



HAL
open science

Unambiguous Separators for Tropical Tree Automata

Thomas Colcombet, Sylvain Lombardy

► **To cite this version:**

Thomas Colcombet, Sylvain Lombardy. Unambiguous Separators for Tropical Tree Automata. 37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, Mar 2020, Montpellier, France. pp.32:1-32:13, 10.4230/LIPIcs.STACS.2020.32 . hal-03106897

HAL Id: hal-03106897

<https://hal.science/hal-03106897v1>

Submitted on 13 Jan 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.


Unambiguous Separators for Tropical Tree Automata*

Thomas Colcombet 

IRIF, CNRS, Université de Paris, Paris, France

<https://www.irif.fr/~colcombe/>

thomas.colcombet@irif.fr

Sylvain Lombardy 

LaBRI, Institut Polytechnique de Bordeaux – Université de Bordeaux – CNRS, France

<https://www.labri.fr/perso/slombard/>

sylvain.lombardy@labri.fr

Abstract

In this paper we show that given a max-plus automaton (over trees, and with real weights) computing a function f and a min-plus automaton (similar) computing a function g such that $f \leq g$, there exists effectively an unambiguous tropical automaton computing h such that $f \leq h \leq g$.

This generalizes a result of Lombardy and Mairesse of 2006 stating that series which are both max-plus and min-plus rational are unambiguous. This generalization goes in two directions: trees are considered instead of words, and separation is established instead of characterization (separation implies characterization). The techniques in the two proofs are very different.

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

Keywords and phrases Tree automata, Tropical semiring, Separation, Unambiguity

Digital Object Identifier 10.4230/LIPIcs.STACS.2020.32

Funding *Thomas Colcombet*: Supported by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No.670624), and the DeLTA ANR project (ANR-16-CE40-0007).

Acknowledgements The authors are grateful to the anonymous referees whose remarks and comments have allowed to improve this paper.

1 Introduction

Tropical automata is a nickname for weighted automata (automata parameterized by a semiring as introduced by Schützenberger [17]) over a tropical semiring. This is a particularly simple model of finite state automata that describe functions rather than languages. It exists in two forms, max-plus and min-plus automata. Essentially, a tropical automaton \mathcal{A} is a non-deterministic automaton for which each transition is labelled by a real weight (or an integer, or a natural number, depending on the variants). This weight is extended into a weight for a run: the sum of the weights of the transitions involved. A max-plus automaton computes the function $\llbracket \mathcal{A} \rrbracket : A^* \rightarrow \mathbb{R} \uplus \{\perp\}$ which to an input word associates the maximum weight of an accepting run over the input, or \perp if there is no accepting runs. If it is a min-plus automaton, minimum is used instead of maximum.

The use of tropical automata arises naturally in different contexts: max-plus automata have been used for modeling scheduling constraints (see for instance [4]) or worst case behaviors (see for instance [3] for computing the asymptotic worst case execution time

* The authors are committed to making professional choices acknowledging the climate emergency. We submitted this work to STACS for its excellence and because its location induces for us a low carbon footprint.



© Thomas Colcombet and Sylvain Lombardy;

licensed under Creative Commons License CC-BY

37th International Symposium on Theoretical Aspects of Computer Science (STACS 2020).

Editors: Christophe Paul and Markus Bläser; Article No. 32; pp. 32:1–32:13

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



of loops under the size-change abstraction); min-plus automata are used for optimisation questions (these are for instance used as a key tool in the decision of the star-height problem [6]). In all these situations, non-trivial decision procedures are used ([5, 12, 2]).

The starting point of this work is a result from 2006 of Lombardy and Mairesse:

► **Theorem 1** ([13, 14]). *A map $f: A^* \rightarrow \mathbb{R} \uplus \{\perp\}$ which is both definable by a min-plus and by a max-plus automaton is definable by an unambiguous tropical automaton.*

Recall that an automaton is unambiguous if there is at most one accepting run per input¹. Unambiguous automata form a very particular class of tropical automata. Most of the problems which are open or undecidable for general tropical automata are easily decidable for unambiguous automata: equivalence with another tropical automaton [9], boundedness, existence of an equivalent deterministic automaton, description of the asymptotic behaviour [1].

It is noteworthy that the decision algorithm to decide whether there exists an equivalent automaton actually applies to unambiguous automata and that algorithms described for larger classes (finitely or polynomially ambiguous), consist indeed in deciding first whether the tropical automaton is equivalent to some unambiguous one [8, 7].

The above Theorem 1 belongs to a fascinating corpus of mathematical statements of the form ‘if X belongs both to class C and to class D , then it belongs to class E ’, where E is structurally simpler than both C and D (often D is some form of dual of C). An archetypical example arises in descriptive set theory: Suslin’s theorem states that

if a set is analytic and coanalytic, it is Borel.

Many other instances of this pattern exist. For instance in automata theory, if an infinite tree language is Büchi and its complement is Büchi, it is weak (Rabin’s theorem [15]). This extends to cost-functions over infinite trees: if a cost-function over infinite trees is both B-Büchi and S-Büchi, it is quasi-weak ; over infinite words, it is even weak (Kuperberg and Vanden Boom [10, 11]). For languages of infinite words beyond regular, if a language is ωB and ωS definable, then it is ω -regular (Skrzypczak [18]). In language theory, a language which is both Σ_2 and Π_2 definable is definable in the two variables fragment (Thérien and Wilke [19]). Also, a language which is both the support and the complement of the support of a rational series over a field is regular [16]. This list continues on and on.

In many situations such statements arise in fact from a more general result of ‘separation’ (or of ‘interpolation’ in the logical terminology). For instance, Lusin’s theorem is the separation version of Suslin’s theorem: It states that

for $X \subseteq Y$ with X analytic and Y coanalytic, then $X \subseteq Z \subseteq Y$ for some Borel set Z .

Such separation results imply the characterization version. For instance, Suslin’s result follows from Lusin’s theorem: take $X = Y$ to be the set which is both analytic and coanalytic. Then $X \subseteq Z \subseteq Y = X$ for Z Borel; hence X is Borel. This relationship is general. The results of Rabin, Vanden Boom and Kuperberg, and Skrzypczak, for instance, exist in a ‘separation variant’.

¹ Note that when a tropical automaton is unambiguous, it makes no difference whether it is a max-plus or a min-plus automaton: It computes the same function.

Contribution

The natural question that we answer in this work is thus:

Does there exist a separation version of Theorem 1 ?

In this paper, we provide a positive answer to this question. It takes the following form:

► **Theorem 2** (separation for tropical tree automata). *Given a max-plus automaton \mathcal{A}_{\max} and a min-plus automaton \mathcal{A}_{\min} such that²*

$$\llbracket \mathcal{A}_{\max} \rrbracket \leq \llbracket \mathcal{A}_{\min} \rrbracket ,$$

there exists effectively an unambiguous tropical automaton \mathcal{A}_{sep} such that

$$\llbracket \mathcal{A}_{\max} \rrbracket \leq \llbracket \mathcal{A}_{\text{sep}} \rrbracket \leq \llbracket \mathcal{A}_{\min} \rrbracket .$$

Let us stress that the above theorem is established in the context of tropical automata over trees. Theorem 1 is now a corollary. Indeed, (a) tropical word automata are a particular case of tree automata over a ranked alphabet made of unary symbols only, plus a constant, and (b) assuming that f is both accepted by a min-plus and by a max-plus automaton, then by Theorem 2, there exists a function h accepted by an unambiguous tropical automaton such that $f \leq h \leq f$. Thus $f = h$ is accepted by an unambiguous tropical automaton.

Note that, though the result is a generalization, the proof of Theorem 2 is very different from the original one of Theorem 1.

Let us finally emphasize that particular care has been taken in order to obtain the result for real weights. Indeed, in the integer case (and as a consequence in the rational case), simpler techniques can be used that involve keeping in the finitely many states of the result automaton some explicit differences of partial weights up to a certain bound. Such a technique (as far as we know) cannot be used in the real case. Our results are effective for real weights as far as there exist an effective representation of the reals in the additive group generated by weights of automata \mathcal{A}_{\max} and \mathcal{A}_{\min} , as well as algorithms that compute the addition, the subtraction, and the comparison on these representations.

Other Related Work

The class of unambiguous tropical automata form an interesting subclass of tropical automata. In particular, equivalence is decidable, while the problem for max-plus or min-plus automata is undecidable [9]. Given a tropical automaton, deciding unambiguity is an open problem. It has been solved when the input automaton is finitely ambiguous in [8], and when it is polynomially ambiguous in [7].

The approach used in this paper is completely different from the original result of [14]. Crossing reachable and productive states refers to technics that have been used since Hashiguchi's papers on limitedness of tropical automata [5], but the basement of our proof is the original pumping Lemma 11.

Structure of the Paper

This paper is organized as follows. In Section 2, we recall the standard definitions concerning trees, automata over trees, and tropical automata. In Section 3, we establish our main theorem of separation, Theorem 2. Section 4 concludes.

² In this statement, we assume that \perp is incomparable with other elements, and thus $\llbracket \mathcal{A}_{\max} \rrbracket$ and $\llbracket \mathcal{A}_{\min} \rrbracket$ are equal to \perp on the same words: they have same support.

2 Definitions

We review in this section classical notions concerning terms, automata, and tropical automata.

2.1 Terms and Contexts

A *ranked alphabet* is a set A , the elements of which are called *letters*, together with a map rank from A to \mathbb{N} . For $n \in \mathbb{N}$, let $\text{Terms}(n)$ be the set of *terms of arity n* over the alphabet $A \cup \{1, \dots, n\}$ in which $1, \dots, n$ are seen as special letters of rank 0 that are used exactly once in each term. We call simply *terms* the terms of arity 0, and the set of terms is simply denoted Terms . We call *context* the terms of arity 1, and the set of contexts is simply denoted Contexts . Note that each letter a of rank n can naturally be seen as a term of arity n consisting solely of a root labelled a and children $1, \dots, n$. The *nodes* of a term of arity n , $\text{Nodes}(t)$ is the set of positions of the letters in the term. The *root node* is denoted root . A node labelled i for $i = 1 \dots n$ is called the *i th-hole*. The nodes that are not holes are called *inner nodes*. Given a node $x \in \text{Nodes}(t)$, $t(x)$ denotes the letter it carries. Given a letter of rank n and terms t_0, \dots, t_{n-1} , we denote by $a(t_0, \dots, t_{n-1})$ the term that has a as root, and as children from left to right t_0, \dots, t_{n-1} . The *height of a term s* , denoted $\text{height}(s)$, is the longest length of a branch, for the standard meaning of a branch. The *size of a term s* , denoted $\text{size}(s)$, is the number of nodes it has. Finally, given c a context and t a term (resp. t another context), we denote $c \circ t$ the term (resp. the context) obtained by plugging the root of t in the hole of c .

2.2 Automata

A *non-deterministic (tree) automaton* (or simply an automaton) has a finite set of *states* Q , an input ranked alphabet A , a *set of final states* F , and a *transition relation* Δ that consists of tuples of the form $(p_0, \dots, p_{n-1}, a, q)$ in which $a \in A$ is a letter of rank n , and p_0, \dots, p_{n-1}, q are states from Q .

A *run of the automaton* over a term t of arity n is a map ρ from $\text{Nodes}(t)$ to Q such that for all inner nodes $x \in \text{Nodes}(t)$ of children x_0, \dots, x_{n-1} , $(\rho(x_0), \dots, \rho(x_{n-1}), t(x), \rho(x)) \in \Delta$. We shall write $\tilde{\rho}(x)$ for this transition. An *accepting run* is a run of the automaton such that $\rho(\text{root}) \in F$. Given a term t , t is *accepted by the automaton* if there exists an accepting run of the automaton over t . The set of terms that are accepted is the *language accepted by the automaton*. We slightly refine the terminology for easier use. Over a term, a *run to state q* is a run that assumes state q at the root. Over a context, a *run from state p to state q* signifies that the state assumed in the hole is p , and the one assumed at the root is q . An *accepting run from p* is a run from p to q for a final state q .

An automaton is *unambiguous* if for all input terms t , there exists at most one accepting run over it. Said differently, for all input terms t , either there are no accepting runs over it, and the term is not accepted, or there is exactly one accepting run, and the term is accepted.

An *automaton with weights*³ \mathcal{A} is a non-deterministic automaton together with a real *weight* for all transitions and all final states, i.e. a map weight from $\Delta \uplus F$ to \mathbb{R} . Given a run ρ of the automaton, the *weight of the run* $\text{weight}(\rho)$ is the sum of the weights of $\tilde{\rho}(x)$ for

³ This is not a weighted automaton, which is parametrized by a semiring and not a monoid. This definition serves here just for holding the structure of our tropical automata irrespective of whether these are min-plus or max-plus.

ranging over the inner nodes of t . Given an accepting run ρ of the automaton, the *weight of the accepting run* $\text{weight}^{\text{acc}}(\rho)$ is the sum of the weight of the run and $\text{weight}(\rho(\text{root}))$.

Tropical automata refer in this work to one of two forms of automata: min-plus automata and max-plus automata defined as follows. A *min-plus automaton* \mathcal{A} is an automaton with weights that computes a function:

$$\llbracket \mathcal{A} \rrbracket_{\min} : \text{Terms} \longrightarrow \mathbb{R} \uplus \{\perp\}$$

$$t \longmapsto \begin{cases} \perp & \text{if there are no accepting runs of } \mathcal{A} \text{ over } t, \\ \min\{\text{weight}^{\text{acc}}(\rho) \mid \rho \text{ accepting run of } \mathcal{A} \text{ over } t\} & \text{otherwise,} \end{cases}$$

in which \perp is a symbol that we understand as ‘undefined’ (it appears classically as an absorbing element for $+$ which is larger than all $x \in \mathbb{R}$, i.e., the zero of the tropical semiring). A *max-plus automaton* is defined in an identical manner, but the semantics $\llbracket \mathcal{A} \rrbracket_{\max}$ is now defined using \max instead of \min . Since it is always clear from the context, we denote simply by $\llbracket \mathcal{A} \rrbracket$ either $\llbracket \mathcal{A} \rrbracket_{\min}$ or $\llbracket \mathcal{A} \rrbracket_{\max}$ depending on whether \mathcal{A} has been declared as a min-plus or as a max-plus automaton.

An *unambiguous tropical automaton* \mathcal{A} is a tropical automaton that has an unambiguous underlying automaton. Note that in this case, $\llbracket \mathcal{A} \rrbracket_{\max} = \llbracket \mathcal{A} \rrbracket_{\min}$, and hence we call it simply tropical automaton and do not have to specify whether it is min-plus or max-plus.

3 Separating Tropical Automata

3.1 Statement and Structure of the Proof

The goal of this section is to prove our main theorem:

► **Theorem 2** (separation for tropical tree automata). *Given a max-plus automaton \mathcal{A}_{\max} and a min-plus automaton \mathcal{A}_{\min} such that⁴*

$$\llbracket \mathcal{A}_{\max} \rrbracket \leq \llbracket \mathcal{A}_{\min} \rrbracket ,$$

there exists effectively an unambiguous tropical automaton \mathcal{A}_{sep} such that

$$\llbracket \mathcal{A}_{\max} \rrbracket \leq \llbracket \mathcal{A}_{\text{sep}} \rrbracket \leq \llbracket \mathcal{A}_{\min} \rrbracket .$$

From now on, we fix the ranked alphabet A , a max-plus automaton \mathcal{A}_{\max} and a min-plus automaton \mathcal{A}_{\min} :

$$\mathcal{A}_{\max} = (Q_{\max}, A, F_{\max}, \Delta_{\max}, \text{weight}_{\max}) \text{ and } \mathcal{A}_{\min} = (Q_{\min}, A, F_{\min}, \Delta_{\min}, \text{weight}_{\min})$$

such that

$$\llbracket \mathcal{A}_{\max} \rrbracket \leq \llbracket \mathcal{A}_{\min} \rrbracket .$$

It will be convenient in what follows to consider a single automaton with weights constructed as the disjoint union of \mathcal{A}_{\max} and \mathcal{A}_{\min} (of course, it should be neither seen as a min-plus

⁴ In this statement, we assume that \perp is incomparable with other elements, and thus $\llbracket \mathcal{A}_{\max} \rrbracket$ and $\llbracket \mathcal{A}_{\min} \rrbracket$ are equal to \perp on the same words: they have same support.

automaton nor as a max-plus automaton). Formally, we assume without loss of generality that Q_{\max} and Q_{\min} are disjoint, and we set this automaton $\mathcal{A} = (Q, A, F, \Delta, \text{weight})$, with

$$Q = Q_{\min} \cup Q_{\max}, \quad F = F_{\max} \cup F_{\min}, \quad \Delta = \Delta_{\max} \cup \Delta_{\min},$$

$$\text{and } \text{weight}(v) = \begin{cases} \text{weight}_{\max}(v) & \text{for } v \in \Delta_{\max} \uplus F_{\max} \\ \text{weight}_{\min}(v) & \text{otherwise.} \end{cases}$$

The rest of this section is devoted to the proof of Theorem 2, and is organized as follows. In Section 3.2, we use some classical automata constructions for accessing in an unambiguous manner the reachable and productive states (Lemma 3). The combinatorial core of the proof is contained in Section 3.3 in which we study how the values of the automata may evolve in a context (Lemma 9), and use it for showing how terms can be substituted for smaller ones while preserving separability (Corollary 12). We finally provide the construction of the automaton \mathcal{A}_{sep} in Section 3.4, and establish its correctness (Lemma 15). This concludes the proof of Theorem 2.

3.2 Reachable and Productive States

An ingredient which is necessary in the proof is that the automaton we construct is always ‘aware’ of what are the states that may lead to an accepting run to the root. This section is concerned with this aspect, and involves only completely standard techniques for tree automata.

Given a term t , set $\text{Reach}(t) \subseteq Q$ to be the set of states p such that there is a run over t to p . We call such states *reachable in t* . Given a context c , set $\text{Prod}(c) \subseteq Q$ to be the set of states p such that there is an accepting run from p . We call such states *productive in c* . We finally set

$$\text{Reachable} = \{\text{Reach}(t) \mid t \in \text{Terms}\} \quad \text{and} \quad \text{Productive} = \{\text{Prod}(c) \mid c \in \text{Contexts}\}.$$

We describe the construction of an automaton $\mathcal{A}_{\text{pro}} = (Q_{\text{pro}}, A, F_{\text{pro}}, \Delta_{\text{pro}})$ that computes the productive states at each node of a term. The states are $Q_{\text{pro}} = \text{Reachable} \times \text{Productive}$. The final states $F_{\text{pro}} = \text{Reachable} \times \{F\}$, and for all letters a of rank n , the automaton has a transition of the form

$$((R_0, P_0), \dots, (R_{n-1}, P_{n-1}), a, (R, P)) \in \Delta_{\text{pro}}$$

whenever

- $R = \{r \in Q \mid (r_0, \dots, r_{n-1}, a, r) \in \Delta, r_j \in R_j \text{ for all } j\}$, and
 - $P_i = \{r_i \in Q \mid (r_0, \dots, r_{n-1}, a, p) \in \Delta, r_j \in R_j \text{ for } j \neq i, p \in P\}$ for all $i = 0 \dots n - 1$.
- In the above definition, the constraint on R induces the computation in a bottom-up deterministic way of the set of states that are reachable from the term below. The constraint on P_i computes similarly in a top-down deterministic way the set of states that are productive in the context above. We do not prove the correctness of this construction further. The important aspects of this construction are summarized in the following lemma.

► **Lemma 3.** *For all $P \in \text{Productive}$ and all terms t , there exists one and one only run of \mathcal{A}_{pro} over t to a state of the form (R, P) for some $R \in \text{Reachable}$. And furthermore, $R = \text{Reach}(t)$.*

For all $R \in \text{Reachable}$ and all contexts c , there exists one and one only accepting run of \mathcal{A}_{pro} over c from a state of the form (R, P) for some $P \in \text{Productive}$. And furthermore, $P = \text{Prod}(c)$.

3.3 The Central Pumping Lemma

In this section, we establish the key Corollary 12. The central concept here is to understand what it does for the value computed by \mathcal{A}_{\max} and by \mathcal{A}_{\min} to substitute a subtree for another subtree. And more precisely, we devise sufficient conditions such that, after performing the substitution, the values of the two automata gets closer one to the other, up to some shifting. This property is expressed in Lemma 5.

The key definition involved is the one of refinement with shift as defined now.

► **Definition 4.** *Given two terms s, t , some set $P \subseteq Q$, and some real number x , then t refines s for P with shift x if*

- $\text{Reach}(s) = \text{Reach}(t)$,
- for all runs ρ of \mathcal{A}_{\max} over s to a state $p \in P$, there is a run ρ' over t to state p such that

$$\text{weight}(\rho) \leq \text{weight}(\rho') + x$$

and

- for all runs τ of \mathcal{A}_{\min} over s to a state $q \in P$, there is a run τ' over t to state q such that

$$\text{weight}(\tau') + x \leq \text{weight}(\tau)$$

The justification of this definition is given by the following lemma. It shows how substituting s for t in a context when t refines s with some shift is done while ‘staying in the separation interval’.

► **Lemma 5.** *Let c be a context, and s, t be terms such that t refines s for $\text{Prod}(c)$ with shift x , then*

$$\llbracket \mathcal{A}_{\max} \rrbracket (c \circ s) \leq \llbracket \mathcal{A}_{\max} \rrbracket (c \circ t) + x \leq \llbracket \mathcal{A}_{\min} \rrbracket (c \circ t) + x \leq \llbracket \mathcal{A}_{\min} \rrbracket (c \circ s) .$$

Proof. Let ρ be an accepting run of \mathcal{A}_{\max} over $c \circ s$. It can be decomposed as an accepting run ρ_c over c from some state p and a run ρ_s over s to state p . The run ρ_c is a witness that $p \in \text{Prod}(c) \cap Q_{\max}$. Hence, since t refines s for $\text{Prod}(c)$ with shift x , there exists a run ρ_t over t to state p such that $\text{weight}(\rho_s) \leq \text{weight}(\rho_t) + x$. By gluing ρ_t with ρ_c , we obtain a new accepting run ρ' of \mathcal{A}_{\max} over $c \circ t$, furthermore,

$$\begin{aligned} \text{weight}^{\text{acc}}(\rho) &= \text{weight}^{\text{acc}}(\rho_c) + \text{weight}(\rho_s) \\ &\leq \text{weight}^{\text{acc}}(\rho_c) + \text{weight}(\rho_t) + x = \text{weight}^{\text{acc}}(\rho') + x . \end{aligned}$$

Since for all ρ there exists such a ρ' , we obtain

$$\llbracket \mathcal{A}_{\max} \rrbracket (c \circ s) \leq \llbracket \mathcal{A}_{\max} \rrbracket (c \circ t) + x .$$

The middle inequality simply comes from the key assumption $\llbracket \mathcal{A}_{\max} \rrbracket \leq \llbracket \mathcal{A}_{\min} \rrbracket$ in Theorem 2.

The third inequality is established as the first one (it is symmetric). ◀

The two following facts are straightforward to verify.

► **Fact 6** (reflexivity of refinement with shift). *For all terms s , and all $P \subseteq Q$, s refines s for P with shift 0.*

► **Fact 7** (transitivity of refinement with shift). *If t refines s for P with shift x , and u refines t for P with shift y , then u refines s for P with shift $x + y$.*

The next lemma is also purely mechanical.

► **Lemma 8** (refinement with shift is a congruence). *Let $((P_0, R_0), \dots, (P_{n-1}, R_{n-1}), a, (P, R))$ be in Δ_{pro} , and for all $i = 0 \dots n-1$, let t_i, s_i be terms such that*

- $\text{Reach}(t_i) = R_i$, and
 - t_i refines s_i for P_i with shift x_i ,
- then $a(t_0, \dots, t_{n-1})$ refines $a(s_0, \dots, s_{n-1})$ for P with shift $x_0 + \dots + x_{n-1}$.

Proof. Let ρ be a run of \mathcal{A}_{max} over $a(t_0, \dots, t_{n-1})$ to state $p \in P$. The run ρ can be decomposed into a transition $(p_0, \dots, p_{n-1}, a, p)$ of weight x at the root, and a run ρ_i of \mathcal{A}_{max} over t_i to p_i for all $i = 0 \dots n-1$. For all $i = 0 \dots n-1$, $p_i \in \text{Reach}(t_i) = R_i$. Since furthermore $p \in P$ and $((P_0, R_0), \dots, (P_{n-1}, R_{n-1}), a, (P, R)) \in \Delta_{\text{pro}}$, we obtain that $p_i \in P_i$ for all $i = 0 \dots n-1$ (second item of the definition of Δ_{pro}). Thus, since t_i refines s_i for P_i with shift x_i , there exists a run ρ'_i of \mathcal{A}_{max} over t_i to p_i such that $\text{weight}(\rho_i) \leq \text{weight}(\rho'_i) + x_i$. We can combine all these runs ρ'_i together with the transition $(p_0, \dots, p_{n-1}, a, p)$ and obtain a new run ρ' of \mathcal{A}_{max} over $a(t_0, \dots, t_{n-1})$ to p such that

$$\begin{aligned} \text{weight}(\rho) &= \text{weight}(\rho_0) + \dots + \text{weight}(\rho_{n-1}) + x \\ &\leq \text{weight}(\rho'_0) + x_0 + \dots + \text{weight}(\rho'_{n-1}) + x_{n-1} + x \\ &= \text{weight}(\rho') + x_0 + \dots + x_{n-1} . \end{aligned}$$

This shows half of the fact that $a(t_0, \dots, t_{n-1})$ refines $a(s_0, \dots, s_{n-1})$ for P with shift $x_0 + \dots + x_{n-1}$. The other half is symmetric. ◀

We aim now at proving Corollary 12 which states that all sufficiently large term is ‘shift refined’ by another one of smaller size. Beforehand, we need a pumping argument to establish:

► **Lemma 9.** *Let $P \in \text{Productive}$, $R \in \text{Reachable}$ and m be a context, then there exists a real number x such that*

- for every p in $Q_{\text{max}} \cap P \cap R$, for all runs ρ of \mathcal{A}_{max} over m from p to p , $\text{weight}(\rho) \leq x$, and
- for every q in $Q_{\text{min}} \cap P \cap R$, for all runs τ of \mathcal{A}_{min} over m from q to q , $x \leq \text{weight}(\tau)$.

Proof. Let t be a term such that $\text{Reach}(t) = R$, and c be a context such that $\text{Prod}(c) = P$.

▷ **Claim 10.** We claim first that for all runs ρ of \mathcal{A}_{max} over m from p to p with $p \in P \cap R$ and all runs τ of \mathcal{A}_{min} over m from q to q with $q \in P \cap R$, $\text{weight}(\rho) \leq \text{weight}(\tau)$.

Otherwise, there would exist some runs ρ, τ as above such that $\text{weight}(\rho) > \text{weight}(\tau)$. I.e.

$$\text{weight}(\tau) - \text{weight}(\rho) < 0 . \tag{*}$$

Consider now for all $n > 0$ the term:

$$u_n = c \circ \overbrace{m \circ \dots \circ m}^{n\text{-times}} \circ t .$$

Let ρ' be some accepting run over c from p (this is possible since $p \in P = \text{Prod}(c)$). Let τ' be some accepting run over c from q (this is possible since $q \in P = \text{Prod}(c)$). Let ρ'' be some run over t to p (this is possible since $p \in R = \text{Reach}(t)$). Let τ'' be some run over t to q (this is possible since $q \in R = \text{Reach}(t)$).

By concatenating ρ' , n -times ρ , and ρ'' , we obtain an accepting run ρ_n over u_n of weight $\text{weight}^{\text{acc}}(\rho_n) = \text{weight}^{\text{acc}}(\rho') + n \text{weight}(\rho) + \text{weight}(\rho'')$. Similarly, by concatenating τ' , n -times τ , and τ'' , we obtain an accepting run τ_n over u_n of weight $\text{weight}^{\text{acc}}(\tau_n) = \text{weight}^{\text{acc}}(\tau') + n \text{weight}(\tau) + \text{weight}(\tau'')$.

Furthermore, since $\llbracket \mathcal{A}_{\max} \rrbracket(u_n) \leq \llbracket \mathcal{A}_{\min} \rrbracket(u_n)$, $\text{weight}^{\text{acc}}(\rho_n) \leq \text{weight}^{\text{acc}}(\tau_n)$. We obtain

$$\begin{aligned} 0 &\leq \text{weight}^{\text{acc}}(\tau_n) - \text{weight}^{\text{acc}}(\rho_n) \\ &= \text{weight}^{\text{acc}}(\tau') + n \text{weight}(\tau) + \text{weight}(\tau'') - \text{weight}^{\text{acc}}(\rho') - n \text{weight}(\rho) - \text{weight}(\rho'') \\ &= \text{weight}^{\text{acc}}(\tau') + \text{weight}(\tau'') - \text{weight}^{\text{acc}}(\rho') - \text{weight}(\rho'') + n(\text{weight}(\tau) - \text{weight}(\rho)). \end{aligned}$$

However, using (\star) , this last quantity tends to $-\infty$ when n tends to ∞ . It contradicts its non-negativeness. The claim is established.

We can now establish the lemma. Let Y be the set of weights $\text{weight}(\rho)$ for ρ ranging over the runs of \mathcal{A}_{\max} over m from p to p with $p \in P \cap R$. Similarly, let Z be the set of weights $\text{weight}(\tau)$ for τ ranging over the runs of \mathcal{A}_{\min} over m from q to q with $q \in P \cap R$. The above claim states that for all $y \in Y$ and all $z \in Z$, $y \leq z$. This implies the existence of some real number x such that for all $y \in Y$, $y \leq x$, and for all $z \in Z$, $x \leq z$ (note that proving it requires to treat the case of Y and/or Z being empty, and thus requires a case distinction). This is exactly the statement of the lemma. \blacktriangleleft

Notice that a fixed context m admits only a finite number of runs; hence, the weights of paths involved in Lemma 9 can be enumerated and the value x be effectively computed.

► Lemma 11. *There exists a computable $k \in \mathbb{N}$ such that for all $P_0 \in \text{Reachable}$ and all terms s of height more than k , there exists effectively a term t such that t refines s for P_0 with some shift and $\text{size}(t) < \text{size}(s)$.*

Proof. Let k be $(4|Q|)^{|Q|}$. Let us fix a context d such that $\text{Prod}(d) = P_0$.

Consider now a term s of height larger than k and some $P_0 \in \text{Reachable}$. We aim at removing some piece of this term while achieving the conclusions of the lemma.

For all states $p \in P_0$, set ρ_p to be an optimal run of \mathcal{A} over s to p , i.e.,

- if $p \in Q_{\max}$, then for all runs τ of \mathcal{A}_{\max} over s to p , $\text{weight}(\tau) \leq \text{weight}(\rho_p)$, and
- if $p \in Q_{\min}$, then for all runs τ of \mathcal{A}_{\min} over s to p , $\text{weight}(\rho_p) \leq \text{weight}(\tau)$.

Since the longest branch of s has length greater than $2^{|Q|}2^{|Q|}|Q|^{|Q|}$, we can apply the pigeonhole principle to the various ways to split this branch in two, and get a factorisation of s into

$$s = c \circ m \circ s',$$

in which c is a context, m is a non-empty context, and s' is a term such that

- $\text{Reach}(s') = \text{Reach}(m \circ s')$; let R be this set;
- $\text{Prod}(d \circ c) = \text{Prod}(d \circ c \circ m)$; let P be this set;
- for all $p \in P_0$, there exists a state $q_p \in Q$ such that ρ_p is decomposed into a run τ_p over s' to q_p , a run τ'_p over m from q_p to q_p , and a run τ''_p over c from q_p to p .

Let us define now our term t as:

$$t = c \circ s'.$$

Since $s = c \circ m \circ s'$, our new term t is nothing but s in which the non-empty part corresponding to m has been removed. Hence $\text{size}(t) < \text{size}(s)$.

32:10 Unambiguous Separators for Tropical Tree Automata

We shall prove now that t refines s for P_0 with shift x where x is obtained by applying Lemma 9 to P, R and m .

Let ρ be a run of \mathcal{A}_{\max} over s to state p for some $p \in P_0$. We know that the run ρ_p as defined above is such that $\text{weight}(\rho) \leq \text{weight}(\rho_p)$. Finally, let ρ' be the run over $t = c \circ s'$ to p obtained by gluing τ_p and τ_p'' together. We have:

$$\begin{aligned} \text{weight}(\rho) &\leq \text{weight}(\rho_p) && \text{(by optimality of } \rho_p) \\ &\leq \text{weight}(\tau_p) + \text{weight}(\tau_p') + \text{weight}(\tau_p'') && \text{(decomposition of } \rho_p) \\ &\leq \text{weight}(\tau_p) + x + \text{weight}(\tau_p'') && \text{(by choice of } x \text{ and Lemma 9)} \\ &\leq \text{weight}(\rho') + x . && \text{(definition of } \rho') \end{aligned}$$

Hence, we have proved the first half of the definition of ‘ t refines s for P_0 with shift x ’. The second half is symmetric. Overall, we conclude that t refines s for P_0 with shift x . ◀

Using iteratively the above Lemma 11, as long as the height of the term is larger than k , together with Fact 6 and 7, we obtain the following corollary.

► **Corollary 12.** *There exists a computable $k \in \mathbb{N}$ such that for all $P \in \text{Reachable}$ and all terms s there exists effectively a term t of height at most k which refines s for P with some shift.*

3.4 The Construction

We are now ready to construct our separating automaton \mathcal{A}_{sep} . It is defined as follows:

$$\mathcal{A}_{\text{sep}} = (Q_{\text{sep}}, A, F_{\text{sep}}, \Delta_{\text{sep}}, \text{weight}_{\text{sep}}) ,$$

in which the set of states is

$$Q_{\text{sep}} = \{(R, P, t) \mid R \in \text{Reachable}, P \in \text{Productive}, \\ t \in \text{Terms}, \text{Reach}(t) = R, \text{height}(t) \leq k\} ,$$

(where k is the constant from Corollary 12), the final states, together with their weight, are

$$F_{\text{sep}} = \{(R, P, t) \in Q_{\text{sep}} \mid P = F, R \cap F \neq \emptyset\} \quad \text{with} \quad \text{weight}_{\text{sep}}(R, F, t) = \llbracket \mathcal{A}_{\max} \rrbracket(t) ,$$

and the transition relation and the weights are defined as follows. For a letter a of rank n , there is a transition of the form

$$\delta = ((R_0, P_0, t_0), \dots, (R_{n-1}, P_{n-1}, t_{n-1}), a, (R, P, t)) \in \Delta_{\text{sep}} \quad \text{with} \quad \text{weight}_{\text{sep}}(\delta) = x ,$$

whenever

- $((R_0, P_0), \dots, (R_{n-1}, P_{n-1}), a, (R, P))$ is a transition of Δ_{pro} .
- $(t, x) = \text{sr}_P(a(t_0, \dots, t_{n-1}))$, where sr is a map of the following form:

$$\begin{aligned} \text{sr}_P : \text{Terms} &\longrightarrow \text{Terms} \times \mathbb{R} \\ s &\longmapsto (t, x) \quad \text{such that } t \text{ refines } s \text{ for } P \text{ with shift } x. \end{aligned}$$

(Such a map exists thanks to Corollary 12.)

Notice first that R is a redundant information in the state (R, P, t) , since $R = \text{Reach}(t)$. The set P in the state ensures that the weights which are considered are really contributing to the run. If the constraint $\text{height}(t) \leq k$ was removed from the definition of Q_{sep} and sr_P was defined as $\text{sr}_P(s) = (s, 0)$, the automaton \mathcal{A}_{sep} would be an infinite unambiguous automaton equivalent to \mathcal{A}_{max} . Thanks to Lemma 11, one can bound the height of t in order to obtain an automaton that realizes a function which is larger than $\llbracket \mathcal{A}_{\text{max}} \rrbracket$ but smaller than $\llbracket \mathcal{A}_{\text{min}} \rrbracket$. The automaton is finite: the number of states is bounded by $2^{|Q|} a^{c^k}$, where a is the size of the alphabet, c is the maximal rank of letters, and k is the constant of Lemma 11, which is smaller than $(4|Q|)^{|Q|}$. This bound is obviously crude. In a practical implementation, an improvement can easily be made. It is not necessary to use all terms t of height up to k in Q_{sep} : it is sufficient to keep minimal ones for the shift refine relation for each $P \in \text{Productive}$.

Let us first note:

► **Lemma 13.** *For all $P \in \text{Productive}$ and all terms s , there exists exactly one run of \mathcal{A}_{sep} over s to a state of the form (R, P, t) .*

Proof. Indeed, we have seen in Lemma 3 that \mathcal{A}_{sep} is unambiguous on its first two components. Then the third component is computed in a bottom-up deterministic manner. Furthermore, it is easy to show by induction that on every input term there is an accepting run. ◀

► **Lemma 14.** *Let ρ be a run of \mathcal{A}_{sep} over s to (R, P, t) , then t refines s for P with shift $\text{weight}_{\text{sep}}(\rho)$.*

Proof. The proof is by induction on $\text{height}(s)$. Assume s of the form $a(s_0, \dots, s_{n-1})$. Let ρ be the run of \mathcal{A}_{sep} over s to (R, P, t) , let $\delta = ((R_0, P_0, t_0), \dots, (R_1, P_1, t_1), a, (R, P, t))$ be the transition assumed by ρ at the root. Let ρ_i be the run ρ restricted to the subterm s_i . By induction hypothesis, t_i refines s_i for P_i with shift $\text{weight}_{\text{sep}}(\rho_i)$. By Fact 7, $a(t_0, \dots, t_{n-1})$ refines s for P with shift $\text{weight}_{\text{sep}}(\rho_0) + \dots + \text{weight}_{\text{sep}}(\rho_{n-1})$. By definition of $\text{weight}_{\text{sep}}$, t refines $a(t_0, \dots, t_{n-1})$ with shift $\text{weight}_{\text{sep}}(\delta)$. By Fact 7, we obtain that t refines s with shift $\text{weight}_{\text{sep}}(\rho_0) + \dots + \text{weight}_{\text{sep}}(\rho_{n-1}) + \text{weight}_{\text{sep}}(\delta) = \text{weight}_{\text{sep}}(\rho)$. ◀

We can now provide the concluding lemma of the proof of Theorem 2.

► **Lemma 15.** $\llbracket \mathcal{A}_{\text{max}} \rrbracket \leq \llbracket \mathcal{A}_{\text{sep}} \rrbracket \leq \llbracket \mathcal{A}_{\text{min}} \rrbracket$.

Proof. Let s be a term. By Lemma 13, there exists one and exactly one run ρ_{sep} of \mathcal{A}_{sep} over s to a state of the form (R, F, t) . By Lemma 14, t refines s for F with shift $\text{weight}_{\text{sep}}(\rho_{\text{sep}})$. Note that in this case $R = \text{Reach}(s) = \text{Reach}(t)$.

Two cases can occur. If (R, F, t) is not final. In this case, there is no accepting run of \mathcal{A}_{sep} over s , and $\llbracket \mathcal{A}_{\text{sep}} \rrbracket(s) = \perp$. However, $(R, F, t) \notin F_{\text{sep}}$ means $\text{Reach}(t) \cap F = \emptyset$, hence $\text{Reach}(s) \cap F = \emptyset$. Thus $\llbracket \mathcal{A}_{\text{max}} \rrbracket(s) = \llbracket \mathcal{A}_{\text{min}} \rrbracket(s) = \perp$. We indeed have $\llbracket \mathcal{A}_{\text{max}} \rrbracket(s) \leq \llbracket \mathcal{A}_{\text{sep}} \rrbracket(s) \leq \llbracket \mathcal{A}_{\text{min}} \rrbracket(s)$.

Otherwise, (R, F, t) is final, i.e. $R \cap F \neq \emptyset$. Assume for instance that there is some $R \cap F \cap Q_{\text{max}} \neq \emptyset$ (it would be the same for Q_{min}). This means that $\llbracket \mathcal{A}_{\text{max}} \rrbracket(s) \neq \perp$. Since $\llbracket \mathcal{A}_{\text{min}} \rrbracket \geq \llbracket \mathcal{A}_{\text{max}} \rrbracket$, this implies also $\llbracket \mathcal{A}_{\text{min}} \rrbracket(s) \neq \perp$.

Let now ρ be an accepting run of \mathcal{A}_{max} over s of maximal value, and let p be its root state. Since t refines s for F with shift $\text{weight}_{\text{sep}}(\rho_{\text{sep}})$, and $p \in F$, there exists a run ρ'

over t to state p such that $\text{weight}(\rho) \leq \text{weight}(\rho') + \text{weight}_{\text{sep}}(\rho_{\text{sep}})$. Hence,

$$\begin{aligned} \llbracket \mathcal{A}_{\text{max}} \rrbracket(s) &= \text{weight}^{\text{acc}}(\rho) \\ &= \text{weight}(\rho) + \text{weight}_{\text{max}}(p) \\ &\leq \text{weight}(\rho') + \text{weight}_{\text{max}}(p) + \text{weight}(\rho_{\text{sep}}) \\ &\leq \llbracket \mathcal{A}_{\text{max}} \rrbracket(t) + \text{weight}(\rho_{\text{sep}}) \\ &= \llbracket \mathcal{A}_{\text{sep}} \rrbracket(s) \end{aligned}$$

In a symmetrical way, we obtain:

$$\begin{aligned} \llbracket \mathcal{A}_{\text{sep}} \rrbracket(s) &= \llbracket \mathcal{A}_{\text{max}} \rrbracket(t) + \text{weight}(\rho_{\text{sep}}) \\ &\leq \llbracket \mathcal{A}_{\text{min}} \rrbracket(t) + \text{weight}(\rho_{\text{sep}}) && \text{(assumption } \llbracket \mathcal{A}_{\text{max}} \rrbracket \leq \llbracket \mathcal{A}_{\text{min}} \rrbracket) \\ &\leq \llbracket \mathcal{A}_{\text{min}} \rrbracket(s) . && \text{(as for the other inequality)} \end{aligned}$$

Hence, we have established the expected $\llbracket \mathcal{A}_{\text{max}} \rrbracket(s) \leq \llbracket \mathcal{A}_{\text{sep}} \rrbracket(s) \leq \llbracket \mathcal{A}_{\text{min}} \rrbracket(s)$. \blacktriangleleft

4 Conclusion

We have established a separation result for tropical automata over trees.

All the results of this paper directly applies to automata on words. The proofs can be restated in this frameworks and are slightly easier, but their complexity actually comes from the fact that we want to encompass automata with (computable) real weights.

Our result is under the assumption that $\llbracket \mathcal{A}_{\text{max}} \rrbracket \leq \llbracket \mathcal{A}_{\text{min}} \rrbracket$. A natural variant is to invert the inequality and ask whether separation is possible when $\llbracket \mathcal{A}_{\text{min}} \rrbracket \leq \llbracket \mathcal{A}_{\text{max}} \rrbracket$. Some separation results exist in both variants (like interpolation results in logic), while some do not (separation of Büchi automata, or Lusin's theorem). For tropical automata, the assumption $\llbracket \mathcal{A}_{\text{min}} \rrbracket \leq \llbracket \mathcal{A}_{\text{max}} \rrbracket$ would be more complicated than the one in our theorem: it can be witnessed for instance by the fact that it is not decidable anymore [9].

Another interesting question is whether similar results hold for weights other than reals. For instance here, our proof requires for the weights of our automata to be equipped with a monoid structure, that it is commutative (otherwise weighted tree automata are not well defined), a total order (for the hypotheses of Theorem 2 to be meaningful), that the product be compatible with the order, and archimedianity (for the pumping argument in Lemma 9 to hold). The usefulness of each of these assumption could be studied. What if the monoid is not commutative (over words)? What if the order is not total (and be, for instance a lattice)? What if the operation is not archimedean (and what does it mean in these more general cases)? And in all these situations, do we capture interesting forms of automata?

More generally, these results of separation are fascinating, and it would be interesting to understand at high level what kind of abstract arguments may explain them, or at least some of them, uniformly.

References

- 1 Thomas Colcombet and Laure Daviaud. Approximate comparison of functions computed by distance automata. *Theory Comput. Syst.*, 58(4):579–613, 2016. doi:10.1007/s00224-015-9643-3.
- 2 Thomas Colcombet, Laure Daviaud, and Florian Zuleger. Size-change abstraction and max-plus automata. In *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part I*, pages 208–219, 2014. doi:10.1007/978-3-662-44522-8_18.

- 3 Thomas Colcombet, Laure Daviaud, and Florian Zuleger. Automata and program analysis. In *Fundamentals of Computation Theory - 21st International Symposium, FCT 2017, Bordeaux, France, September 11-13, 2017, Proceedings*, pages 3–10, 2017. doi:10.1007/978-3-662-55751-8_1.
- 4 Stéphane Gaubert and Jean Mairesse. Idempotency. In Jeremy Gunawardena, editor, *Idempotency*, volume 11 of *Publications of the Newton Institute*, chapter Task resource models and (max,+) automata, pages 133–144. Cambridge University Press, 1998.
- 5 Kosaburo Hashiguchi. Limitedness theorem on finite automata with distance functions. *J. Comput. Syst. Sci.*, 24(2):233–244, 1982. doi:10.1016/0022-0000(82)90051-4.
- 6 Kosaburo Hashiguchi. Algorithms for determining relative star height and star height. *Inf. Comput.*, 78(2):124–169, 1988. doi:10.1016/0890-5401(88)90033-8.
- 7 Daniel Kirsten and Sylvain Lombardy. Deciding unambiguity and sequentiality of polynomially ambiguous min-plus automata. In *26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009*, pages 589–600, 2009. doi:10.4230/LIPIcs.STACS.2009.1850.
- 8 Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theor. Comput. Sci.*, 327(3):349–373, 2004. doi:10.1016/j.tcs.2004.02.049.
- 9 Daniel Krob. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *IJAC*, 4(3):405–426, 1994. doi:10.1142/S0218196794000063.
- 10 Denis Kuperberg and Michael Vanden Boom. Quasi-weak cost automata: A new variant of weakness. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011*, pages 66–77, 2011. doi:10.4230/LIPIcs.FSTTCS.2011.66.
- 11 Denis Kuperberg and Michael Vanden Boom. On the expressive power of cost logics over infinite words. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, pages 287–298, 2012. doi:10.1007/978-3-642-31585-5_28.
- 12 Hing Leung. Limitedness theorem on finite automata with distance functions: An algebraic proof. *Theor. Comput. Sci.*, 81(1):137–145, 1991. doi:10.1016/0304-3975(91)90321-R.
- 13 Sylvain Lombardy and Jean Mairesse. Series which are both max-plus and min-plus rational are unambiguous. *ITA*, 40(1):1–14, 2006. doi:10.1051/ita:2005042.
- 14 Sylvain Lombardy and Jean Mairesse. Series which are both max-plus and min-plus rational are unambiguous. *arXiv e-prints*, page arXiv:0709.3257, September 2007. arXiv:0709.3257.
- 15 Michael O. Rabin. Weakly definable relations and special automata. *Mathematical Logic and Foundations of Set Theory*, pages 1–23, 1970.
- 16 Antonio Restivo and Christophe Reutenauer. On cancellation properties of languages which are support of rational series. *Journal of Computer System Sciences*, 29:153–159, 1984.
- 17 Marcel Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2-3):245–270, 1961. doi:10.1016/S0019-9958(61)80020-X.
- 18 Michał Skrzypczak. Separation property for wb- and ws-regular languages. *Logical Methods in Computer Science*, 10(1), 2014. doi:10.2168/LMCS-10(1:8)2014.
- 19 Denis Thérien and Thomas Wilke. Nesting until and since in linear temporal logic. *Theory Comput. Syst.*, 37(1):111–131, 2004. doi:10.1007/s00224-003-1109-3.