Compositions of Constant Weighted Extended Tree Transducers

Malte Blattmann[0000−0003−2282−0723] and Andreas Maletti[0000−0003−3202−0498]

Universität Leipzig, Faculty of Mathematics and Computer Science, PO Box 100 920, 04009 Leipzig, Germany
malteblattmann@gmx.de, andreas.maletti@uni-leipzig.de

Abstract. Conjecture 11 of [Lagoutte, Maletti: Survey — Weighted extended top-down tree transducers — Part III: Composition. Proc. AFCS, LNCS 7020, p. 272–308, Springer 2011] is confirmed. It is demonstrated that the composition of a constant weighted extended tree transducer with a linear weighted top-down tree transducer can be computed by a single weighted extended tree transducer. Whereas linearity and the top-down property are syntactic, the constant property is semantic. The decidability of the constant property is investigated in several restricted settings.

Keywords: Weighted tree transducer · Top-down tree transducer · Extended tree transducer · Composition · Decidability

1 Introduction

Weighted tree transducers [5,7,16] are a straightforward generalization of classical tree transducers [23,24,25] such that each rule carries a weight from a semiring. They compute a weighted relation on trees, which assigns a weight to each pair of an input and an output tree. Overall, they thus allow a much more fine-grained classification of the input-output relation. A good overview of weighted tree transducers is presented in [8].

The weighted extended tree transducers [12,18,20] have been introduced to model certain syntax-based translation systems in machine translation [15] and have also been utilized in that capacity [12]. Whereas (non-extended) tree transducers permit only a single input symbol in the left-hand side of each rule, the extended variants allow arbitrary many input symbols in the left-hand side of their rules, which makes the model more symmetric. In the unweighted case, this asymmetry was noted much earlier and has been thoroughly investigated [3,2].

In this contribution we study compositions of certain weighted extended tree transducers. Composition is one of the basic operations on relations and can straightforwardly be extended to weighted relations. More precisely, given weighted relations $\tau_1 \colon A \times B \to \mathbb{Q}$ and $\tau_2 \colon B \times C \to \mathbb{Q}$ with weights in the rational numbers $\mathbb{Q}$, their composition is the weighted relation $\tau_1 \,;\, \tau_2$ given for every $a \in A$ and $c \in C$ by

$$(\tau_1 \,;\, \tau_2)(a, c) = \sum_{b \in B} \tau_1(a, b) \cdot \tau_2(b, c) \quad .$$

© Springer Nature Switzerland AG 2021
N. Moreira and R. Reis (Eds.): DLT 2021, LNCS 12811, pp. 66–77, 2021.
https://doi.org/10.1007/978-3-030-81508-0_6

Note that this composition is essentially a matrix product. Compositions of weighted relations naturally occur in the development of speech recognition systems [22], where the standard methodology composes from right-to-left a language-model transducer, a lexicon transducer, a context-dependency transducer, and a final HMM transducer each computing corresponding weighted relations. While the transducers for speech recognition usually work on strings, the transducers in syntax-based machine translation operate on trees [27] and individual components of the cascade reorder the subtrees of the input, insert additional subtrees, and finally translate the lexical entries. Representing the composed weighted relation computed by the cascade by just a single transducer offers significant advantages [1,12,21].

We continue the investigation started in [17] and confirm [17, Conjecture 11]. To this end, we show how to compose an arbitrary constant weighted extended tree transducer with a linear weighted top-down tree transducer. In other words, we require that the first transducer is constant, which is a semantic property and essentially states that a certain weight total does not depend on the actual input tree, but only on the state of the transducer. The second transducer needs to be linear (i.e., is not allowed to copy subtrees) and non-extended (i.e., handles a single input symbol in each rule). However, we place no constraints on the utilized weight structure. Our construction works for any commutative semiring [14,11]. Both main features, commutativity and distributivity, of the weight structure are heavily utilized in the construction. The history and corresponding unweighted composition results are discussed at length in [17].

Besides the construction, we offer an illustration of the problem that occurs in the standard composition construction of [17] and motivate the adjustment in this manner. A proof sketch for the correctness of the composition construction is provided. Since the constant property is semantic, it is not trivially decidable whether a given transducer is constant. In the final section, we explore a few cases, in which decidability of the constant property can be established.

2 Preliminaries

The nonnegative integers are $\mathbb{N}$, and we let $[n] = \{i \in \mathbb{N} \mid 1 \leq i \leq n\}$ for every $n \in \mathbb{N}$. Given relations $R \subseteq A \times B$ and $S \subseteq B \times C$ their composition $R\,;S$ is $R\,;S = \big\{(a,c) \in A \times C \mid \exists b \in B\colon (a,b) \in R,\, (b,c) \in S\big\}$. The inverse R^{-1}, domain $\mathrm{dom}(R)$, and range $\mathrm{ran}(R)$ of R are $R^{-1} = \{(b,a) \mid (a,b) \in R\}$,

$$\mathrm{dom}(R) = \big\{a \in A \mid \exists b \in B\colon (a,b) \in R\big\} \qquad \text{and} \qquad \mathrm{ran}(R) = \mathrm{dom}(R^{-1})\ .$$

Finite sets are also called alphabets, and for any alphabet A we let A^* be the set of all finite words over A including the empty word ε. The length of a word $w \in A^*$ is written as $|w|$, and for all $k \in \mathbb{N}$ we let $A^{\leq k} = \{w \in A^* \mid k \geq |w|\}$ be the words of length at most k.

A ranked alphabet (Σ, rk) consists of an alphabet Σ together with a mapping $\mathrm{rk}\colon \Sigma \to \mathbb{N}$ that assigns a *rank* to each element of Σ. For every $k \in \mathbb{N}$ we let $\Sigma^{(k)} = \{\sigma \in \Sigma \mid \mathrm{rk}(\sigma) = k\}$ be the set of all symbols of Σ that have rank k.

We write $\sigma^{(k)}$ to indicate that $\mathrm{rk}(\sigma) = k$. To simplify the notation, we often refer to the ranked alphabet (Σ, rk) by Σ alone. The set $T_\Sigma(A)$ of all trees is the smallest set T such that $A \subseteq T$ and $\sigma(t_1, \dots, t_k) \in T$ for all $k \in \mathbb{N}$, $\sigma \in \Sigma^{(k)}$, and $t_1, \dots, t_k \in T$. Instead of $T_\Sigma(\emptyset)$ we simply write T_Σ. Given a finite set Q and a subset $T \subseteq T_\Sigma(A)$, we let $Q(T) = \{q(t) \mid q \in Q, t \in T\} \subseteq T_{\Sigma \cup Q}(A)$, where each element of q is considered as a unary symbol. The *positions* of a tree $t \in T_\Sigma(A)$ are inductively defined by $\mathrm{pos}(a) = \{\varepsilon\}$ for all $a \in A$ and

$$\mathrm{pos}\big(\sigma(t_1, \dots, t_k)\big) = \{\varepsilon\} \cup \big\{iw \mid i \in [k], w \in \mathrm{pos}(t_i)\big\}$$

for all $k \in \mathbb{N}$, $\sigma \in \Sigma^{(k)}$, and $t_1, \dots, t_k \in T_\Sigma(A)$. Note that the positions $\mathrm{pos}(t)$ are totally ordered by the usual lexicographic order $\leq_{\mathrm{lex}}$. We write $t(w)$, $t|_w$, and $t[u]_w$ to refer to the symbol at position $w \in \mathrm{pos}(t)$ in the tree $t \in T_\Sigma(A)$, the subtree of t rooted in w, and the tree obtained from t by replacing the subtree rooted in w by the tree $u \in T_\Sigma(A)$, respectively. Formally, $a(\varepsilon) = a|_\varepsilon = a$ and $a[u]_\varepsilon = u$ for all $a \in A$ and for $t = \sigma(t_1, \dots, t_k)$

$$\begin{aligned} t(\varepsilon) &= \sigma & t(iw) &= t_i(w) & t|_\varepsilon &= t & t|_{iw} &= t_i|_w \\ t[u]_\varepsilon &= u & t[u]_{iw} &= \sigma(t_1, \dots, t_{i-1}, t_i[u]_w, t_{i+1}, \dots, t_k) \end{aligned}$$

for all $k \in \mathbb{N}$, $\sigma \in \Sigma^{(k)}$, $t_1, \dots, t_k \in T_\Sigma(A)$, $i \in [k]$, and $w \in \mathrm{pos}(t_i)$. Given labels $L \subseteq \Sigma \cup A$ we let $\mathrm{pos}_L(t) = \{w \in \mathrm{pos}(t) \mid t(w) \in L\}$ be the set of positions of t labeled by elements of L, and $\mathrm{pos}_a(t) = \mathrm{pos}_{\{a\}}(t)$ for all $a \in A$. We use the countable set $X = \{x_i \mid i \in \mathbb{N}\}$ of (formal) variables and its finite subsets $X_k = \{x_i \mid i \in [k]\}$ for every $k \in \mathbb{N}$. We let $\mathrm{var}(t) = \{x \in X \mid \mathrm{pos}_x(t) \neq \emptyset\}$ for every tree $t \in T_\Sigma(A \cup X)$. The tree t is called *linear* if $|\mathrm{pos}_x(t)| \leq 1$ for all $x \in X$. For every $V \subseteq X$, we let $C_\Sigma(V) = \{t \in T_\Sigma(V) \mid \mathrm{var}(t) = V, t \text{ linear}\}$ be the set of those trees that contain exactly one position labeled v for every $v \in V$. Given a *substitution* $\theta \colon V \to T_\Sigma(A \cup X)$ with $V \subseteq X$ finite, its application to a tree $t \in T_\Sigma(A \cup X)$ is given by $v\theta = \theta(v)$ for all $v \in V$, $x\theta = x$ for all $x \in X \setminus V$, $a\theta = a$ for all $a \in A$, and $\sigma(t_1, \dots, t_k)\theta = \sigma(t_1\theta, \dots, t_k\theta)$ for all $k \in \mathbb{N}$, $\sigma \in \Sigma^{(k)}$, and $t_1, \dots, t_k \in T_\Sigma(A \cup X)$. For every $t \in T_\Sigma$ we let

$$\mathrm{match}(t) = \big\{(c, \theta) \mid k \in \mathbb{N}, c \in C_\Sigma(X_k), \theta \colon X_k \to T_\Sigma, t = c\theta\big\} \ .$$

A (commutative) *semiring* [14,11] is an algebraic structure $(S, +, \cdot, 0, 1)$, in which $(S, +, 0)$ and $(S, \cdot, 1)$ are both commutative monoids, $s \cdot 0 = 0$ for all $s \in S$, and multiplication $\cdot$ distributes over addition $+$. The semiring is *idempotent* if $1 + 1 = 1$. Moreover, the semiring is *zero-sum free* if $s + s' = 0$ implies $s = 0$ for all $s, s' \in S$, and *zero-divisor free* if $s \cdot s' = 0$ implies $0 \in \{s, s'\}$ for all $s, s' \in S$. Note that every idempotent semiring is zero-sum free. If there exists $(-1) \in S$ such that $1 + (-1) = 0$, then S is a *ring*. Finally, given a mapping $f \colon A \to S$, we let $\mathrm{supp}(f) = \{a \in A \mid f(a) \neq 0\}$.

3 Weighted Extended Tree Transducers

Let us start by introducing the main tree transducer model, the weighted extended tree transducer [12,18,20], for which we want to study composition. It

is the weighted version of the bimorphism model studied in [3,2]. For convenience we use the minor syntactic variant of [17, Definition 3], which introduces an additional indirection via rule identifiers. For the rest of the contribution, let $(S, +, \cdot, 0, 1)$ be an arbitrary commutative semiring.

Definition 1 (see [17, Definition 3]). *A* weighted extended tree transducer *(for short: wxtt) is a tuple* $(Q, \Sigma, \Delta, Q_0, I, \chi)$*, in which*

- Q *is a finite set of* states
- Σ *and* Δ *are ranked alphabets of* input *and* output symbols*, respectively, such that* $Q \cap (\Sigma \cup \Delta) = \emptyset$*,*
- $Q_0 \subseteq Q$ *is a set of* initial states*,*
- I *is a finite set of* rule identifiers*, and*
- $\chi \colon I \to Q\big(T_\Sigma(X)\big) \times S \times T_\Delta\big(Q(X)\big)$ *assigns a* weighted rule $\chi(i) = \langle \ell, s, r \rangle$ *to each identifier* $i \in I$ *such that* $\ell \notin Q(X)$*,* ℓ *is linear, and* $\mathrm{var}(r) \subseteq \mathrm{var}(\ell)$*.*

In the following, let $M = (Q, \Sigma, \Delta, Q_0, I, \chi)$ be a wxtt. To simplify the notation, we also write $\ell \xrightarrow{s} r$ instead of $\langle \ell, s, r \rangle$. Moreover, for every $i \in I$ we let ℓ_i, s_i, and r_i be such that $\chi(i) = \langle \ell_i, s_i, r_i \rangle$. Since we can select the identifiers such that they uniquely determine M (i.e., different wxtt have disjoint sets of identifiers), the notation ℓ_i, s_i, and r_i should not lead to confusion. The wxtt M is called *linear* if r_i is linear for every $i \in I$, and it is called BOOLEAN if $s_i \in \{0, 1\}$ for every $i \in I$. Finally, M is a *weighted top-down tree transducer* (for short: wtdtt), if $|\mathrm{pos}_\Sigma(\ell_i)| = 1$ for all $i \in I$.

Next, we introduce the semantics of M. Later in our composition construction it proves to be convenient to handle symbols and states of another wxtt. To this end, we consider ranked alphabets Σ' and Δ' such that $\Sigma \subseteq \Sigma'$ and $\Delta \subseteq \Delta'$. Moreover, let $q \in Q$, $i \in I$, and $\xi \in T_{\Delta'}\big(Q(T_{\Sigma'}(X))\big)$, which we treat as a tree of $T_{\Delta' \cup Q \cup \Sigma'}(X)$. This treatment entails certain technical difficulties since the rank of each symbol needs to be unique, but we largely ignore those issues here in the interest of clarity. A position $w \in \mathrm{pos}_q(\xi)$ is *i-reducible* if there exists a substitution $\theta \colon \mathrm{var}(\ell_i) \to T_{\Sigma'}(X)$ such that $\xi|_w = \ell_i\theta$. Note that if such a substitution exists, then it is unique. Let $w \in \mathrm{pos}_Q(\xi)$ be the lexicographically least position labeled by a state. If w is i-reducible, then we let $i(\xi) = \xi\big[r_i\theta\big]_w$, which we also write as $\xi \overset{i}{\Rightarrow}_M \big[r_i\theta\big]_w$. Otherwise, $i(\xi)$ is undefined. Given $i_1, \ldots, i_n \in I$ we let $(i_1 \cdots i_n)(\xi) = i_n\big(\cdots i_1(\xi) \cdots\big)$, which we also write as $\xi \overset{i_1 \cdots i_n}{\Longrightarrow}_M (i_1 \cdots i_n)(\xi)$. A sequence $d \in I^*$ is called *derivation* for ξ if $d(\xi)$ is defined, and the finite set of all such derivations is denoted by $D_M(\xi)$. Moreover, we let

$$D_M^\perp(\xi) = \big\{ d \in D_M(\xi) \mid d(\xi) \in T_\Delta \big\}$$

be the subset of terminal derivations. Given a derivation $i_1 \cdots i_n \in D_M(\xi)$ with $n \in \mathbb{N}$ and $i_1, \ldots, i_n \in I$ we let $\mathrm{wt}_M(i_1 \cdots i_n) = \prod_{j=1}^n s_{i_j}$. Finally, we define the mapping $\tau'_M \colon T_{\Delta'}\big(Q(T_{\Sigma'}(X))\big) \times T_{\Delta'}\big(Q(T_{\Sigma'}(X))\big) \to S$ by

$$\tau'_M(\xi, \zeta) = \sum_{\substack{d \in D_M(\xi) \\ d(\xi) = \zeta}} \mathrm{wt}_M(d)$$

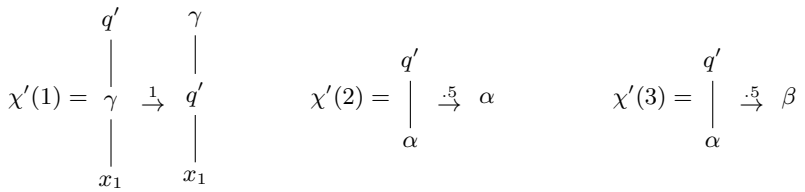

Fig. 1: Rules of the wtdtt M' of Example 3.

for all $\xi, \zeta \in T_{\Delta'}\big(Q(T_{\Sigma'}(X))\big)$. The semantics of M is the *weighted relation* $\tau_M \colon T_\Sigma \times T_\Delta \to S$ given by

$$\tau_M(t,u) = \sum_{q \in Q_0} \tau'_M\big(q(t), u\big)$$

for all trees $t \in T_\Sigma$ and $u \in T_\Delta$. The wxtt M is *total* if for all states $q \in Q$ and input trees $t \in T_\Sigma$ there is an output tree $u \in T_\Delta$ such that $\big(q(t), u\big) \in \operatorname{supp}(\tau'_M)$. Finally, M is *unambiguous* if for every state q and input tree $t \in T_\Sigma$ there exists at most one derivation $d \in D^\perp_M\big(q(t)\big)$.

Next, let us introduce the special property, for which we provide a composition construction. The property *constant* was introduced in [17, Definition 9] and essentially says that for any given state $q \in Q$ there exists a constant c_q such that for every input tree $t \in T_\Sigma$ the sum of all weights of derivations $d \in D^\perp_M\big(q(t)\big)$ is exactly c_q.

Definition 2 (see [17, Definition 9]). *Let $q \in Q$ and $c \in S$. State q is* c-constant *if $c = \sum_{d \in D^\perp_M(q(t))} \operatorname{wt}_M(d)$ for every $t \in T_\Sigma$. The wxtt M is* constant *if for every state $q \in Q$ there exists $c_q \in S$ such that q is c_q-constant.*

To conclude this section, let us quickly discuss a small example to illustrate the notions introduced in this section.

Example 3. Let $M' = (\{q'\}, \Sigma, \Delta, \{q'\}, \{1,2,3\}, \chi')$ be the wtdtt over the semiring $(\mathbb{R}, +, \cdot, 0, 1)$ with $\Sigma = \{\gamma^{(1)}, \alpha^{(0)}\}$, $\Delta = \{\gamma^{(1)}, \alpha^{(0)}, \beta^{(0)}\}$ and χ' presented in Fig. 1. Then for

$$t = \underbrace{\gamma\big(\cdots\gamma}_{n \text{ times } \gamma}(\alpha)\cdots\big) \qquad \text{we have} \qquad D^\perp_{M'}\big(q'(t)\big) = \big\{\underbrace{1\cdots1}_{n \text{ times}}2,\ \underbrace{1\cdots1}_{n \text{ times}}3\big\}$$

and those derivations have weight $s_1 \cdot \ldots \cdot s_1 \cdot s_2 = .5$ and $s_1 \cdot \ldots \cdot s_1 \cdot s_3 = .5$. The derivations are illustrated in Fig. 2 for $n = 2$. This illustrates that state q' is 1-constant, which also proves that M is constant.

4 Composition

Our overall goal is to settle [17, Conjecture 11], which deals with compositions of the weighted relations computed by certain wxtt. Before stating the con-

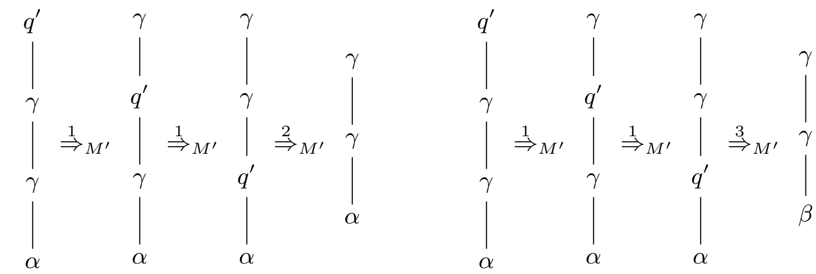

Fig. 2: Illustration of derivations of M' discussed in Example 3.

jecture, let us settle the relevant notion of composition first. A weighted relation $\tau\colon T_\Sigma \times T_\Delta \to S$ is *finitary* if the set $\{u \in T_\Delta \mid (t,u) \in \operatorname{supp}(\tau)\}$ is finite for every $t \in T_\Sigma$. The weighted relation τ_M computed by a wxtt M is always finitary by [6, Note before Lemma 2], which also applies to nonlinear wxtt. Now, let us first formally introduce the composition of 2 weighted relations. Let $\tau\colon T_\Sigma \times T_\Delta \to S$ and $\tau'\colon T_\Delta \times T_\Gamma \to S$ be weighted relations such that τ is finitary. Their composition $\tau \mathbin{;} \tau'\colon T_\Sigma \times T_\Gamma \to S$ is given by

$$(\tau \mathbin{;} \tau')(t, t'') = \sum_{t' \in T_\Delta} \tau(t, t') \cdot \tau'(t', t'')$$

for all $t \in T_\Sigma$ and $t'' \in T_\Gamma$, where the sum is finite since τ is finitary. Now we can state the conjecture. Given a constant wxtt M' and a linear wtdtt M, [17, Conjecture 11] claims that the composition $\tau_M \mathbin{;} \tau_{M'}$ can be computed by a wxtt.

Let us first provide some insight into the difficulties that arise when attempting the standard composition constructions. Let $M' = (Q', \Sigma, \Delta, Q'_0, I', \chi')$ be the constant wxtt and $M = (Q, \Delta, \Gamma, Q_0, I, \chi)$ be the linear wtdtt. Moreover, for every state $q' \in Q'$, let $c_{q'} \in S$ be such that q' is $c_{q'}$-constant. We first investigate the composition of M' and M using the generic composition construction of [17, Definition 6] to highlight the problem. Afterwards we provide a solution and prove that it is correct. We start with the exposition of the generic construction and an illustration of the inherent problem on a simplistic example.

Example 4. We reconsider the constant wtdtt $M' = (Q', \Sigma, \Delta, Q', I', \chi')$ of Example 3 together with the linear wtdtt $M = \big(\{q\},\ \Delta,\ \{\alpha^{(0)}\},\ \{q\},\ \{4\},\ \chi\big)$, where $\chi(4) = q\big(\gamma(x_1)\big) \xrightarrow{1} \alpha$. For the sake of illustration, we adjust M' such that $s_2 = s_3 = 2$; i.e., the weight of rules 2 and 3 is adjusted to weight 2. It can be verified as in Example 3 that q' is 4-constant in the adjusted wtdtt M'. Now, let us consider the input tree $t = \gamma\big(\gamma(\alpha)\big)$. As illustrated in Example 3 there are exactly 2 derivations in $D^{\perp}_{M'}\big(q'(t)\big)$, which are 112 and 113 as demonstrated in Fig. 2. Their weights are $s_1 \cdot s_1 \cdot s_2 = 1 \cdot 1 \cdot 2 = 2$ and $s_1 \cdot s_1 \cdot s_3 = 1 \cdot 1 \cdot 2 = 2$. There exists a single derivation $d_1 \in D^{\perp}_M\big(q(\gamma(\gamma(\alpha)))\big)$ on the output tree $\gamma\big(\gamma(\alpha)\big)$ of derivation 112 and a single derivation $d_2 \in D^{\perp}_M\big(q(\gamma(\gamma(\beta)))\big)$ on the output tree $\gamma\big(\gamma(\beta)\big)$ of derivation 113. In both cases the derivation is $d_1 = d_2 = 4$, and it is illustrated in Fig. 3. Its weight is obviously $s_4 = 1$.

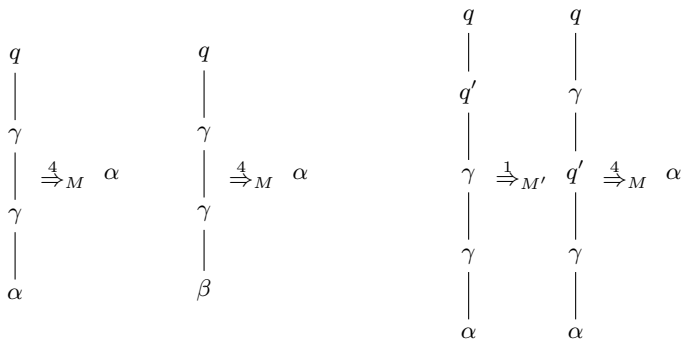

Fig. 3: Illustration of derivations discussed in Example 4.

The main idea of the generic composition construction is to apply rules of M immediately after the input symbol it consumes has been produced by a rule of M'. In this manner we avoid the explicit construction of the intermediate trees, which were $\gamma\big(\gamma(\alpha)\big)$ and $\gamma\big(\gamma(\beta)\big)$ in our example. Every rule i' of M' together with the rules of M that consume the symbols output by i' will form a new rule in the composed wxtt. In this manner we mix the rule applications of M' and M and obtain the derivation 14 starting at $q\big(q'(t)\big)$, which is illustrated right-most in Fig. 3. Note that since rule identifier 1 belongs to M', it will select the least occurrence of a state in the set $\{q'\}$ of states of M'. This derivation 14 has weight $s_1 \cdot s_4 = 1 \cdot 1 = 1$. It is evident that applications of rules 1, 2, and 3 are missing since the subtree $q'\big(\gamma(\alpha)\big)$ is deleted by the application of rule 4. This has no impact on the generated output tree, but the missing application of rules 1, 2, and 3 impacts the weight. Indeed the rule weights might even contain a factor 0 (directly or as a product of factors), which would potentially remove the pair of the input and output tree from the support of the computed weighted relation altogether.

This change of weight, which in our case always amounts to missing factors due to missing rule applications, needs to be corrected in an adjusted composition construction. We also remark that in the derivation presented the subtree $\gamma(\alpha)$ in the input tree was never processed and could potentially been processed utilizing different rules. As presented in the example, we could utilize derivation 12 or 13 to process the remaining subtree $q'\big(\gamma(\alpha)\big)$. The newly constructed rule needs to account for all derivations that would have been possible starting in $q'\big(\gamma(\alpha)\big)$, which have total weight $c_{q'}$ by the definition of $c_{q'}$-constant. Moreover, due to the definition of $c_{q'}$-constant, we know that $c_{q'}$ is the total weight irrespective of the actual subtree that would need to still be processed.

The solution is relatively straightforward. Whenever M deletes a subtree (i.e., a variable $x \in X$ occurs in the left-hand side of a rule, but does not occur in the corresponding right-hand side), we charge the weight $c_{q'}$ for every occurrence of state q' in the deleted part. Since the wtdtt M is linear, it will not copy

subtrees and thus not duplicate occurrences of states $q' \in Q'$. This motivates the following composition of the given wxtt.

Definition 5 (see [17, Definition 6]). *Let $M' = (Q', \Sigma, \Delta, Q'_0, I', \chi')$ be a constant wxtt and $M = (Q, \Delta, \Gamma, Q_0, I, \chi)$ be a linear wtdtt. Moreover, for every $q' \in Q'$, let $c_{q'} \in S$ be such that q' is $c_{q'}$-constant. Finally, let $b \in \mathbb{N}$ be such that $|\mathrm{pos}(r_{i'})| \leq b$ for every $i' \in I'$. The composed wxtt $M' ; M$ is the wxtt $(Q \times Q', \Sigma, \Gamma, Q_0 \times Q'_0, P, \rho)$ such that $P = I' \times Q \times I^{\leq b}$ and*

$$\rho\big(\langle i', q, d\rangle\big) = \begin{cases} \big(q(\ell_{i'}),\, s,\, r\big) & \text{if } r = d\big(q(r_{i'})\big) \in T_\Delta\Big(Q\big(Q'(X)\big)\Big) \\ 0 & \text{otherwise} \end{cases}$$

for every identifier $i' \in I'$, state $q \in Q$, and identifier sequence $d \in I^{\leq b}$, where

$$s = s_{i'} \cdot \mathrm{wt}_M(d) \cdot \prod_{q' \in Q'} c_{q'}^{|\mathrm{pos}_{q'}(r_{i'})| - |\mathrm{pos}_{q'}(r)|} \quad .$$

Note that we identify $Q\big(Q'(T)\big)$ with $(Q \times Q')(T)$.

For the given example derivation in Fig. 3 we observe that the application of rule 4 deletes a subtree containing 1 occurrence of state q', for which $c_{q'} = 4$. Hence the newly constructed rule with identifier $\langle 1, q, 4\rangle$ would be assigned weight $s_1 \cdot s_4 \cdot c_{q'}$. Let us state the main result, which confirms [17, Conjecture 11].

Theorem 6. *Let M' be a constant wxtt and M be a linear wtdtt. Then there exists a wxtt N such that $\tau_N = \tau_{M'} ; \tau_M$.*

Proof (sketch). Let $M' = (Q', \Sigma, \Delta, Q'_0, I', \chi')$. Without loss of generality, we can assume that for every $i' \in I'$ there exists $k_{i'} \in \mathbb{N}$ such that $\mathrm{var}(\ell_{i'}) = X_{k_{i'}}$. We call a tree $c \in C_\Sigma(X_k)$ *normalized* if the leafs labeled $x_1, \ldots, x_k$ occur in exactly in this order when read left-to-right. Following [19, Definition 3] we define the mapping $\mathrm{h}_{M'} \colon Q'(T_\Sigma) \times T_\Delta \to S$ for every $\xi \in Q'(T_\Sigma)$ and $u \in T_\Delta$ by

$$\mathrm{h}_{M'}(\xi, u) = \sum_{\substack{i' \in I' \\ (l_{i'}, \theta') \in \mathrm{match}(\xi) \\ (c, \theta'') \in \mathrm{match}(u) \\ c \text{ normalized} \\ \theta \colon \mathrm{var}(c) \to Q'(X_{k_{i'}}) \\ r_{i'} = c\theta}} s_{i'} \cdot \prod_{x \in \mathrm{var}(c)} \mathrm{h}_{M'}(x\theta\theta', x\theta'') \quad .$$

The proof of [19, Theorem 5] shows that $\mathrm{h}_{M'}(\xi, u) = \tau'_{M'}(\xi, u)$ for all $\xi \in Q'(T_\Sigma)$ and $u \in T_\Delta$. The same alternative semantics also applies to M and N naturally. The following statement can be proven for every $t \in T_\Sigma$ and $t'' \in T_\Gamma$ by induction on t

$$\mathrm{h}_N\big(\langle q, q'\rangle(t), t''\big) = \sum_{t' \in T_\Delta} \mathrm{h}_{M'}\big(q'(t), t'\big) \cdot \mathrm{h}_M\big(q(t'), t''\big) \quad .$$

This directly yields

$$\begin{aligned}
&\tau_N(t, t'') \\
&= \sum_{\langle q,q'\rangle \in Q_0 \times Q'_0} \tau'_N\big(\langle q, q'\rangle(t), t''\big) = \sum_{\substack{\langle q,q'\rangle \in Q_0 \times Q'_0 \\ t' \in T_\Delta}} \tau'_{M'}\big(q'(t), t'\big) \cdot \tau'_M\big(q(t'), t''\big) \\
&= \sum_{t' \in T_\Delta} \Big(\sum_{q' \in Q'_0} \tau'_{M'}\big(q'(t), t'\big) \Big) \cdot \Big(\sum_{q \in Q_0} \tau'_M\big(q(t'), t''\big) \Big) = \sum_{t' \in T_\Delta} \tau_{M'}(t, t') \cdot \tau_M(t', t'') \\
&= (\tau_{M'} \,;\, \tau_M)(t, t'')
\end{aligned}$$

and thus the desired $\tau_N = \tau_{M'} \,;\, \tau_M$. An alternative argumentation based on the original derivation semantics is provided in the appendix. □

5 Decidability of Constant Property

To make Theorem 6 effective, we would like to decide whether a given state is c-constant for a given $c \in S$ and, more generally, whether a given wxtt is constant. In the following let $M = (Q, \Sigma, \Delta, Q_0, I, \chi)$ be a wxtt, for which we want to check whether it is constant. Some straightforward results are already mentioned in [17, Example 10] for wtdtt, which we generalize easily to wxtt here.

Lemma 7 (see [17, Example 10]). *Every state $q \in Q$ is 1-constant if*

1. *M is* Boolean *and total and the semiring S is idempotent, or*
2. *M is* Boolean*, total, and unambiguous.*

Clearly, totality of M is necessary for every state $q \in Q$ to be 1-constant because otherwise there exists a state $q \in Q$ and input tree $t \in T_\Sigma$ such that $\tau'_M\big(q(t), u\big) = 0$ for all output trees $u \in T_\Delta$. Consequently, for that q and t the sum in the definition of 1-constant (see Definition 2) is 0, which shows that q is not 1-constant. Already [17] mentions that totality is not sufficient. Next, we demonstrate that none of the properties mentioned in Lemma 7 besides totality are necessary for every state $q \in Q$ to be 1-constant.

Example 8. Recall the wxtt M' over the semiring $(\mathbb{R}, +, \cdot, 0, 1)$ of Example 3. We already observed in Example 3 that q' is 1-constant, but M' is neither unambiguous nor Boolean. In addition, the semiring $(\mathbb{R}, +, \cdot, 0, 1)$ is clearly not idempotent.

Next we investigate some special cases, for which we can decide the constant property. We start with the Boolean semiring as a starting point. Clearly, the Boolean semiring $(\{0, 1\}, \max, \min, 0, 1)$ is idempotent and all wxtt over the Boolean semiring are trivially Boolean, so according to Lemma 7 we only need to check totality. We present a slightly more general statement, which states decidability of totality for all Boolean wxtt over an idempotent semiring.

Lemma 9. *Totality of* Boolean *wxtt over an idempotent semiring is decidable.*

Proof. Clearly, rules with weight 0 are useless, so without loss of generality, let $M = (Q, \Sigma, \Delta, Q_0, I, \chi)$ be a BOOLEAN wxtt such that $s_i = 1$ for all $i \in I$; i.e., all rules have weight 1. For every state $q \in Q$ we also consider the variant $M_q = (Q, \Sigma, \Delta, \{q\}, I, \chi)$, which is essentially M but with the single initial state q. Clearly, the subsemiring of S generated by $\{0, 1\}$, which are the only weights permitted in M (as well as M_q for all $q \in Q$), is isomorphic to the BOOLEAN semiring $(\{0, 1\}, \max, \min, 0, 1)$ because S is idempotent. Let $q \in Q$ be an arbitrary state. Hence M_q is essentially a wxtt over the BOOLEAN semiring, and thus also an (unweighted) extended tree transducer of [20]. By [20, Theorem 4.8] such a transducer can equivalently be presented as a top-down tree transducer N_q with regular look-ahead [4]. It is well-known [4, Corollary 2.7] that the domain $\mathrm{dom}(\tau_{N_q})$ of the tree transformation $\tau_{N_q} \subseteq T_\Sigma \times T_\Delta$ computed by N_q is a recognizable tree language [9,10].

Obviously, M is total if and only if $\mathrm{dom}(\tau_{N_q}) = T_\Sigma$ for every $q \in Q$. Since $\mathrm{dom}(\tau_{N_q})$ is recognizable and universality is decidable [9,10] for recognizable tree languages, totality is also decidable.

A closer analysis of the proof shows that a (semiring) homomorphism [14,11] from the subsemiring S' of S generated by the weights in M into the BOOLEAN semiring would be sufficient. Such a homomorphism exists if S' is not a ring by [26, Theorem 2.1]. Additionally, the restrictions on the addition can be avoided, if the wxtt is restricted such that multiple derivations for the same state, input and output tree are impossible. Using similar techniques we can thus also show that totality is decidable for

- BOOLEAN wxtt over zero-sum free semirings,
- wxtt over zero-sum and zero-divisor free semirings,
- BOOLEAN unambiguous wxtt, and
- unambiguous wxtt over zero-divisor free semirings.

Thus the conditions of the first item of Lemma 7 can effectively be checked in an idempotent semiring. It is beyond the scope of this contribution to develop general decidability results for unambiguity. However, a simpler condition exists for wtdtt. Suppose that M is a wtdtt. It is called *deterministic* if for every state $q \in Q$ and $\sigma \in \Sigma$ there exists at most one rule identifier $i \in I$ such that $\ell_i(\varepsilon) = q$ and $\ell_i(1) = \sigma$. Every deterministic wtdtt is guaranteed to be unambiguous.

Let us provide another relevant scenario. Let M be a deterministic wtdtt without useless rules, which can efficiently be checked, and no rules of weight 0 (i.e., $s_i \neq 0$ for all $i \in I$). Moreover, suppose that S is multiplicatively cancellative (i.e., $s \cdot s' = s \cdot s''$ implies $s' = s''$ for all $s, s', s'' \in S$ with $s \neq 0$). For example, every field S is multiplicatively cancellative. Then S is zero-divisor free and by the last item of the previous list, totality is decidable. In fact, due to the special shape of left-hand sides of an wtdtt it suffices to check whether for every state $q \in Q$ and $\sigma \in \Sigma$ there exists a rule identifier $i \in I$ with $\ell_i(\varepsilon) = q$ and $\ell_i(1) = \sigma$. Together with determinism there is thus exactly one rule identifier $i \in I$ with $\ell_i(\varepsilon) = q$ and $\ell_i(1) = \sigma$ for every state $q \in Q$ and $\sigma \in \Sigma$.

Due to the cancellation property of S we can now utilize the weight pushing strategy [22,13] to determine whether a given state $q \in Q$ is c-constant for some $c \in S$. The standard pushing strategy cannot be applied directly since M might copy or delete. We consider it interesting to extend the existing pushing strategies to these scenarios where (i) weights might not be applied due to deletion or (ii) weights might be applied multiple times due to copying. We leave the details of this adaptation to future work, but believe that this avenue allows an efficient test of the constant property in this relevant scenario (without requiring the wtdtt to be BOOLEAN).

References

1. Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., Mohri, M.: OpenFst — a general and efficient weighted finite-state transducer library. In: Proc. 12th Int. Conference Implementation and Application of Automata. Lecture Notes in Computer Science, vol. 4783, pp. 11–23. Springer (2007). https://doi.org/10.1007/978-3-540-76336-9_3
2. Arnold, A., Dauchet, M.: Morphismes et bimorphismes d'arbres. Theoretical Computer Science **20**(4), 33–93 (1982). https://doi.org/10.1016/0304-3975(82)90098-6
3. Dauchet, M.: Transductions inversibles de forêts. Thèse 3ème cycle, Université de Lille (1975), http://ori.univ-lille1.fr/notice/view/univ-lille1-ori-70098
4. Engelfriet, J.: Top-down tree transducers with regular look-ahead. Mathematical Systems Theory **10**, 289–303 (1977). https://doi.org/10.1007/BF01683280
5. Engelfriet, J., Fülöp, Z., Vogler, H.: Bottom-up and top-down tree series transformations. Journal of Automata, Languages and Combinatorics **7**(1), 11–70 (2002). https://doi.org/10.25596/jalc-2002-011
6. Fülöp, Z., Maletti, A.: Composition closure of linear weighted extended top-down tree transducers. In: Proc. 24th Int. Conf. Implementation and Application of Automata. Lecture Notes in Computer Science, vol. 11601, pp. 133–145. Springer (2019). https://doi.org/10.1007/978-3-030-23679-3_11
7. Fülöp, Z., Vogler, H.: Tree series transformations that respect copying. Theory of Computing Systems **36**(3), 247–293 (2003). https://doi.org/10.1007/s00224-003-1072-z
8. 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). https://doi.org/10.1007/978-3-642-01492-5_9
9. Gécseg, F., Steinby, M.: Tree Automata. Akadémiai Kiadó, Budapest (1984), 2nd revision available at https://arxiv.org/abs/1509.06233
10. Gécseg, F., Steinby, M.: Tree languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. 3, chap. 1, pp. 1–68. Springer (1997). https://doi.org/10.1007/978-3-642-59126-6_1
11. Golan, J.S.: Semirings and their Applications. Kluwer Academic Publishers (1999). https://doi.org/10.1007/978-94-015-9333-5
12. Graehl, J., Knight, K., May, J.: Training tree transducers. Computational Linguistics **34**(3), 391–427 (2008). https://doi.org/10.1162/coli.2008.07-051-R2-03-57
13. Hanneforth, T., Maletti, A., Quernheim, D.: Pushing for weighted tree automata. Logical Methods in Computer Science **14**(1:5), 1–16 (2018). https://doi.org/10.23638/LMCS-14(1:5)2018

14. Hebisch, U., Weinert, H.J.: Semirings — Algebraic Theory and Applications in Computer Science. World Scientific Publishing (1998). https://doi.org/10.1142/3903
15. Koehn, P.: Statistical Machine Translation. Cambridge University Press (2010). https://doi.org/10.1017/CBO9780511815829
16. Kuich, W.: Tree transducers and formal tree series. Acta Cybernetica **14**(1), 135–149 (1999), https://cyber.bibl.u-szeged.hu/index.php/actcybern/article/view/3516
17. Lagoutte, A., Maletti, A.: Survey: Weighted extended top-down tree transducers — part III: Composition. In: Kuich, W., Rahonis, G. (eds.) Proc. Workshop Algebraic Foundations in Computer Science. Lecture Notes in Computer Science, vol. 7020, pp. 272–308. Springer (2011). https://doi.org/10.1007/978-3-642-24897-9_13
18. Maletti, A.: Compositions of extended top-down tree transducers. Information and Computation **206**(9–10), 1187–1196 (2008). https://doi.org/10.1016/j.ic.2008.03.019
19. Maletti, A.: *Survey:* weighted extended top-down tree transducers — part I: Basics and expressive power. Acta Cybernetica **20**(2), 223–250 (2011). https://doi.org/10.14232/actacyb.20.2.2011.2
20. Maletti, A., Graehl, J., Hopkins, M., Knight, K.: The power of extended top-down tree transducers. SIAM Journal on Computing **39**(2), 410–430 (2009). https://doi.org/10.1137/070699160
21. May, J., Knight, K., Vogler, H.: Efficient inference through cascades of weighted tree transducers. In: Proc. 48th Ann. Meeting Association for Computational Linguistics. pp. 1058–1066. Association for Computational Linguistics (2010), https://www.aclweb.org/anthology/P10-1108
22. Mohri, M., Pereira, F., Riley, M.: Weighted finite-state transducers in speech recognition. Computer Speech & Language **16**(1), 69–88 (2002). https://doi.org/10.1006/csla.2001.0184
23. Rounds, W.C.: Mappings and grammars on trees. Mathematical Systems Theory **4**(3), 257–287 (1970). https://doi.org/10.1007/BF01695769
24. Thatcher, J.W.: Generalized^2 sequential machine maps. Journal of Computer and System Sciences **4**(4), 339–367 (1970). https://doi.org/10.1016/S0022-0000(70)80017-4
25. Thatcher, J.W.: Tree automata: An informal survey. In: Aho, A.V. (ed.) Currents in the Theory of Computing, chap. 4, pp. 143–172. Prentice Hall (1973)
26. Wang, H.: On characters of semirings. Houston Journal of Mathematics **23**(3), 391–405 (1997), https://www.math.uh.edu/~hjm/restricted/archive/v023n3/0391WANG.pdf
27. Yamada, K., Knight, K.: A decoder for syntax-based statistical MT. In: Proc. 40th Ann. Meeting Association for Computational Linguistics. pp. 303–310. Association for Computational Linguistics (2002). https://doi.org/10.3115/1073083.1073134