Finite Sequentiality of Unambiguous Max-Plus Tree Automata

Erik Paul
Institute of Computer Science, Leipzig University, 04109 Leipzig, Germany
epaul@informatik.uni-leipzig.de

Abstract

We show the decidability of the finite sequentiality problem for unambiguous max-plus tree automata. A max-plus tree automaton is called unambiguous if there is at most one accepting run on every tree. The finite sequentiality problem asks whether for a given max-plus tree automaton, there exist finitely many deterministic max-plus tree automata whose pointwise maximum is equivalent to the given automaton.

2012 ACM Subject Classification Theory of computation → Quantitative automata; Theory of computation → Tree languages

Keywords and phrases Weighted Tree Automata, Max-Plus Tree Automata, Finite Sequentiality, Decidability, Ambiguity

Digital Object Identifier 10.4230/LIPIcs.STACS.2019.55

Funding This work was supported by Deutsche Forschungsgemeinschaft (DFG), Graduiertenkolleg 1763 (QuantLA).

1 Introduction

A max-plus automaton is a finite automaton which assigns real numbers to words over a given alphabet. The transitions of a max-plus automaton each carry a weight from the real numbers. To every run of the automaton, a weight is associated by summing over the weights of the transitions which constitute the run. The weight of a word is given by the maximum over the weights of all runs on this word.

More generally, max-plus automata and their min-plus counterparts are weighted automata [31, 30, 22, 5, 11] over the max-plus or min-plus semiring. Min-plus automata were originally introduced by Imre Simon as a means to show the decidability of the *finite power property* [34, 35]. Since their introduction, max-plus and min-plus automata enjoy a continuing interest [21, 14, 18, 10, 12, 6] and they have been employed in many different contexts. To only name some examples, they can be used to determine the star height of a language [13], to prove the termination of some string rewriting systems [36], and to model certain discrete event systems [19]. Additionally, they appear in the context of natural language processing [24], where for reasons of numerical stability, probabilities are often computed in the min-plus semiring as negative log-likelihoods.

A very prominent open question about max-plus automata is the *sequentiality problem*, the problem of deciding whether for an arbitrary max-plus automaton there exists a deterministic equivalent. A max-plus automaton is called *deterministic* or *sequential* if for each pair of a state and an input symbol, there is at most one valid transition into a next state. Although the decidability of this problem is unknown for max-plus automata in general, it is known to be decidable for the subclasses of *unambiguous* [24], *finitely ambiguous* [18], and even *polynomially ambiguous* [17] automata. A max-plus automaton is called *unambiguous* if there exists at most one accepting run on every word. It is called *finitely ambiguous* if the number of runs on each word is bounded by a global constant. If on every word the number of accepting runs is bounded polynomially in the length of the word, the automaton is said to

be *polynomially ambiguous*. Note that the ambiguity of a max-plus automaton is a decidable property, as it is easily reduced to deciding the ambiguity of a finite automaton. Deciding the sequentiality of a finite automaton is trivial, polynomial time algorithms for deciding the unambiguity, the finite ambiguity, and the polynomial ambiguity of a finite automaton can be found in [7, 37, 33]. Furthermore, the classes of functions definable by deterministic, unambiguous, finitely ambiguous, polynomially ambiguous, and arbitrary max-plus automata form a strictly ascending hierarchy [18, 15, 23].

A decidability problem which is closely related to the sequentiality problem is the *finite sequentiality problem*. The finite sequentiality problem asks whether a given max-plus automaton can be represented as a pointwise maximum of deterministic max-plus automata. In [14], it was left as an open question to determine the decidability of the finite sequentiality problem for finitely ambiguous max-plus automata. It was shown only recently that for the classes of unambiguous as well as finitely ambiguous automata, the finite sequentiality problem is decidable [3, 2]. The class of functions which allow a finitely sequential representation by max-plus automata lies strictly between the classes of functions definable by deterministic and by finitely ambiguous max-plus automata, and it is incomparable to the class of functions definable by unambiguous max-plus automata [18].

In this paper, we show that the finite sequentiality problem is decidable for unambiguous max-plus tree automata. Max-plus tree automata are a generalization of max-plus automata and operate on trees instead of words. Applications for max-plus tree automata include proving the termination of certain term rewriting systems [20], and they are also commonly employed in natural language processing [27] in the form of *probabilistic context-free grammars*. Our approach to show the decidability of the finite sequentiality problem employs ideas from [3]. In [3], the *fork property* is shown to be a decidable criterion to determine the existence of a finitely sequential equivalent. More precisely, a max-plus word automaton is shown to possess a finitely sequential representation if and only if it does not satisfy the fork property. It is shown elementarily that an automaton satisfying the fork property cannot possess a finitely sequential equivalent. The proof for the existence of a finitely sequential representation in case that the fork property is not satisfied, on the other hand, relies on the construction of finitely many unambiguous max-plus automata whose pointwise maximum is equivalent to the original automaton, and which all satisfy the *twins property*. It was shown by Mohri [24] that an unambiguous max-plus automaton which satisfies the twins property is determinizable. A finitely sequential representation is thus found by determinizing the unambiguous automata.

For tree automata, we generalize the fork property to the *tree fork property* by adding a condition which accounts for the nonlinear structure of trees. We then prove that an unambiguous max-plus tree automaton possesses a finitely sequential representation if and only if it does not satisfy the tree fork property. As in the word case, the most challenging part of the proof is to show the existence of a finitely sequential representation whenever the tree fork property is not satisfied. Like in the proof for word automata, we construct finitely many unambiguous max-plus tree automata which possess a deterministic equivalent. However, we need to take a different approach in order to obtain these automata. In [3], a modified *Schützenberger covering* [32, 29] is first constructed from the unambiguous max-plus automaton, from which in turn an automaton is constructed which monitors the occurrence of certain states of the modified Schützenberger covering. This latter automaton is then decomposed into the finitely many unambiguous automata. This approach, however, is not applicable to trees, as the monitoring of states requires all relevant states to occur linearly. This happens trivially for word automata due to the inherent linear structure of words, but for

tree automata examples can be found where relevant states occur nonlinearly. The approach we use here relies on constructing a max-plus automaton which tracks certain pairs of states of the original automaton. When applied to word automata, this immediately yields an automaton which can be decomposed into the desired unambiguous automata. Unfortunately, for tree automata this tracking of pairs of states again fails due to states occurring nonlinearly. Surprisingly however, our construction can be applied to the Schützenberger covering of the original tree automaton, as the states relevant for tracking all occur pairwise linearly in the Schützenberger covering. The most difficult part of our proof is to show that the Schützenberger covering indeed has the property we just indicated.

2 Preliminaries

For a set X, we denote the power set of X by $\mathcal{P}(X)$ and the cardinality of X by $|X|$. For two sets X and Y and a mapping $f\colon X \to Y$, we call X the *domain* of f, denoted by $\mathrm{dom}(f)$, and Y the *range* of f, denoted by $\mathrm{range}(f)$. For a subset $X' \subseteq X$, we call the set $f(X') = \{y \in Y \mid \exists x \in X'\colon f(x) = y\}$ the *image* or *range of X' under f*. For an element $y \in Y$, we call the set $f^{-1}(y) = \{x \in X \mid f(x) = y\}$ the *preimage of y under f*. For a second mapping $g\colon X \to Y$, we write $f = g$ if for all $x \in X$ we have $f(x) = g(x)$.

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$. By $\mathbb{N}^*$ we denote the set of all finite words over $\mathbb{N}$. The empty word is denoted by ε, and the length of a word $w \in \mathbb{N}^*$ by $|w|$. The set $\mathbb{N}^*$ is partially ordered by the prefix relation $\leq_p$ and totally ordered with respect to the lexicographic ordering $\leq_l$. Two words from $\mathbb{N}^*$ are called *prefix-dependent* if they are in prefix relation, and otherwise they are called *prefix-independent*.

A *ranked alphabet* is a pair $(\Gamma, \mathrm{rk}_\Gamma)$, often abbreviated by Γ, where Γ is a finite set and $\mathrm{rk}_\Gamma\colon \Gamma \to \mathbb{N}$ a mapping which assigns a rank to every symbol. For every $m \geq 0$ we define $\Gamma^{(m)} = \mathrm{rk}_\Gamma^{-1}(m)$ as the set of all symbols of rank m. The rank of Γ is defined as $\mathrm{rk}(\Gamma) = \max\{\mathrm{rk}_\Gamma(a) \mid a \in \Gamma\}$. The set of (*finite, labeled, and ordered*) Γ-*trees*, denoted by T_Γ, is the set of all pairs $t = (\mathrm{pos}(t), \mathrm{label}_t)$, where $\mathrm{pos}(t) \subset \mathbb{N}^*$ is a finite non-empty prefix-closed set of *positions*, $\mathrm{label}_t\colon \mathrm{pos}(t) \to \Gamma$ is a mapping, and for every $w \in \mathrm{pos}(t)$ we have $wi \in \mathrm{pos}(t)$ iff $1 \leq i \leq \mathrm{rk}_\Gamma(\mathrm{label}_t(w))$. We write $t(w)$ for $\mathrm{label}_t(w)$ and $|t|$ for $|\mathrm{pos}(t)|$. We also refer to the elements of $\mathrm{pos}(t)$ as *nodes*, to ε as the *root* of t, and to prefix-maximal nodes as *leaves*. The *height* of t is defined as $\mathrm{height}(t) = \max_{w \in \mathrm{pos}(t)} |w|$. For a leaf $w \in \mathrm{pos}(t)$, the set $\{v \in \mathrm{pos}(t) \mid v \leq_p w\}$ is called a *branch* of t.

Now let $s, t \in T_\Gamma$ and $w \in \mathrm{pos}(t)$. The *subtree of t at w*, denoted by $t{\restriction_w}$, is a Γ-tree defined as follows. We let $\mathrm{pos}(t{\restriction_w}) = \{v \in \mathbb{N}^* \mid wv \in \mathrm{pos}(t)\}$ and for $v \in \mathrm{pos}(t{\restriction_w})$, we let $\mathrm{label}_{t{\restriction_w}}(v) = t(wv)$. The *substitution of s into w of t*, denoted by $t\langle s \to w\rangle$, is a Γ-tree defined as follows. We let $\mathrm{pos}(t\langle s \to w\rangle) = \{v \in \mathrm{pos}(t) \mid w \not\leq_p v\} \cup \{wv \mid v \in \mathrm{pos}(s)\}$. For $v \in \mathrm{pos}(t\langle s \to w\rangle)$, we let $\mathrm{label}_{t\langle s \to w\rangle}(v) = s(u)$ if $v = wu$ for some $u \in \mathrm{pos}(s)$, and otherwise $\mathrm{label}_{t\langle s \to w\rangle}(v) = t(v)$.

For $a \in \Gamma^{(m)}$ and trees $t_1, \ldots, t_m \in T_\Gamma$, we also write $a(t_1, \ldots, t_m)$ to denote the tree t with $\mathrm{pos}(t) = \{\varepsilon\} \cup \{iw \mid i \in \{1, \ldots, m\}, w \in \mathrm{pos}(t_i)\}$, $\mathrm{label}_t(\varepsilon) = a$, and $\mathrm{label}_t(iw) = t_i(w)$.

For a ranked alphabet Γ, a tree over the alphabet $\Gamma_\diamond = (\Gamma \cup \{\diamond\}, \mathrm{rk}_\Gamma \cup \{\diamond \mapsto 0\})$ is called a Γ-*context*. Let $t \in T_{\Gamma_\diamond}$ be a Γ-context and let $w_1, \ldots, w_n \in \mathrm{pos}(t)$ be a lexicographically ordered enumeration of all leaves of t labeled $\diamond$. Then we call t an n-Γ-*context* and define $\Diamond_i(t) = w_i$ for $i \in \{1, \ldots, n\}$. For an n-Γ-context t and contexts $t_1, \ldots, t_n \in T_{\Gamma_\diamond}$, we define $t(t_1, \ldots, t_n) = t\langle t_1 \to \Diamond_1(t)\rangle \ldots \langle t_n \to \Diamond_n(t)\rangle$ by substitution of $t_1, \ldots, t_n$ into the $\diamond$-leaves of t. A 1-Γ-context is also called a Γ-*word*. For a Γ-word s, we define $s^1 = s$ and $s^{n+1} = s(s^n)$ for $n \geq 1$.

A *commutative semiring* is a tuple $(K, \oplus, \odot, \mathbb{0}, \mathbb{1})$, abbreviated by K, with operations sum $\oplus$ and product $\odot$ and constants $\mathbb{0}$ and $\mathbb{1}$ such that $(K, \oplus, \mathbb{0})$ and $(K, \odot, \mathbb{1})$ are commutative monoids, multiplication distributes over addition, and $\kappa \odot \mathbb{0} = \mathbb{0} \odot \kappa = \mathbb{0}$ for every $\kappa \in K$. In this paper, we only consider the *max-plus semiring* $\mathbb{R}_{\max} = (\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$ where the sum and the product operations are max and $+$, respectively, extended to $\mathbb{R} \cup \{-\infty\}$ in the usual way.

A *max-plus weighted bottom-up finite state tree automaton (short: max-plus-WTA) over* Γ is a tuple $\mathcal{A} = (Q, \Gamma, \mu, \nu)$ where Q is a finite set (of states), Γ is a ranked alphabet (of input symbols), $\mu\colon \bigcup_{m=0}^{\mathrm{rk}(\Gamma)} Q^m \times \Gamma^{(m)} \times Q \to \mathbb{R}_{\max}$ (the function of transition weights), and $\nu\colon Q \to \mathbb{R}_{\max}$ (the function of final weights). We define $\Delta_\mathcal{A} = \mathrm{dom}(\mu)$. A tuple $(\bar{p}, a, q) \in \Delta_\mathcal{A}$ is called a *transition* and $(\bar{p}, a, q)$ is called *valid* if $\mu(\bar{p}, a, q) \neq -\infty$. A state $q \in Q$ is called *final* if $\nu(q) \neq -\infty$.

For a tree $t \in T_\Gamma$, a mapping $r\colon \mathrm{pos}(t) \to Q$ is called a *quasi-run of* $\mathcal{A}$ *on* t. For a quasi-run r on t and a position $w \in \mathrm{pos}(t)$ with $t(w) = a \in \Gamma^{(m)}$, the tuple $\mathbb{t}(t, r, w) = (r(w1), \ldots, r(wm), a, r(w))$ is called the *transition at* w. The quasi-run r is called a *(valid) run* if for every $w \in \mathrm{pos}(t)$ the transition $\mathbb{t}(t, r, w)$ is valid with respect to $\mathcal{A}$. We call a run r *accepting* if $r(\varepsilon)$ is final. By $\mathrm{Run}_\mathcal{A}(t)$ and $\mathrm{Acc}_\mathcal{A}(t)$ we denote the sets of all runs and all accepting runs of $\mathcal{A}$ on t, respectively. Similar to trees, we define restrictions of runs as follows. Let $t \in T_\Gamma$, $r \in \mathrm{Run}_\mathcal{A}(t)$, and $w \in \mathrm{pos}(t)$. We define $r\restriction_w \in \mathrm{Run}_\mathcal{A}(t\restriction_w)$ by $r\restriction_w(v) = r(wv)$ for $v \in \mathrm{pos}(t\restriction_w)$.

For $r \in \mathrm{Run}_\mathcal{A}(t)$, the *weight of* r is defined by $\mathrm{wt}_\mathcal{A}(t, r) = \sum_{w \in \mathrm{pos}(t)} \mu(\mathbb{t}(t, r, w))$. The *behavior of* $\mathcal{A}$, denoted by $[\![\mathcal{A}]\!]$, is the mapping defined for every $t \in T_\Gamma$ by $[\![\mathcal{A}]\!](t) = \max_{r \in \mathrm{Acc}_\mathcal{A}(t)}(\mathrm{wt}_\mathcal{A}(t, r) + \nu(r(\varepsilon)))$, where the maximum over the empty set is $-\infty$ by convention.

For a max-plus-WTA $\mathcal{A} = (Q, \Gamma, \mu, \nu)$, a run of $\mathcal{A}$ on a Γ-context t is a run of the max-plus-WTA $\mathcal{A}' = (Q, \Gamma_\diamond, \mu', \nu)$ on t, where $\mu'(\diamond, q) = 0$ for all $q \in Q$ and $\mu'(d) = \mu(d)$ for $d \in \Delta_\mathcal{A}$. We denote $\mathrm{Run}^\diamond_\mathcal{A}(t) = \mathrm{Run}_{\mathcal{A}'}(t)$ and for $r \in \mathrm{Run}^\diamond_\mathcal{A}(t)$ write $\mathrm{wt}^\diamond_\mathcal{A}(t, r) = \mathrm{wt}_{\mathcal{A}'}(t, r)$. For a Γ-word s, we write $p \xrightarrow{s|x} q$ if there exists a run $r \in \mathrm{Run}^\diamond_\mathcal{A}(s)$ with $r(\Diamond_1(s)) = p$, $r(\varepsilon) = q$, and $\mathrm{wt}^\diamond_\mathcal{A}(s, r) = x$. In this case, r is said to *realize* $p \xrightarrow{s|x} q$. Note that $r \in \mathrm{Run}^\diamond_\mathcal{A}(s)$ implies $x \neq -\infty$.

For a max-plus-WTA $\mathcal{A}$, we define a relation $\leq$ on Q by $q \leq p$ iff $p \xrightarrow{s|x} q$ for some Γ-word $s \in T_{\Gamma_\diamond}$. We call $\mathcal{A}$ *trim* if for every $p \in Q$ there exists $t \in T_\Gamma$, $r \in \mathrm{Acc}(t)$, and $w \in \mathrm{pos}(t)$ with $r(w) = p$. The *trim part of* $\mathcal{A}$ is the automaton obtained by removing all states $p \in Q$ for which no such t, r, and w exist. This process obviously has no influence on $[\![\mathcal{A}]\!]$.

A max-plus-WTA $\mathcal{A}$ is called *deterministic* or *sequential* if for every $m \geq 0$, $a \in \Gamma^{(m)}$, and $\bar{p} \in Q^m$, there exists at most one $q \in Q$ with $\mu(\bar{p}, a, q) \neq -\infty$. We call $\mathcal{A}$ *unambiguous* if $|\mathrm{Acc}_\mathcal{A}(t)| \leq 1$ for every $t \in T_\Gamma$. We call the behavior $[\![\mathcal{A}]\!]$ of $\mathcal{A}$ *finitely sequential* if there exist deterministic max-plus-WTA $\mathcal{A}_1, \ldots, \mathcal{A}_n$ over Γ with $[\![\mathcal{A}]\!] = \max_{i=1}^n [\![\mathcal{A}_i]\!]$, where the maximum is taken pointwise.

3 Main Result

We will show that for an unambiguous max-plus-WTA $\mathcal{A}$, it is decidable whether its behavior $[\![\mathcal{A}]\!]$ is finitely sequential. Moreover, if it is finitely sequential, we will obtain that the deterministic max-plus-WTA $\mathcal{A}_1, \ldots, \mathcal{A}_n$ can be effectively constructed. For this, we follow ideas from [3], where the decidability of the finite sequentiality problem was proved for unambiguous max-plus word automata. The general outline of our proof is similar to that of [3] and presents itself as follows. We introduce the *tree fork property* and show that it

is decidable whether an unambiguous max-plus-WTA $\mathcal{A}$ satisfies this property. Then we show that the behavior of an unambiguous max-plus-WTA is finitely sequential if and only if it does not satisfy the tree fork property. In conclusion, we obtain the decidability of the finite sequentiality problem for unambiguous max-plus-WTA. Elementary proof methods can be used to show that $[\![\mathcal{A}]\!]$ is not finitely sequential if $\mathcal{A}$ satisfies the tree fork property. On the other hand, if $\mathcal{A}$ does not satisfy the tree fork property, we show how to construct finitely many unambiguous max-plus-WTA whose pointwise maximum is $[\![\mathcal{A}]\!]$, and which all satisfy the *twins property* [24]. Every unambiguous max-plus-WTA which satisfies the twins property possesses an effectively constructable deterministic equivalent [9]. Thus, we obtain finitely many deterministic max-plus-WTA whose pointwise maximum is $[\![\mathcal{A}]\!]$, which is hence finitely sequential.

In the following, we recall the twins property and introduce the tree fork property. Let Γ be a ranked alphabet. We begin with the concepts of *siblings* and *twins*. Intuitively, two states are called *siblings* if they can be "reached" by the same tree. Two siblings are called *twins* if for every Γ-word which can "loop" in both states, the maximal weight for the loop is the same in both states.

▶ **Definition 1.** *Let $\mathcal{A} = (Q, \Gamma, \mu, \nu)$ be a max-plus-WTA. Two states $p, q \in Q$ are called* siblings *if there exists a tree $u \in T_\Gamma$ and runs $r^p, r^q \in \mathrm{Run}_\mathcal{A}(u)$ with $r^p(\varepsilon) = p$ and $r^q(\varepsilon) = q$. We recall that $\mathrm{Run}_\mathcal{A}(u)$ contains only valid runs.*

Two siblings p, q are called twins *if for every Γ-word s and weights*

$$x = \max_{\substack{r \in \mathrm{Run}^\diamond_\mathcal{A}(s) \\ r(\varepsilon) = r(\Diamond_1(s)) = p}} \mathrm{wt}^\diamond_\mathcal{A}(s, r) \qquad y = \max_{\substack{r \in \mathrm{Run}^\diamond_\mathcal{A}(s) \\ r(\varepsilon) = r(\Diamond_1(s)) = q}} \mathrm{wt}^\diamond_\mathcal{A}(s, r),$$

we have $x = y$ whenever $x \neq -\infty$ and $y \neq -\infty$ holds.

A max-plus-WTA is said to satisfy the *twins property* if all of its siblings are twins. For unambiguous max-plus-WTA, the twins property is a criterion to decide the sequentiality problem. An unambiguous max-plus-WTA possesses a deterministic equivalent if and only if it satisfies the twins property. For words, this result is due to [24, Theorem 12], for trees, we cite the following theorem.

▶ **Theorem 2** ([9, Lemma 5.10], [26, Lemma 17])**.** *Let $\mathcal{A}$ be a trim unambiguous max-plus-WTA. There exists a deterministic max-plus-WTA $\mathcal{A}'$ with $[\![\mathcal{A}]\!] = [\![\mathcal{A}']\!]$ if and only if $\mathcal{A}$ satisfies the twins property. If it exists, it can be effectively constructed.*

It is intuitive that the twins property is a necessary condition if we consider the following *Lipschitz property* which every deterministic max-plus word automaton $\mathcal{A}$ satisfies [18, End of Section 2.4][24, Section 3.2]. If $\mathcal{A}$ is deterministic and L is the largest weight, in terms of absolute value, occurring in $\mathcal{A}$ (excluding $-\infty$), then for two words $w_1 = uv_1$ and $w_2 = uv_2$ which have an accepting run in $\mathcal{A}$, the difference between $[\![\mathcal{A}]\!](w_1)$ and $[\![\mathcal{A}]\!](w_2)$ can be at most $|L|(|v_1| + |v_2| + 2)$. This is clear since the unique runs of $\mathcal{A}$ on w_1 and w_2 will be identical on the prefix u, and then with every remaining letter of each word the difference between both runs cannot grow more than $|L|$.

If an unambiguous max-plus word automaton $\mathcal{A}$ does not satisfy the twins property, we can find states p and q which are siblings and not twins. We assume that our witnesses for this are u and s as above. We consider words of the form $w_1 = us^N v_p$ and $w_2 = us^N v_q$, where v_p and v_q are two fixed words which lead from p and q, respectively, to some final state. For every fixed L, we can choose N sufficiently large to ensure that $|[\![\mathcal{A}]\!](w_1) - [\![\mathcal{A}]\!](w_2)| > |L|(|v_p| + |v_q| + 2)$. It is thus not possible to determinize $\mathcal{A}$ if it does not satisfy the twins property.

The twins property is decidable for both max-plus word automata [1, 4, 24, 25, 16] and max-plus tree automata [8, Section 3]. Deciding whether a max-plus word automaton satisfies the twins property is PSPACE-complete [16]. For max-plus tree automata, the problem is thus PSPACE-hard, but no upper complexity bound is stated in [8]. Note that in general, it is undecidable whether two given siblings are twins [16], but for unambiguous max-plus automata, it was shown to be decidable on both words [1, Section 4] and trees [8, Section 3].

There exist unambiguous max-plus automata which cannot be determinized, but whose behavior is finitely sequential [18, Section 3.1], see also Figure 1. Thus, for the finite

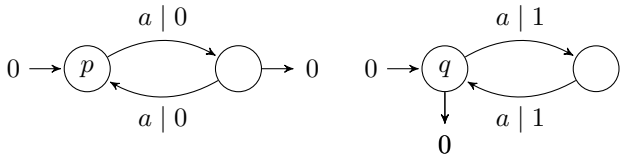

Figure 1 A max-plus word automaton $\mathcal{A}$ over the alphabet $\{a\}$ which is unambiguous, whose behavior is finitely sequential, but which does not satisfy the twins property as p and q are siblings but not twins. The behavior $[\![\mathcal{A}]\!]$ of $\mathcal{A}$ assigns 0 to all words of odd length and $|w|$ to all words w of even length.

sequentiality problem we inevitably have to deal with unambiguous automata in which not all siblings are twins. In the following, we will call two such states *rivals*. For unambiguous automata, which are the only type of max-plus-WTA we consider in this paper, the following definition is equivalent to being siblings and not twins.

▶ **Definition 3.** *Let $\mathcal{A} = (Q, \Gamma, \mu, \nu)$ be a max-plus-WTA. Two states $p, q \in Q$ are called* rivals *if there exists a tree $u \in T_\Gamma$, runs $r^p, r^q \in \mathrm{Run}_\mathcal{A}(u)$ with $r^p(\varepsilon) = p$ and $r^q(\varepsilon) = q$, and a Γ-word s such that $p \xrightarrow{s|x} p$ and $q \xrightarrow{s|y} q$ with $x \neq y$.*

We do not have to consider a maximum over runs here since $\mathcal{A}$ is unambiguous. Also note that by our definition of $\mathrm{Run}^\diamond_\mathcal{A}(s)$, we have $x \neq -\infty$ and $y \neq -\infty$ above.

We now come to the tree fork property which, as we will show, is satisfied by an unambiguous max-plus-WTA if and only if its behavior is not finitely sequential. The property consists of two separate conditions. The first condition intuitively states that there exist two rivals p and q and a Γ-word t which can loop in p, and which can also lead from p to q. The second condition states that there exist two rivals which can occur at prefix-independent positions.

▶ **Definition 4.** *Let $\mathcal{A} = (Q, \Gamma, \mu, \nu)$ be a max-plus-WTA. We say that $\mathcal{A}$ satisfies the* tree fork property *if at least one of the following two conditions is satisfied.*

- **(i)** *There exist rivals $p, q \in Q$ and a Γ-word t with $p \xrightarrow{t|z_p} p$ and $p \xrightarrow{t|z_q} q$ for some weights $z_p, z_q \in \mathbb{R}$. In this case, t is also called a p-q-*fork*.*
- **(ii)** *There exist rivals $p, q \in Q$, a 2-Γ-context $t \in T_{\Gamma_\diamond}$, and a run $r \in \mathrm{Run}^\diamond_\mathcal{A}(t)$ with $r(\Diamond_1(t)) = p$ and $r(\Diamond_2(t)) = q$.*

The tree fork property can be regarded as an extension of the *fork property* which was introduced in [3] and which for max-plus word automata plays the same role as the tree fork property does for max-plus tree automata. Condition (i) is essentially a tree version of the fork property. Casually put, if we take only condition (i) and replace "Γ-word" by "word", we obtain the fork property. The automaton depicted in Figure 2 is unambiguous and satisfies the fork property. Condition (ii) is new and possesses no counterpart in the fork property. We have the following theorem which relates the tree fork property to the finite sequentiality problem.

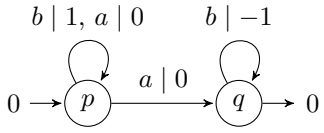

Figure 2 An unambiguous max-plus word automaton $\mathcal{A}$ over the alphabet $\{a, b\}$ which satisfies the fork property. With $u = a$ and $s = b$, we see that p and q are rivals, and a is a p-q-fork. All b's after the last a in a word are treated differently from the b's before the last a. A deterministic automaton cannot "guess" which a is the last in the word, and since there may be arbitrarily many a's in a word, even finitely many deterministic automata cannot compensate this inability to guess.

▶ **Theorem 5.** *Let $\mathcal{A} = (Q, \Gamma, \mu, \nu)$ be a trim unambiguous max-plus-WTA over Γ. Then there exist deterministic max-plus-WTA $\mathcal{A}_1, \dots, \mathcal{A}_n$ over Γ with $[\![\mathcal{A}]\!] = \max_{i=1}^{n} [\![\mathcal{A}_i]\!]$ if and only if $\mathcal{A}$ does not satisfy the tree fork property. In particular, the finite sequentiality problem is decidable for unambiguous max-plus-WTA.*

Proof. Here, we only show that it is decidable whether $\mathcal{A}$ satisfies the tree fork property. The rest of the proof is deferred to Sections 4 and 5, where we show that the behavior of $\mathcal{A}$ is finitely sequential if and only if $\mathcal{A}$ does not satisfy the tree fork property.

To decide whether $\mathcal{A}$ satisfies condition (i), we first show that if there exists a p-q-fork t for two rivals p and q, then there exists a p-q-fork t' of height at most $|Q|^2$. If t is a p-q-fork with $\mathrm{height}(t) > |Q|^2$ and r_p and r_q are runs that realize $p \xrightarrow{t|z_p} p$ and $p \xrightarrow{t|z_q} q$ for some weights $z_p, z_q \in \mathbb{R}$, then by pigeon hole principle there are positions $w_1 <_p w_2$ in s with $r_p(w_1) = r_p(w_2)$ and $r_q(w_1) = r_q(w_2)$. Thus, by removing the part of t between w_1 and w_2, we obtain that $t' = t\langle t|_{w_2} \to w_1\rangle$ is a p-q-fork as well. Iterating this process, we obtain a p-q-fork of height at most $|Q|^2$.

Next, we identify all pairs of rivals, which is possible since for unambiguous max-plus tree automata, we can decide for every pair of states whether they are siblings and not twins [8, Section 3]. Then, for every pair of rivals p, q and all Γ-words t of height at most $|Q|^2$, we check whether t is a p-q-fork. If this yields no p-q-fork, $\mathcal{A}$ does not satisfy condition (i).

In order to decide whether $\mathcal{A}$ satisfies condition (ii), we first compute the relation $\leq$ on Q. This is possible since Q is a finite set and $\leq$ is the smallest transitive and reflexive relation satisfying $\mu(q_1, \dots, q_m, a, q_0) \neq -\infty \to q_0 \leq q_i$ for all transitions $(q_1, \dots, q_m, a, q_0) \in \Delta_{\mathcal{A}}$ and $i \in \{1, \dots, m\}$. Then, by the trimness of $\mathcal{A}$, condition (ii) is satisfied if and only if there exist two rivals p and q, a transition $(q_1, \dots, q_m, a, q_0) \in \Delta_{\mathcal{A}}$ with $\mu(q_1, \dots, q_m, a, q_0) \neq -\infty$, and indices $i, j \in \{1, \dots, m\}$ with $i \neq j$, $q_i \leq p$, and $q_j \leq q$. ◀

The following two sections are dedicated to completing the proof of Theorem 5.

4 Necessity

In this section, we show that if an unambiguous max-plus-WTA $\mathcal{A}$ satisfies either condition (i) or condition (ii) of the tree fork property, then its behavior $[\![\mathcal{A}]\!]$ is not finitely sequential. For condition (i), we adapt the corresponding proof from the word case [3, Theorem 2]. The proof relies on the Lipschitz property of deterministic max-plus automata and its approach is similar to the above outline that the twins property is a necessary condition for determinizability. We omit the proof here as it is a straightforward generalization of the proof from [3].

▶ **Theorem 6.** *Let $\mathcal{A}$ be a trim unambiguous max-plus-WTA over Γ. If $\mathcal{A}$ satisfies condition* (i) *of the tree fork property, then there do not exist deterministic max-plus-WTA $\mathcal{A}_1, \dots, \mathcal{A}_n$ over Γ with $[\![\mathcal{A}]\!] = \max_{i=1}^{n} [\![\mathcal{A}_i]\!]$.*

We consider condition (ii) of the tree fork property. On words, states cannot occur in prefix-independent positions. Thus, this condition is new for the tree case. Intuitively, the reason that the behavior of an unambiguous max-plus-WTA $\mathcal{A}$ cannot be finitely sequential if it satisfies condition (ii) is as follows. Assume we have a 2-Γ-context t and two rivals p and q as in condition (ii) and let u and s be as in the definition of rivals. Then we can construct trees of the form $t(s^n(u), s^n(u))$ such that, by increasing n, the difference between the weights on the two subtrees $s^n(u)$ is arbitrarily large. However, a deterministic automaton necessarily assigns the same weight to both subtrees.

▶ **Theorem 7.** *Let $\mathcal{A}$ be a trim unambiguous max-plus-WTA over Γ. If $\mathcal{A}$ satisfies condition* (ii) *of the tree fork property, then there do not exist deterministic max-plus-WTA $\mathcal{A}_1, \dots, \mathcal{A}_n$ over Γ with $[\![\mathcal{A}]\!] = \max_{i=1}^n [\![\mathcal{A}_i]\!]$.*

Proof (Sketch). For contradiction, we assume that $\mathcal{A}$ satisfies condition (ii) of the tree fork property and that there exist deterministic max-plus-WTA $\mathcal{A}_1, \dots, \mathcal{A}_n$ over Γ with $[\![\mathcal{A}]\!] = \max_{i=1}^n [\![\mathcal{A}_i]\!]$. We write $\mathcal{A}_i = (Q_i, \Gamma, \mu_i, \nu_i)$ and let $N = \max_{i=1}^n |Q_i|$. Let p, q, t be as in condition (ii) of the tree fork property and for the rivals p and q, let u and s be as in the definition of rivals.

By assumption, u can reach both p and q, and s can loop both in p and in q. In particular, the tree $s^N(u)$ can reach p by looping s in p and q by looping s in q. Due to our assumption on t, there hence exists a run on the tree $t' = t(s^N(u), s^N(u))$ which loops s in p on the left branch and in q on the right branch. Since $\mathcal{A}$ is trim, we may even assume that this run is accepting, as on top of t we can always add a Γ-word which leads to a final state.

We assume that $[\![\mathcal{A}]\!](t') = \max_{i=1}^n [\![\mathcal{A}_i]\!](t')$, so there must be some j with $[\![\mathcal{A}]\!](t') = [\![\mathcal{A}_j]\!](t')$. As $\mathcal{A}_j$ is deterministic, the unique accepting run of $\mathcal{A}_j$ on t' is identical on both $s^N(u)$-subtrees. Furthermore, since $N \geq |Q_j|$, we find that by pigeon hole principle some sub-Γ-word s^m of s^N loops in a state of $\mathcal{A}_j$ in the subtrees $s^N(u)$, say with weight z.

We let x and y be the weights such that $\mathcal{A}$ loops s^m in p with weight x and in q with weight y. By choice of s, we have $x \neq y$. We may assume that $x < y$. We consider two cases. First, if $z \geq \frac{x+y}{2}$, then for the tree $t^+ = t(s^{N+m}(u), s^N(u))$ we obtain

$$\max_{i=1}^n [\![\mathcal{A}_i]\!](t^+) \geq [\![\mathcal{A}_j]\!](t^+) = [\![\mathcal{A}_j]\!](t') + z \geq [\![\mathcal{A}_j]\!](t') + \frac{x+y}{2} > [\![\mathcal{A}_j]\!](t') + x = [\![\mathcal{A}]\!](t^+).$$

Note that this follows because $\mathcal{A}$ and $\mathcal{A}_j$ are both unambiguous, i.e., if we construct an accepting run on a given tree, we know that the weight of this run must be the weight assigned to the tree by the automaton. For the other case, namely that $z \leq \frac{x+y}{2}$, we see that for the tree $t^- = t(s^N(u), s^{N-m}(u))$ we obtain

$$\max_{i=1}^n [\![\mathcal{A}_i]\!](t^-) \geq [\![\mathcal{A}_j]\!](t^-) = [\![\mathcal{A}_j]\!](t') - z \geq [\![\mathcal{A}_j]\!](t') - \frac{x+y}{2} > [\![\mathcal{A}_j]\!](t') - y = [\![\mathcal{A}]\!](t^-).$$

In both cases, we see that $[\![\mathcal{A}]\!] = \max_{i=1}^n [\![\mathcal{A}_i]\!]$ does not hold, which is a contradiction. ◀

Together, Theorems 6 and 7 show that if a trim unambiguous max-plus-WTA satisfies the tree fork property, then its behavior is not finitely sequential.

5 Sufficiency

In this section, we show that the behavior of an unambiguous max-plus-WTA $\mathcal{A}$ which does not satisfy the tree fork property is finitely sequential. For simplicity, we begin with a description of our method of proof on max-plus word automata and compare it to the proof method of Bala and Koniński [3].

Both proofs work by distributing the runs of $\mathcal{A}$ across a finite set of unambiguous max-plus word automata such that all of these automata satisfy the twins property. This distribution essentially has the aim of separating the rivals of $\mathcal{A}$. By Theorem 2, these unambiguous automata can then be determinized. The major difference between our approach and that of [3] lies in way we obtain these unambiguous automata. To understand our approach, let p and q be two rivals of $\mathcal{A}$. Furthermore, let $u = u_1 \cdots u_n$ be a word for which there exist valid runs $r^p = p_0 \xrightarrow{u_1} p_1 \xrightarrow{u_2} \cdots \xrightarrow{u_{n-1}} p_{n-1} \xrightarrow{u_n} p$ and $r^q = q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} \cdots \xrightarrow{u_{n-1}} q_{n-1} \xrightarrow{u_n} q$ of $\mathcal{A}$ on u.

We now show that the first occurrence of either p or q in the runs r^p and r^q serves as a "distinguisher" between the two runs. We let i be the smallest index with the property that $p_i \in \{p, q\}$. Similarly, we let j be the smallest index with the property that $q_j \in \{p, q\}$. We obtain valid runs $p_i \xrightarrow{u_{i+1} \cdots u_n} p$ and $q_j \xrightarrow{u_{j+1} \cdots u_n} q$.

Now assume it would hold that $i = j$ and $p_i = q_j$, i.e., the first occurrence is at the same position in the word, and also the state at this position is the same in both runs. Then with $t = u_{i+1} \cdots u_n$, we see that we have valid runs $p_i \xrightarrow{t} p$ and $p_i \xrightarrow{t} q$, where $p_i \in \{p, q\}$. Thus, $\mathcal{A}$ would satisfy the fork property. Since our assumption is that $\mathcal{A}$ does not satisfy the fork property, we have either $i \neq j$ or $p_i \neq q_j$.

This fundamental property is also used in the corresponding proof of [3], but our way of exploiting it differs from [3]. In their proof for word automata, Bala and Koniński use this property implicitly to show that certain states of a modified Schützenberger covering of $\mathcal{A}$ occur at most once in every run [3, Lemma 6]. They can therefore construct a new max-plus automaton which for each run keeps a record of all occurrences of these states. The above mentioned unambiguous automata are then obtained by separating runs with differing records into different automata. For tree automata, the number of these occurrences is unfortunately not bounded, for reasons which we will also indicate below.

For now, we continue outlining our new approach, which is to construct an automaton which adds a distinguishing marker to every run when first encountering one of the rivals p or q. This marker consists of a number, which is used to distinguish occurrences at different positions, and the state from $\{p, q\}$ which was visited first. Whenever reading a letter which causes some valid run to visit p or q for the first time, the automaton selects the smallest marker which was not used by any valid run on the prefix read so far, and annotates it to the run. For example, assume that neither p nor q occur in any valid run the word u, but that our run r on ua leads to p. Then r obtains the marker 1_p. Now assume there is a valid run on uaa which leads to p and which visited neither p nor q before that. Then this run obtains the marker 2_p, since 1_p is already assigned to r. Next, assume that after reading $uaaa$ another marker for p has to be assigned, and that r cannot be extended to a valid run on uaa. Then we assign the marker 1_p, as now no valid run on uaa exists to which the marker 1_p is assigned. See Figure 3 for an example of this annotation process on the word aaa for the automaton depicted there.

With this procedure, runs like r^p and r^q above receive different markers since either one run obtains a marker later than the other, and therefore a different marker, or at least the states they visit first are different, which also leads to a different marker. To separate the rivals of $\mathcal{A}$, we can thus make a copy of $\mathcal{A}$ for every marker, and only allow runs which carry the respective automaton's marker. Whenever a different marker would be assigned, the execution of the run is blocked.

Note here that the number of markers we need for this annotation process is bounded. Since the automaton $\mathcal{A}$ is unambiguous, the number of *valid* runs on every given word is bounded by the number of states in $\mathcal{A}$. If this were not the case, there would exist two distinct

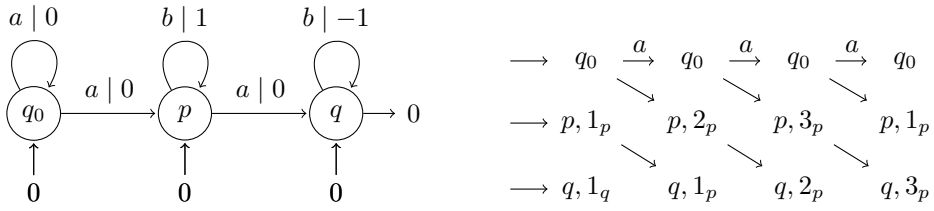

Figure 3 On the left, an unambiguous max-plus word automaton over the alphabet $\{a, b\}$ which does not satisfy the twins property but whose behavior is finitely sequential. On the right, an illustration of the runs of the automaton on the words ε, a, aa, and aaa together with appropriate markers. Arrows indicate a transition. The states p and q are rivals with witnesses $u = \varepsilon$ and $s = b$.

valid runs on the same word which lead to the same state, from which a counterexample to the unambiguity of $\mathcal{A}$ could be constructed. In particular, the number of markers assigned at any given "time" is bounded by the number of states of $\mathcal{A}$.

All of this can easily be generalized to the situation where there is more than one pair of rivals. Then, runs simply obtain a marker for each pair of rivals of the automaton, and the copies of $\mathcal{A}$ allow a distinguished marker for each of these pairs.

Unfortunately, these ideas do not translate to trees as easily. For example, consider the runs in Figure 4. Intuitively, both runs should obtain the marker 1_p. However, since p and q are rivals, this marker does not serve the purpose of distinguishing runs as it does in the word case. The first p occurs in different subtrees of both runs, thus the annotation of distinct markers is not possible. Also, it is easy to construct an automaton where a rival p can occur at arbitrarily many pairwise prefix-independent positions, thus a simple lexicographic distinction is not possible. This is also the reason why the approach from [3] does not work for tree automata.

$$\begin{aligned} \nu(q) &= 0 \\ \mu(p, a, q) &= 0 \\ \mu(p, b, p) &= 1 \\ \mu(q, b, q) &= -1 \\ \mu(p, q_0, c, p) = \mu(q_0, p, c, q) &= 0 \\ \mu(d, p) = \mu(d, q_0) &= 0 \end{aligned}$$

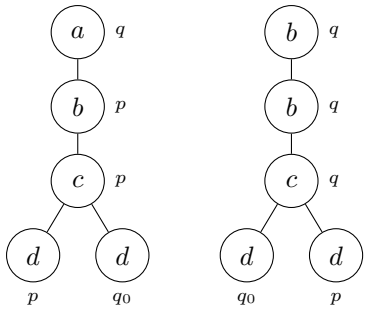

Figure 4 Two accepting runs of the max-plus tree automaton $\mathcal{A} = (\{q_0, p, q\}, \Gamma, \mu, \nu)$ over the ranked alphabet $\Gamma = \{a, b, c, d\}$ where $c \in \Gamma^{(2)}, a, b \in \Gamma^{(1)}, d \in \Gamma^{(0)}$. All unspecified weights are assumed to be $-\infty$. The states p and q are rivals.

Our solution is to distribute not the runs of the automaton $\mathcal{A}$, but the runs of its *Schützenberger covering*. The Schützenberger covering of a max-plus automaton $\mathcal{A}$ is a max-plus automaton which possesses the same behavior as $\mathcal{A}$. It has already been employed in a number of decidability results for max-plus automata [18, 3, 2, 26]. Its construction is inspired by a paper of Schützenberger [32] and was made explicit by Sakarovitch in [29].

To better explain the idea behind its construction, we first point out a certain aspect of the classical powerset construction for finite automata [28]. Assume that $\mathcal{D}$ is the result of applying the powerset construction to an NFA $\mathcal{B}$. Then we might say that for a word

$w = w_1 w_2$, the state which $\mathcal{D}$ is in after reading the prefix w_1 is the set of all states which $\mathcal{B}$ *could* be in after reading w_1. Similarly, the Schützenberger covering of a max-plus automaton $\mathcal{A}$ annotates to every state of a run of $\mathcal{A}$ on a word w the set of all states which "$\mathcal{A}$ could be in" at this point, i.e., which can be reached by some valid run on the considered prefix of w. Like the powerset construction, these ideas easily carry over to trees.

The reason we consider the Schützenberger covering of $\mathcal{A}$ is that each pair $\mathbf{p}, \mathbf{q}$ of its rivals satisfies the following property. For every tree t, either (1) $\mathbf{p}$ and $\mathbf{q}$ do not occur together in any run on t or (2) $\mathbf{p}$ and $\mathbf{q}$ occur only linearly, i.e., there is a distinguished branch of t such that for every run on t, all occurrences of $\mathbf{p}$ and $\mathbf{q}$ lie on this branch. In particular, the situation of Figure 4 is not possible. All pairs which satisfy the first condition can simply be separated into different automata, all pairs which satisfy the second condition can be handled like in the word case. The proof of this is non-trivial and needs some preparation. We begin with the formal definition of the Schützenberger covering.

For the rest of this section, let $\mathcal{A} = (Q, \Gamma, \mu, \nu)$ be a trim unambiguous max-plus-WTA which does not satisfy the tree fork property.

▶ **Definition 8** (Schützenberger covering, [29]). *The Schützenberger covering $\mathcal{S} = (Q_\mathcal{S}, \Gamma, \mu_\mathcal{S}, \nu_\mathcal{S})$ of $\mathcal{A}$ is the trim part of the max-plus-WTA $(Q \times \mathcal{P}(Q), \Gamma, \mu', \nu')$ defined for $a \in \Gamma$ with $\mathrm{rk}_\Gamma(a) = m$ and $(p_0, P_0), \dots, (p_m, P_m) \in Q \times \mathcal{P}(Q)$ by*

$$\mu'((p_1, P_1), \dots, (p_m, P_m), a, (p_0, P_0)) =$$
$$\begin{cases} \mu(p_1, \dots, p_m, a, p_0) & \text{if } P_0 = \{q_0 \in Q \mid \exists (q_1, \dots, q_m) \in P_1 \times \dots \times P_m \text{ with} \\ & \mu(q_1, \dots, q_m, a, q_0) \neq -\infty\} \\ -\infty & \text{otherwise} \end{cases}$$
$$\nu'(p_0, P_0) = \nu(p_0).$$

We let $\pi_1 \colon Q \times \mathcal{P}(Q) \to Q$, $(p, P) \mapsto p$ and $\pi_2 \colon Q \times \mathcal{P}(Q) \to \mathcal{P}(Q)$, $(p, P) \mapsto P$ be the projections.

It is elementary to show that for a run of $\mathcal{S}$ on a tree t, the second entry of the state at a position w consists of all states of $\mathcal{A}$ which can be reached by a valid run of $\mathcal{A}$ on $t|_w$. In particular, two runs on the same tree coincide on their second entries. Furthermore, projecting all states of a run of $\mathcal{S}$ to their first coordinate yields a run of $\mathcal{A}$, and the weights of these runs coincide. It follows that $\mathcal{S}$ is unambiguous and satisfies $[\![\mathcal{S}]\!] = [\![\mathcal{A}]\!]$. Also, $\mathcal{S}$ is trim by definition.

We can make the following observation about the rivals of $\mathcal{S}$. Let $\mathbf{p}$ and $\mathbf{q}$ be rivals of $\mathcal{S}$ and let u and s be as in the definition of rivals. Since all runs of $\mathcal{S}$ on u coincide on the second entry of the state at the root, $\mathbf{p}$ and $\mathbf{q}$ also coincide on their second entry. Moreover, as projecting the runs of $\mathcal{S}$ on u and s to their first entries yields runs of $\mathcal{A}$ on u and s, respectively, we additionally see that the first entries of $\mathbf{p}$ and $\mathbf{q}$ are rivals in $\mathcal{A}$. Thus, if two states $\mathbf{p}, \mathbf{q} \in Q_\mathcal{S}$ are rivals in $\mathcal{S}$, then $\mathbf{p} = (p, P)$ and $\mathbf{q} = (q, P)$ for some set $P \subseteq Q$ and two states $p, q \in Q$ which are rivals in $\mathcal{A}$.

In order to prove some deeper results about the rivals of $\mathcal{S}$, we need two preparatory lemmata. As a first simplification, we show that we may assume that two rivals p and q of $\mathcal{A}$ are always comparable with respect to the relation $\leq$. To see this, note that by condition (ii) of the tree fork property, p and q may not occur in prefix-independent positions in a run. If in addition, p and q can also not appear in prefix-dependent positions in a run, they never appear in the same run of $\mathcal{A}$. Thus, we can create two copies of $\mathcal{A}$, one in which we remove p and one in which we remove q, and the pointwise maximum of these two automata will be equivalent to the behavior of $\mathcal{A}$.

▶ **Lemma 9.** *For all rivals $p, q \in Q$, we may assume that either $p \leq q$ or $q \leq p$, or both.*

Next, we note an elementary statement about self-maps $f\colon X \to X$. Namely, if X is a finite set and $f\colon X \to X$ a mapping, then for every $a \in X$ there exists some element $b \in X$ and an integer $n \geq 1$ such that after n iterations of f, both a and b are mapped to b. To see this, consider the elements $a, f(a), f^2(a), \ldots, f^{|X|}(a)$. By pigeon hole principle, there are numbers $0 \leq m_1 < m_2 \leq |X|$ with $f^{m_1}(a) = f^{m_2}(a)$. Then if we choose $n \geq m_1$ as a multiple of $m_2 - m_1$ and $b = f^n(a)$, we see that $f^n(a) = b = f^n(b)$.

▶ **Lemma 10.** *Let X be a finite set and $f\colon X \to X$ a mapping. Then for every $a \in X$, there exists an element $b \in X$ and an integer $n \geq 1$ with $f^n(a) = b = f^n(b)$. Here, f^n is the n-th iterate of f, i.e., $f^0 = \mathrm{id}_X$ and $f^{m+1} = f \circ f^m$.*

We come to the first important property which all rivals of $\mathcal{S}$ satisfy. Namely, if $P \subseteq Q$ is the second entry of *some* rival, then it cannot occur in the form of a "triangle" in any valid run of $\mathcal{S}$. More precisely, if we have a run r and positions w, wv_1, and wv_2 such that the second entry of $r(w)$, $r(wv_1)$, and $r(wv_2)$ is P, then wv_1 and wv_2 are prefix-dependent.

▶ **Lemma 11.** *Let $(p, P), (q, P) \in Q_{\mathcal{S}}$ be rivals in $\mathcal{S}$. Furthermore, let $t' \in T_\Gamma$ be a tree, $r' \in \mathrm{Run}_{\mathcal{S}}(t')$ a run of $\mathcal{S}$ on t', and $w_1, w_2 \in \mathrm{pos}(t')$ be positions in t'. If $\pi_2 \circ r'(\varepsilon) = \pi_2 \circ r'(w_1) = \pi_2 \circ r'(w_2) = P$, then w_1 and w_2 are prefix-dependent.*

Proof (Sketch). We proceed by contradiction and assume that t', r', w_1, w_2 as in the statement of the lemma exist such that w_1 and w_2 are prefix-independent. We show that then, $\mathcal{A}$ satisfies condition (i) of the tree fork property. For the rivals (p, P) and (q, P), let u and s be as in the definition of rivals and let $v = \Diamond_1(s)$.

By assumption, u can reach (p, P) and s can loop in (p, P), thus the trees $s^{|P|}(u)$ and $s^{|P|^{|P|}}(u)$ can reach (p, P). Due to the construction of $\mathcal{S}$, this means both of these trees can also reach the states of r' at w_1 and w_2. In particular, there exists a run of $\mathcal{S}$ on the tree $t = t'\langle s^{|P|}(u) \to w_1\rangle\langle s^{|P|^{|P|}}(u) \to w_2\rangle$ and for this run, the second entry of every state at the beginning or end of an s-loop is P. In addition, t leads to a state with second entry P, so there in fact exist $|P|$ runs of $\mathcal{S}$ on t, one for each state in P. We let $r_1, \ldots, r_{|P|}$ be the projections of these runs to their first entry and obtain $|P|$ runs of $\mathcal{A}$ on t where the state at the root and all states at the beginning or end of an s-loop are from P.

By pigeonhole principle, there is some subloop s^n below w_2 which loops in all runs at the same time, i.e., where for some n_1 we have $r_i(w_2 v^{n_1}) = r_i(w_2 v^{n_1+n})$ for all runs r_i. For each r_i, we let $q_i = r_i(w_2 v^{n_1}) \in P$ be the state which r_i loops in and let x_i be the weight of this loop.

If $x_i \neq x_j$ for some i and j, the states q_i and q_j are rivals in $\mathcal{A}$ with witnesses u and s^n. By Lemma 9, we may therefore assume $q_i \leq q_j$. Again by pigeon hole principle, the run r_i loops below w_1 in s^m for some $m \geq 1$ with some state $p_i \in P$, say with weight y_i. Due to $x_i \neq x_j$, we have $mx_i \neq ny_i$ or $mx_j \neq ny_i$. Since u can reach every state from P, the state p_i is thus a rival of q_i or q_j with witnesses u and s^{nm}. From the existence of r_i and the assumption that $q_i \leq q_j$, we see that p_i can occur prefix-independently both from q_i and from q_j. This is a contradiction to the assumption that $\mathcal{A}$ does not satisfy the tree fork property. It must therefore hold that $x_1 = \ldots = x_{|P|}$.

We let x and y be the weights such that $\mathcal{A}$ loops s in p with weight x and in q with weight y. Then from $x \neq y$ it follows that $nx \neq x_1$ or $ny \neq x_1$, so the states q_i are either all rivals of p or all rivals of q with witnesses u and s^n. We assume all q_i to be rivals of p and apply Lemma 10 to the mapping $f\colon P \to \{q_1, \ldots, q_{|P|}\}, r_i(\varepsilon) \mapsto q_i$ with $a = p$ to obtain $q_j \in P$ and $m \geq 1$ such that $f^m(p) = q_j = f^m(q_j)$. Then with $\tilde{s} = t\langle \diamond \to w_2 v^{n_1}\rangle$, we see that the Γ-word $\tilde{s}^m$ is a q_j-p-fork, i.e., $\mathcal{A}$ satisfies condition (i) of the tree fork property. ◀

The previous lemma showed that if P is the second entry of some rival from $\mathcal{S}$, states with second entry P do not occur in the form of a triangle. The next lemma shows that even prefix-independent occurrences are restricted to a certain degree. Namely, if we have two rivals (p, P) and (q, P) with $p \leq q$, then all occurrences of P are prefix-dependent on (p, P).

▶ **Lemma 12.** *Let $(p, P), (q, P) \in Q_{\mathcal{S}}$ be rivals in $\mathcal{S}$ with $p \leq q$. Furthermore, let $t' \in T_\Gamma$ be a tree, $r' \in \mathrm{Run}_{\mathcal{S}}(t')$ a run of $\mathcal{S}$ on t', and $w_1 \in \mathrm{pos}(t')$ a position in t' with $r'(w_1) = (p, P)$. Then all positions $w_2 \in \mathrm{pos}(t')$ with $\pi_2 \circ r'(w_2) = P$ are prefix-dependent on w_1.*

Proof (Sketch). We proceed by contradiction and take $(p, P), (q, P), t', r', w_1$ as in the statement of the lemma and assume that there exists a position $w_2 \in \mathrm{pos}(t')$ which is prefix-independent from w_1 and for which $\pi_2 \circ r'(w_2) = P$. We show that under these assumptions, $\mathcal{A}$ satisfies condition (ii) of the tree fork property. For the rivals (p, P) and (q, P), let u and s be as in the definition of rivals.

As we have seen in the proof of Lemma 11, the tree $s^{|P|}(u)$ can reach (p, P), so due to the construction of $\mathcal{S}$, it can also reach the state of r' at w_2. Thus, there exists a run of $\mathcal{S}$ on the tree $t = t'\langle s^{|P|}(u) \to w_2\rangle$ for which the state at w_1 is (p, P) and for which the second entry of every state at the beginning or end of an s-loop is P. We let r be the projection of this run to the first entries of the states.

By pigeonhole principle, we find some subloop s^n below w_2 in r which loops in a state $p' \in P$. Let z be the weight of this loop and let x and y be the weights such that $\mathcal{A}$ loops s in p with weight x and in q with weight y. Due to $x \neq y$, we have $nx \neq z$ or $ny \neq z$. Since u can reach every state from P, the state p' is a rival of p or q with witnesses u and s^n. From the fact that $r(w_1) = p$ and the assumption that $p \leq q$, we see that p' can occur prefix-independently both from p and from q. This is a contradiction to the assumption that $\mathcal{A}$ does not satisfy the tree fork property. ◀

We can now prove that every run of $\mathcal{S}$ satisfies at least one of the following two conditions. If (p, P) and (q, P) are rivals with $p \leq q$, then for every run r on a tree t either (i) (p, P) does not occur in r or (ii) all states with second entry P occur along a distinguished branch of t. This property enables us to apply the idea from the word case of using markers to indicate the first visit of a rival in a run. If u is a witness for (p, P) and (q, P) to be siblings, there is in particular a run on u which leads to (p, P). This run then satisfies condition (ii) and since the second entries of runs on the same tree coincide, *all* states with second entry P occur along a distinguished branch of u in *every* run of $\mathcal{S}$ on u. This is true in particular for the two rivals (p, P) and (q, P).

▶ **Theorem 13.** *Let $(p, P), (q, P) \in Q_{\mathcal{S}}$ be rivals in $\mathcal{S}$ with $p \leq q$. Then for every tree $t \in T_\Gamma$ and every run $r \in \mathrm{Run}_{\mathcal{S}}(t)$ of $\mathcal{S}$ on t, at least one of the following two conditions holds.*

- **(i)** *The state (p, P) does not occur in r, i.e., $r(w) \neq (p, P)$ for all $w \in \mathrm{pos}(t)$.*
- **(ii)** *All states with second entry P occur linearly in r, i.e., for all $w_1, w_2 \in \mathrm{pos}(t)$ with $\pi_2 \circ r(w_1) = \pi_2 \circ r(w_2) = P$ we have $w_1 \leq_p w_2$ or $w_2 \leq_p w_1$.*

Proof. Let $(p, P), (q, P), t, r$ be as in the statement of the theorem. Assume that (i) does not hold, i.e., there is a position $w \in \mathrm{pos}(t)$ with $r(w) = (p, P)$. Let $w_1, w_2 \in \mathrm{pos}(t)$ be two positions with $\pi_2 \circ r(w_1) = \pi_2 \circ r(w_2) = P$. By Lemma 12, we see that then w_1 and w_2 are prefix-dependent on w. From the definition of the prefix relation, we see that if either $w_1 \leq_p w$ or $w_2 \leq_p w$, then all three positions are in prefix relation. We thus consider the case that $w \leq_p w_1$ and $w \leq_p w_2$. In this case, we see from Lemma 11 that w_1 and w_2 are prefix-dependent. ◀

We now construct the automaton which tracks the first occurrences of rivals, and whose runs we distribute across multiple automata in order to separate all rivals.

▶ **Construction 14.** Let $R_1, \dots, R_n \subseteq Q_{\mathcal{S}}$ be an enumeration of all (unordered) pairs of rivals of $\mathcal{S}$, i.e., for all $i \in \{1, \dots, n\}$ we have $R_i = \{(p_i, P_i), (q_i, P_i)\}$ such that (p_i, P_i) and (q_i, P_i) are rivals in $\mathcal{S}$ and for every two rivals $(p, P), (q, P) \in Q_{\mathcal{S}}$, we have $R_i = \{(p, P), (q, P)\}$ for some $i \in \{1, \dots, n\}$. Since by Lemma 9, we may assume that all rivals in $\mathcal{A}$ are in $\leq$-relation, we assume in the following that p_i and q_i are named such that $p_i \leq q_i$ for all $i \in \{1, \dots, n\}$.

For each pair of rivals R_i, we define a set of markers by $I_i = \{0, |Q|+1\} \cup (\{1, \dots, |Q|\} \times R_i)$. The set of all combined records of markers is defined by $I = I_1 \times \ldots \times I_n$. For $\bar{a} \in I$, we denote by $\bar{a}[i]$ the i-th entry of $\bar{a}$.

Intuitively, the states of our new automaton will consist of a state from $\mathcal{S}$ together with a record of markers from I. However, in order to properly update markers, we need to know in each step the records of *all* other runs as well. Thus, our states will be from $Q_{\mathcal{S}} \times I \times \mathcal{P}(Q_{\mathcal{S}} \times I)$. In order to define the transition function of our new automaton, we first define how markers are updated. Assume we transition into the state $\mathbf{q} \in Q_{\mathcal{S}}$, we have m subtrees below our current position in the tree, the runs we consider on these subtrees have obtained markers $\bar{a}_1, \dots, \bar{a}_m \in I$, and the sets of states we *could* be in on these trees, together with their markers, are given by $A_1, \dots, A_m \subseteq Q_{\mathcal{S}} \times I$.

Every pair $(\mathbf{p}, \bar{a}) \in A_k$ corresponds to exactly one run of $\mathcal{S}$ on the k-th subtree together with its markers. Since $\mathcal{S}$ is unambiguous, we can therefore assume that $|A_k| \leq |Q|$. Also, since $\bar{a}_k$ is the marker of a run on the k-th subtree, we may assume that $(Q_{\mathcal{S}} \times \{\bar{a}_k\}) \cap A_k \neq \emptyset$.

For $k \in \{1, \dots, m\}$ and $i \in \{1, \dots, n\}$, we define the sets of unassigned counters $B_k[i] \subseteq \{1, \dots, |Q|\}$ by $B_k[i] = \{1, \dots, |Q|\} \setminus \{j \mid \exists (\mathbf{p}, \bar{a}) \in A_k \text{ with } \bar{a}[i] \in \{j\} \times R_i\}$. Then if for all $k \in \{1, \dots, m\}$ we have $|A_k| \leq |Q|$ and $(Q_{\mathcal{S}} \times \{\bar{a}_k\}) \cap A_k \neq \emptyset$, we define the record of markers $\bar{b}$ for our current position by (explanations below)

$$\bar{b}[i] = \begin{cases} 0 & \text{if } m = 0 \text{ and } \mathbf{q} \notin R_i \\ (1, \mathbf{q}) & \text{if } m = 0 \text{ and } \mathbf{q} \in R_i \\ \bar{a}_k[i] & \text{if } k \in \{1, \dots, m\} \text{ satisfies: } \bar{a}_l[i] = 0 \text{ for all } l \neq k \text{ and} \\ & \text{either } \bar{a}_k[i] \neq 0 \text{ or } \mathbf{q} \notin R_i \\ (\min B_k[i], \mathbf{q}) & \text{if } \mathbf{q} \in R_i \text{ and } k \in \{1, \dots, m\} \text{ satisfies:} \\ & \bar{a}_k[i] = 0 \text{ and for all } l \neq k \text{ and all } (\mathbf{p}, \bar{a}) \in A_l \colon \bar{a}[i] = 0 \\ |Q| + 1 & \text{otherwise} \end{cases}$$

for $i \in \{1, \dots, n\}$. If $|A_k| > |Q|$ or $Q_{\mathcal{S}} \times \{\bar{a}_k\} \cap A_k = \emptyset$ for some k, we let $\bar{b}[1] = \ldots = \bar{b}[n] = |Q| + 1$. Note that $\min B_k[i]$ in above case distinction always exists since $|A_k| \leq |Q|$, $(Q_{\mathcal{S}} \times \{\bar{a}_k\}) \cap A_k \neq \emptyset$, and in the case in question we have $\bar{a}_k[i] = 0$. We define $\mathcal{I}(\mathbf{q}, \bar{a}_1, \dots, \bar{a}_m, A_1, \dots, A_m) = \bar{b}$.

Case 1 of the definition above means our current position is a leaf and $\mathbf{q}$ is not from R_i, so we assign the dummy marker 0. Case 2 means our current position is a leaf and $\mathbf{q}$ is from R_i, so we assign the marker $(1, \mathbf{q})$. Case 3 means that either (1) there is exactly one subtree below our current position which already obtained a marker different from 0 and we keep this marker for our current position, or (2) the markers of all subtrees are 0 and $\mathbf{q}$ is also not from R_i, so we continue with the dummy marker 0.

Case 4 means the markers of all subtrees below our current position are 0, the state $\mathbf{q}$ is from R_i, and there is at most one subtree on which runs exist that obtained a marker for R_i. Then, we take the smallest number which is not already used in a marker for R_i in any run on this subtree, and use this number together with $\mathbf{q}$ as the marker for our current position.

Case 5, the "otherwise-case", applies in two situations. This case means that either (1) two distinct subtrees below our current position have already obtained a marker, or that (2) all markers below our current position are 0 and $\mathbf{q}$ is from R_i, but we cannot apply case 4 as there are two distinct subtrees on which runs exist which obtained markers for R_i. In other words, markers were assigned nonlinearly, and our run satisfies only condition (i) of Theorem 13. In this case, we assign the dummy marker $|Q|+1$.

The extra case covers the situation where in case 4, the set $B_k[i]$ would be empty. This case is necessary to ensure our definition is formally complete, but in our applications of the operator $\mathcal{I}$ it will not actually occur.

We define our "run-marking" max-plus-WTA $\mathcal{B} = (\tilde{Q}, \Gamma, \tilde{\mu}, \tilde{\nu})$ as follows. We let $\tilde{Q}' = Q_{\mathcal{S}} \times I \times \mathcal{P}(Q_{\mathcal{S}} \times I)$ and let $\mathcal{B}$ be the trim part of the automaton $\mathcal{B}' = (\tilde{Q}', \Gamma, \tilde{\mu}', \tilde{\nu}')$ defined for $a \in \Gamma$ with $\mathrm{rk}_\Gamma(a) = m$ and $(\mathbf{p}_0, \bar{a}_0, A_0), \dots, (\mathbf{p}_m, \bar{a}_m, A_m) \in Q_{\mathcal{S}} \times I \times \mathcal{P}(Q_{\mathcal{S}} \times I)$ by

$$\tilde{\mu}'((\mathbf{p}_1, \bar{a}_1, A_1), \dots, (\mathbf{p}_m, \bar{a}_m, A_m), a, (\mathbf{p}_0, \bar{a}_0, A_0)) =$$
$$\begin{cases} \mu_{\mathcal{S}}(\mathbf{p}_1, \dots, \mathbf{p}_m, a, \mathbf{p}_0) & \text{if } \bar{a}_0 = \mathcal{I}(\mathbf{p}_0, \bar{a}_1, \dots, \bar{a}_m, A_1, \dots, A_m) \text{ and} \\ & A_0 = \{(\mathbf{q}_0, \bar{b}_0) \in Q_{\mathcal{S}} \times I \mid \exists((\mathbf{q}_1, \bar{b}_1), \dots, (\mathbf{q}_m, \bar{b}_m)) \in \\ & A_1 \times \ldots \times A_m \text{ with } \mu_{\mathcal{S}}(\mathbf{q}_1, \dots, \mathbf{q}_m, a, \mathbf{q}_0) \neq -\infty \text{ and} \\ & \bar{b}_0 = \mathcal{I}(\mathbf{q}_0, \bar{b}_1, \dots, \bar{b}_m, A_1, \dots, A_m)\} \\ -\infty & \text{otherwise} \end{cases}$$
$$\tilde{\nu}'(\mathbf{p}_0, \bar{a}_0, A_0) = \nu_{\mathcal{S}}(\mathbf{p}_0).$$

The automaton $\mathcal{B}$ is trim, unambiguous, and satisfies $[\![\mathcal{B}]\!] = [\![\mathcal{A}]\!]$. Furthermore, if $(\mathbf{p}, \bar{a}, A)$ and $(\mathbf{q}, \bar{b}, B)$ are rivals in $\mathcal{B}$, we have $\bar{a}[i] \neq \bar{b}[i]$ for some $i \in \{1, \dots, n\}$.

We come to our final construction where we distribute the runs of $\mathcal{B}$ across multiple automata. For every record of markers $\bar{c} \in I$, we construct one automaton $\mathcal{B}_{\bar{c}}$ which for each pair of rivals R_i admits only runs using the markers 0 and $\bar{c}[i]$. All runs in which rivals occur nonlinearly are covered by admitting the marker $|Q|+1$. All other runs are covered by admitting an appropriate marker from $\{1, \dots, |Q|\} \times R_i$.

▶ **Construction 15.** For every tuple $\bar{c} \in I$, we define a max-plus-WTA $\mathcal{B}_{\bar{c}} = (\tilde{Q}_{\bar{c}}, \Gamma, \tilde{\mu}, \tilde{\nu})$ by removing states from $\mathcal{B}$ through

$$\tilde{Q}_{\bar{c}} = \{(\mathbf{p}, \bar{a}, A) \in \tilde{Q} \mid \text{for all } i \in \{1, \dots, n\} \text{ it holds: if } \bar{c}[i] = |Q|+1 \text{ then } \mathbf{p} \neq (p_i, P_i),$$
$$\text{and if } \bar{c}[i] \neq |Q|+1 \text{ then } \bar{a}[i] \in \{0, \bar{c}[i]\}\}.$$

The automata $\mathcal{B}_{\bar{c}}$ are unambiguous, their pointwise maximum is equivalent to the behavior of $\mathcal{A}$, and they all satisfy the twins property, which means that they can be determinized.

▶ **Theorem 16.** *We have* $[\![\mathcal{A}]\!] = \max_{\bar{c} \in I} [\![\mathcal{B}_{\bar{c}}]\!]$ *and for every* $\bar{c} \in I$*, the automaton* $\mathcal{B}_{\bar{c}}$ *is unambiguous and satisfies the twins property.*

We now obtain a finitely sequential representation of $\mathcal{A}$ by applying Theorem 2 to the automata $\mathcal{B}_{\bar{c}}$. In particular, we see that the behavior of a trim unambiguous max-plus-WTA is finitely sequential if it does not satisfy the tree fork property. This concludes the proof of Theorem 5. ◀

References

1 Cyril Allauzen and Mehryar Mohri. Efficient Algorithms for Testing the Twins Property. *Journal of Automata, Languages and Combinatorics*, 8(2):117–144, 2003.

2 Sebastian Bala. Which Finitely Ambiguous Automata Recognize Finitely Sequential Functions? In Krishnendu Chatterjee and Jiří Sgall, editors, *38th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 8087 of *Lecture Notes in Computer Science*, pages 86–97. Springer, 2013.

3 Sebastian Bala and Artur Koniński. Unambiguous Automata Denoting Finitely Sequential Functions. In Adrian-Horia Dediu, Carlos Martín-Vide, and Bianca Truthe, editors, *7th International Conference on Language and Automata Theory and Applications (LATA)*, volume 7810 of *Lecture Notes in Computer Science*, pages 104–115. Springer, 2013.

4 Marie-Pierre Béal, Olivier Carton, Christophe Prieur, and Jacques Sakarovitch. Squaring transducers: an efficient procedure for deciding functionality and sequentiality. *Theoretical Computer Science*, 292(1):45–63, 2003.

5 Jean Berstel and Christophe Reutenauer. *Rational Series and Their Languages*. Springer, 1988.

6 Johanna Björklund, Frank Drewes, and Niklas Zechner. An Efficient Best-Trees Algorithm for Weighted Tree Automata over the Tropical Semiring. In Adrian-Horia Dediu, Enrico Formenti, Carlos Martín-Vide, and Bianca Truthe, editors, *9th International Conference on Language and Automata Theory and Applications (LATA)*, volume 8977 of *Lecture Notes in Computer Science*, pages 97–108. Springer, 2015.

7 Meera Blattner and Tom Head. Automata That Recognize Intersections of Free Submonoids. *Information and Control*, 35(3):173–176, 1977.

8 Matthias Büchse and Anja Fischer. Deciding the Twins Property for Weighted Tree Automata over Extremal Semifields. In Frank Drewes and Marco Kuhlmann, editors, *Proceedings of the Workshop on Applications of Tree Automata Techniques in Natural Language Processing (ATANLP)*, pages 11–20. Association for Computational Linguistics, 2012.

9 Matthias Büchse, Jonathan May, and Heiko Vogler. Determinization of Weighted Tree Automata Using Factorizations. *Journal of Automata, Languages and Combinatorics*, 15(3/4):229–254, 2010.

10 Laure Daviaud, Pierre Guillon, and Glenn Merlet. Comparison of Max-Plus Automata and Joint Spectral Radius of Tropical Matrices. In Kim G. Larsen, Hans L. Bodlaender, and Jean-François Raskin, editors, *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 83 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 19:1–19:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

11 Manfred Droste, Werner Kuich, and Heiko Vogler, editors. *Handbook of Weighted Automata*. EATCS Monographs in Theoretical Computer Science. Springer, 2009.

12 Emmanuel Filiot, Ismaël Jecker, Nathan Lhote, Guillermo A. Pérez, and Jean-François Raskin. On delay and regret determinization of max-plus automata. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12. IEEE Computer Society, 2017.

13 Kōsaburō Hashiguchi. Algorithms for Determining Relative Star Height and Star Height. *Information and Computation*, 78(2):124–169, 1988.

14 Kōsaburō Hashiguchi, Kenichi Ishiguro, and Shūji Jimbo. Decidability of The Equivalence Problem for Finitely Ambiguous Finance Automata. *International Journal of Algebra and Computation*, 12(3):445–461, 2002.

15 Daniel Kirsten. A Burnside Approach to the Termination of Mohri's Algorithm for Polynomially Ambiguous Min-Plus-Automata. *Informatique Théorique et Applications*, 42(3):553–581, 2008.

16 Daniel Kirsten. Decidability, undecidability, and PSPACE-completeness of the twins property in the tropical semiring. *Theoretical Computer Science*, 420:56–63, 2012.

17 Daniel Kirsten and Sylvain Lombardy. Deciding Unambiguity and Sequentiality of Polynomially Ambiguous Min-Plus Automata. In Susanne Albers and Jean-Yves Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 3 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 589–600. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2009.

18 Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theoretical Computer Science*, 327(3):349–373, 2004.

19 Jan Komenda, Sébastien Lahaye, Jean-Louis Boimond, and Ton van den Boom. Max-plus algebra in the history of discrete event systems. *Annual Reviews in Control*, 45:240–249, 2018.

20 Adam Koprowski and Johannes Waldmann. Max/Plus Tree Automata for Termination of Term Rewriting. *Acta Cybernetica*, 19(2):357–392, 2009.

21 Daniel Krob. The Equality Problem for Rational Series with Multiplicities in the tropical Semiring is Undecidable. *International Journal of Algebra and Computation*, 4(3):405–426, 1994.

22 Werner Kuich and Arto Salomaa. *Semirings, Automata, Languages*. Springer, 1986.

23 Filip Mazowiecki and Cristian Riveros. Pumping Lemmas for Weighted Automata. In Rolf Niedermeier and Brigitte Vallée, editors, *35th Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 50:1–50:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

24 Mehryar Mohri. Finite-State Transducers in Language and Speech Processing. *Computational Linguistics*, 23(2):269–311, 1997.

25 Mehryar Mohri. Weighted automata algorithms. In Manfred Droste, Werner Kuich, and Heiko Vogler, editors, *Handbook of Weighted Automata*, EATCS Monographs in Theoretical Computer Science, chapter 6, pages 213–254. Springer, 2009.

26 Erik Paul. The Equivalence, Unambiguity and Sequentiality Problems of Finitely Ambiguous Max-Plus Tree Automata are Decidable. In Kim G. Larsen, Hans L. Bodlaender, and Jean-François Raskin, editors, *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 83 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 53:1–53:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

27 Slav Petrov. Latent Variable Grammars for Natural Language Parsing. In *Coarse-to-Fine Natural Language Processing*, chapter 2, pages 7–46. Springer, 2012.

28 Michael O. Rabin and Dana S. Scott. Finite Automata and Their Decision Problems. *IBM Journal of Research and Development*, 3(2):114–125, 1959.

29 Jacques Sakarovitch. A Construction on Finite Automata that has Remained Hidden. *Theoretical Computer Science*, 204(1-2):205–231, 1998.

30 Arto Salomaa and Matti Soittola. *Automata-Theoretic Aspects of Formal Power Series*. Springer, 1978.

31 Marcel-Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2–3):245–270, 1961.

32 Marcel-Paul Schützenberger. Sur les Relations Rationnelles Entre Monoïdes Libres. *Theoretical Computer Science*, 3(2):243–259, 1976.

33 Helmut Seidl. On the Finite Degree of Ambiguity of Finite Tree Automata. *Acta Informatica*, 26(6):527–542, 1989.

34 Imre Simon. Limited Subsets of a Free Monoid. In *19th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 143–150. IEEE Computer Society, 1978.

35 Imre Simon. Recognizable Sets with Multiplicities in the Tropical Semiring. In Michal Chytil, Ladislav Janiga, and Václav Koubek, editors, *Mathematical Foundations of Computer Science (MFCS)*, volume 324 of *Lecture Notes in Computer Science*, pages 107–120. Springer, 1988.

36 Johannes Waldmann. Weighted Automata for Proving Termination of String Rewriting. *Journal of Automata, Languages and Combinatorics*, 12(4):545–570, 2007.

37 Andreas Weber and Helmut Seidl. On the Degree of Ambiguity of Finite Automata. *Theoretical Computer Science*, 88(2):325–349, 1991.