words, the relabeling removes the annotation and charges the weight of the annotated rule. Obviously, the constructed tree transducer is Boolean. In addition, since the derivation is completely visible in the output, the tree transducer T' is unambiguous. □

Using Lemmas 2 and 3 we can now separate the weights from a composition chain because

$$\begin{aligned}
\text{s-wTT} \,;\, \text{s-wTT}^2 &\subseteq \text{su-TT}_{\text{inj}} \,;\, \text{wREL} \,;\, \text{s-wTT}^2 && \text{(Lemma 3)} \\
&\subseteq \text{su-TT}_{\text{inj}} \,;\, \text{s-wTT}^2 \subseteq \text{su-TT}^2_{\text{inj}} \,;\, \text{wREL} \,;\, \text{s-wTT} && \text{(Lemmas 2 and 3)} \\
&\subseteq \text{su-TT}^2_{\text{inj}} \,;\, \text{s-wTT} \subseteq \text{su-TT}^3_{\text{inj}} \,;\, \text{wREL} \;. && \text{(Lemmas 2 and 3)}
\end{aligned}$$

Now we can apply the result of [2] on the unweighted composition closure of s-TT. Note that the composition of injective translations is naturally again injective.

$$\text{su-TT}^3_{\text{inj}} \,;\, \text{wREL} \subseteq \underbrace{\text{s-TT}^2}_{\text{injective}} \,;\, \text{wREL} \;,$$

[4] using only the weights 0 and 1

[5] A tree translation $\tau \colon T_\Sigma(L) \times T_\Sigma(L) \to A$ is *injective* if for every output tree $u \in T_\Sigma(L)$ there exists at most one input tree $t \in T_\Sigma(L)$ such that $\tau(t, u) \neq 0$.

where s-TT is the class of translations computed by simple unweighted tree transducers. We cannot simply simulate those unweighted tree transducers directly by weighted tree transducers. We first use standard techniques (regular restrictions; see [22]) to make both unweighted translations injective. Moreover each injective translation can be made unambiguous using essentially the same techniques, so we obtain

$$\underbrace{\text{s-TT}^2}_{\text{injective}} \,;\, \text{wREL} \subseteq \text{su-TT}^2_{\text{inj}} \,;\, \text{wREL} \subseteq \text{s-wTT}^2 \ ,$$

where the last step uses Lemma 2. Note that unambiguous unweighted tree transducers can easily be simulated by the corresponding weighted tree transducers. Thus, we derived the difficult part of the composition closure.

Theorem 4. *The composition closure of weighted simple STSGs is achieved at the second level.*

Proof. We showed that $\text{s-wTT}^3 \subseteq \text{s-wTT}^2$, so the composition hierarchy of the class s-wTT collapses at level 2. Moreover using the linking arguments of [21] we can also conclude that $\text{s-wTT} \subset \text{s-WTT}^2$. □

Finally, for the remaining classes (i.e., strict STSGs and ε-free STSGs), we can essentially import the infinite composition hierarchy from the unweighted case using the linking technique of [21]. We omit the details.

Theorem 5. *The composition hierarchy of strict weighted STSGs and ε-free weighted STSGs is infinite.*

Conclusion

We have investigated the expressive power of compositions of the well-established tree-to-tree translation models: SCFGs, STSGs, and tree transducers. In the unweighted case, the results for the local devices [i.e., SCFGs and STSGs] closely mirror the known composition results for tree transducers due to the fact that we can encode the finite-state information in the intermediate trees of a composition. The same picture presents itself in the weighted setting, for which we showed how to lift the corresponding results from the unweighted setting to the weighted setting. This uses a novel decomposition separating the weights from simple tree transducers and then constructions for the obtained unambiguous and injective tree transducers. Overall, we demonstrated that in the relevant cases, short (length 1 or 2) composition chains are necessary and sufficient to simulate arbitrarily long composition chains.

References

1. Aho, A.V., Ullman, J.D.: Syntax directed translations and the pushdown assembler. J. Comput. System Sci. 3(1), 37–56 (1969)

2. Arnold, A., Dauchet, M.: Morphismes et bimorphismes d'arbres. Theoret. Comput. Sci. 20(1), 33–93 (1982)
3. Chen, S., Matsumoto, T.: Translation of quantifiers in Japanese-Chinese machine translation. In: Proc. JapTAL. LNAI, vol. 7614, pp. 11–22. Springer (2012)
4. Clifton, A., Sarkar, A.: Combining morpheme-based machine translation with post-processing morpheme prediction. In: Proc. ACL. pp. 32–42. ACL (2011)
5. Collins, M., Koehn, P., Kucerová, I.: Clause re-structuring for statistical machine translation. In: Proc. ACL. pp. 531–540. ACL (2005)
6. Eisner, J.: Learning non-isomorphic tree mappings for machine translation. In: Proc. ACL. pp. 205–208. ACL (2003)
7. Engelfriet, J.: Bottom-up and top-down tree transformations: A comparison. Math. Systems Theory 9(3), 198–231 (1975)
8. Engelfriet, J., Fülöp, Z., Maletti, A.: Composition closure of linear extended top-down tree transducers. Theory Comput. Syst. (2016), to appear
9. Fülöp, Z., Maletti, A., Vogler, H.: Weighted extended tree transducers. Fundam. Inform. 111(2), 163–202 (2011)
10. Fülöp, Z., Vogler, H.: Weighted tree transducers. J. Autom. Lang. Combin. 9(1), 31–54 (2004)
11. Fülöp, Z., Vogler, H.: Weighted tree automata and tree transducers. In: Droste, M., Kuich, W., Vogler, H. (eds.) Handbook of Weighted Automata, chap. 9, pp. 313–403. Springer (2009)
12. Gécseg, F., Steinby, M.: Tree Automata. Akadémiai Kiadó, Budapest (1984)
13. Gécseg, F., Steinby, M.: Tree Automata. arXiv 1509.06233 (2015)
14. Golan, J.S.: Semirings and their Applications. Kluwer Academic (1999)
15. Graehl, J., Knight, K.: Training tree transducers. In: Proc. HLT-NAACL. pp. 105–112. ACL (2004)
16. Hebisch, U., Weinert, H.J.: Semirings—Algebraic Theory and Applications in Computer Science. World Scientific (1998)
17. Koehn, P.: Statistical Machine Translation. Cambridge University Press (2010)
18. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: Open source toolkit for statistical machine translation. In: Proc. ACL. pp. 177–180. ACL (2007)
19. Kuich, W.: Full abstract families of tree series I. In: Jewels Are Forever, pp. 145–156. Springer (1999)
20. Lerner, U., Petrov, S.: Source-side classifier preordering for machine translation. In: Proc. EMNLP. pp. 513–523. ACL (2013)
21. Maletti, A.: The power of weighted regularity-preserving multi bottom-up tree transducers. Int. J. Found. Comput. Sci. 26(7), 987–1005 (2015)
22. Maletti, A., Graehl, J., Hopkins, M., Knight, K.: The power of extended top-down tree transducers. SIAM J. Comput. 39(2), 410–430 (2009)
23. May, J., Knight, K., Vogler, H.: Efficient inference through cascades of weighted tree transducers. In: Proc. ACL. pp. 1058–1066. ACL (2010)
24. Mohri, M.: Finite-state transducers in language and speech processing. Comput. Linguist. 23(2), 269–311 (1997)
25. Stymne, S.: Text Harmonization Strategies for Phrase-Based Statistical Machine Translation. Ph.D. thesis, Linköping University (2012)
26. Xia, F., McCord, M.C.: Improving a statistical MT system with automatically learned rewrite patterns. In: Proc. CoLing. pp. 508–514 (2004)