Size-Change Abstraction and Max-Plus Automata ⋆

Thomas Colcombet[1], Laure Daviaud[1], and Florian Zuleger[2]

[1] LIAFA, CNRS, Université Paris Diderot
[2] Vienna University of Technology

Abstract. Max-plus automata (over $\mathbb{N} \cup \{-\infty\}$) are finite devices that map input words to non-negative integers or $-\infty$. In this paper we present (a) an algorithm allowing to compute the asymptotic behaviour of max-plus automata, and (b) an application of this technique to the evaluation of the computational time complexity of programs.

1 Introduction

The contributions of this paper are two-fold. First, we provide an algorithm that given a function computed by a max-plus automaton over $\mathbb{N} \cup \{-\infty\}$ computes the asymptotic minimal behaviour of the automaton as a function of the length of the input. We then apply this result for characterizing the asymptotic complexity bounds that can be obtained by the size-change abstraction, which is a widely used technique in automated termination analysis. These two contributions are of independent interest. Let us introduce them successively.

Weighted automata, and the main theorem

Max-plus automata belong to the wider family of weighted automata, as introduced by Schützenberger [8]. The principle of weighted automata is to consider non-deterministic automata that produce values in a semiring $(S, \oplus, \otimes, 0, 1)$ (i.e., a ring in which the addition is not required to have an inverse). Weighted automata interpret the non-determinism of the automaton as the sum in the semiring and the sequence as the product. Standard non deterministic automata correspond to the case of the Boolean semiring $(\{0,1\}, \vee, \wedge, 0, 1)$. Probabilistic automata correspond to the case $([0,1], +, \times, 0, 1)$ (with a stochasticity restriction). Distance automata (or min-plus automata) correspond to the case $(\mathbb{N} \cup \{\infty\}, \min, +, 0, \infty\}$.

In this paper, we concentrate our attention to max-plus automata, which correspond to the semiring $(\mathbb{N} \cup \{-\infty\}, \max, +, 0, -\infty)$. Such automata have

⋆ The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°259454 and from the Vienna Science and Technology Fund (WWTF) through grant ICT12-059.

transition with weights in $\mathbb{N}$. Over a given input, they output the maximum over all accepting runs of the sum of the weights of transitions (and $-\infty$ if there is no accepting run). Such automata are natural candidates for modelling worst case behaviours of systems, as shown in the subsequent application. Remark that max-plus automata share a lot of common points with min-plus automata, and indeed, many results for max-plus automata can be converted into results for min-plus automata and vice-versa[3].

We seek to analyse the asymptotic behaviour of such automata. More precisely, fix a max-plus automaton computing a function f from the words in $\mathbb{A}^*$ to $\mathbb{N} \cup \{-\infty\}$. We study the asymptotic evolution of $c(n)$ defined for $n \in \mathbb{N}$ as:

$$c(n) = \inf\{f(w) \ : \ w \in \mathbb{A}^*, |w| \geq n\} \ .$$

We show that this quantity either is $-\infty$ for all n, or it is in $\Theta(n^\beta)$ for a computable rational $\beta \in [0, 1]$. Our main theorem, Theorem 2, expresses this property in a dual, yet equivalent, way as the asymptotic behaviour of the longest word that happens to have a value smaller than n.

From a logical perspective, it has to do with a quantifier alternation since the quantity studied is computed as a minimum (inf) of a function which, itself, is defined as a maximum (as a max-plus-automaton). In particular, in our case, it is immediately PSPACE hard (using reduction of the universality problem for non-deterministic automata). Such quantifier alternations are often even more complex when weighted automata are considered. For instance, a natural question involving such an alternation is to test whether $f(u) < |u|$ for some u, and it turns out to be undecidable [5]. On the other side, the boundedness question for min-plus automata (determining if there exists n such that $f(u) \leq n$ for all words u), which also has a similar quantifier alternation flavour, turns out to be decidable [4]. The work of Simon [10] has the most similarities with our contribution. It shows that, for a min-plus automaton computing a function g, the dual quantity $d(n) = \sup\{g(w) \ : \ w \in \mathbb{A}^*, |w| \leq n\}$ has a behaviour that is asymptotically between $n^{1/(k+1)}$ and $n^{1/k}$ for some non-negative integer k. Our result differs in two ways. First, the results for min-plus automata and for max-plus automata cannot be converted directly into results over the other form of automata. Second, our main result is significantly more precise since it provides the exact asymptotic coefficient. The proof of this theorem is the subject of the first part of this paper.

Program Analysis and Size Change Abstraction

The second contribution in this work consists in applying Theorem 2 for characterizing the asymptotic complexity bounds that can be obtained by the size-change abstraction, which is a popular program abstraction for automated ter-

[3] Indeed, if we allow negative weights, then negating all weights turns max-plus automata into min-plus automata and vice-versa, while preserving the semantics. However, such kind of reductions can get more complicated, if not impossible, when negative values are forbidden, as it is in our case.

mination analysis (e.g. [6, 7]). This question was the primary reason for this investigation.

We start with definitions needed to precisely state our contribution. We fix some finite set of *size-change variables* Var. We denote by Var' the set of primed versions of the variables Var. A *size-change predicate* (SCP) is a formula $x \triangleright y'$ with $x, y \in Var$, where $\triangleright$ is either $>$ or $\geq$. A *size-change transition* (SCT) T is a set of SCPs. A *size-change system* (SCS) $\mathcal{S}$ is a set of SCTs.

We define the semantics of size-change systems by *valuations* $\sigma : Var \to [0..N]$ of the size-change variables to natural numbers in the interval $[0..N]$, where N is a (symbolic) natural number. We write $\sigma, \tau' \models x \triangleright y'$ for two valuations σ, τ, if $\sigma(x) \triangleright \tau(y)$ holds over the natural numbers. We write $\sigma, \tau' \models T$, if $\sigma, \tau' \models x \triangleright y'$ holds for all $x \triangleright y' \in T$. A *trace* of an SCS $\mathcal{S}$ is a sequence $\sigma_1 \xrightarrow{T_1} \sigma_2 \xrightarrow{T_2} \cdots$ such that $T_i \in \mathcal{S}$ and $\sigma_i, \sigma'_{i+1} \models T_i$ for all i. The *length* of a trace is the number of SCTs that the trace uses, counting multiple SCTs multiple times. An SCS $\mathcal{S}$ is *terminating*, if $\mathcal{S}$ does not have a trace of infinite length.

We note that in earlier papers, e.g. [6], the definition of a size-change system includes a control flow graph that restricts the set of possible traces. For the ease of development we restrain from adding control structure but our result also holds when we add control structure. Moreover, earlier papers, e.g. [6], consider SCSs semantics over the natural numbers, i.e., valuations $\sigma : Var \to \mathbb{N}$. In contrast, we restrict values to the interval $[0, N]$ in order to guarantee that the length of traces is bounded for terminating SCSs: no valuation $\sigma \in Var \to [0..N]$ can appear twice in a trace (otherwise we would have a cycle, which could be pumped to an infinite trace); thus the length of traces is bounded by $(N+1)^k$ for SCSs with k variables.

Problem Statement: Our goal is to determine a function $h_{\mathcal{S}} : \mathbb{N} \to \mathbb{N}$ such that the length of the longest trace of a terminating SCS $\mathcal{S}$ is of asymptotic order $\Theta(h_{\mathcal{S}}(N))$. This question has also been of interest in a recent report [1], which claims that SCSs always have a polynomial bound, i.e., a bound $\Theta(N^k)$ for some $k \in \mathbb{N}$. However, this is not the case (see example below). We believe that the development in [1] either contains a gap or that the results of [1] have to be stated differently.

Example 1. The length of the longest trace of the SCS $\mathcal{S} = \{T_1, T_2, T_3\}$ with
$T_1 = \{x_1 > x'_1, x_2 \geq x'_2, x_3 > x'_3, x_4 \geq x'_4\}$,
$T_2 = \{x_1 > x'_1, x_2 \geq x'_2, x_2 \geq x'_3, x_2 > x'_4, x_3 > x'_4, x_4 > x'_4\}$ and
$T_3 = \{x_2 > x'_2, x_2 > x'_3, x_2 > x'_4, x_3 > x'_2, x_3 > x'_3, x_3 > x'_4, x_4 > x'_2, x_4 > x'_3, x_4 > x'_4\}$ is of asymptotic order $\Theta(N^{\frac{3}{2}})$. For comparison, [1] considers SCSs bounded in terms of the initial state; we can make $\mathcal{S}$ bounded in terms of the initial state by adding a new variable x_N to $\mathcal{S}$, and adding the constraints $\{x_N \geq x'_N, x_N \geq x'_1, x_N \geq x'_2, x_N \geq x'_3, x_N \geq x'_4\}$ to each of T_1, T_2, T_3.

The asymptotic order $\Theta(N^{\frac{3}{2}})$ of $\mathcal{S}$ can be established by Theorem 1 stated below (a corresponding max-plus automaton is stated in Example 2). For illustration purposes, we sketch here an elementary proof. For the lower bound we consider the sequence $s_N = ((T_1^{\frac{\sqrt{N}}{2}-1} T_2)^{\frac{\sqrt{N}}{2}-1} T_3)^{\frac{\sqrt{N}}{2}-1}$. For example, for

$N = 36$ we have $s_N = T_1T_1T_2T_1T_1T_2T_3T_1T_1T_2T_1T_1T_2T_3$. Note that s_N is of length $l_N = \frac{\sqrt{N}}{2} \cdot \frac{\sqrt{N}}{2} \cdot (\frac{\sqrt{N}}{2} - 1) = \Omega(N^{\frac{3}{2}})$. We define valuations σ_i, with $0 \leq i \leq l_N$, that demonstrate that s_N belongs to a trace of $\mathcal{S}$: given some index $0 \leq i \leq l_N$, let t_3 denote the number of T_3 before index i in the sequence s_N, let t_2 denote the number of T_2 before index i since the last T_3, and let t_1 denote the number of T_1 before index i since the last T_2 (note that we have $0 \leq t_1, t_2, t_3 < \frac{\sqrt{N}}{2}$ by the shape of s_N); we set $\sigma_i(x_1) = N - t_2 \cdot \frac{\sqrt{N}}{2} - t_1, \sigma_i(x_2) = N - t_3 \cdot \sqrt{N}, \sigma_i(x_3) = N - t_3 \cdot \sqrt{N} - t_1, \sigma_i(x_4) = N - t_3 \cdot \sqrt{N} - \frac{\sqrt{N}}{2} - t_2$. It is easy to verify that the valuations σ_i satisfy all constraints of s_N.

We move to the upper bound. Let S be a sequence of SCTs that belongs to a trace of $\mathcal{S}$. We decompose $S = S_1T_3S_2T_3\cdots$ into subsequences S_i that do not contain any occurrence of T_3. We define a_i to be the maximal number of consecutive T_1 in S_i, and b_i to be the total number of T_2 in S_i. We set $c_i = \max\{a_i, b_i\}$. We start with some observations: We have $|S_i| \leq c_i(c_i+1)+c_i = c_i(c_i+2)$ (i) by the definition of the c_i. We have $|S_i| \leq N$ (ii) because the inequality $x_1 > x_1'$ is contained in T_1 as well as in T_2 and the value of x_1 can only decrease N times in S_i. Combining (i) and (ii) we get $|S_i| \leq \min\{c_i(c_i+2), N\}$ (iii). We have $\sum_i c_i \leq N$ (iv); this holds because there is a chain of inequalities from the beginning to the end of S that for every i either uses all inequalities $x_3 > x_3'$ of the consecutive T_1 or all inequalities $x_4 > x_4'$ of the T_2 in S_i, and this chain can only contain N strict inequalities. Finally, by the definition of the S_i we have $|S| \leq \sum_i |S_i| + 1$. With (iii) we get $|S| \leq \sum_i \min\{c_i(c_i+2), N\} + 1 \leq 5\sum_i \min\{c_i^2, N\}$ (v). Using associativity and commutativity we rearrange the sum $\sum_i c_i = \sum_i d_i + \sum_i e_i + r$, where the d_i are summands $c_i > \sqrt{N}$ and the e_i and r are the sum of summands $c_i \leq \sqrt{N}$ with $\frac{\sqrt{N}}{2} \leq e_i \leq \sqrt{N}$ and $r < \frac{\sqrt{N}}{2}$; we denote $e_i = \sum_j c_{ij}$ for some c_{ij}. By (iv) there are at most $\sqrt{N}$ of the d_i and at most $2\sqrt{N}$ of the e_i. Using these definitions in (v) we get $|S| \leq 5(\sum_i \min\{d_i^2, N\} + \sum_{i,j} \min\{c_{ij}^2, N\} + \min\{r^2, N\}) \leq 5(\sqrt{N} \cdot N + \sum_{i,j} c_{ij}^2 + N) \leq 5(\sqrt{N} \cdot N + \sum_i e_i^2 + N) \leq 5(\sqrt{N} \cdot N + 2\sqrt{N} \cdot N + N) = O(N^{\frac{3}{2}})$.

In this paper we establish the fundamental result that the *computational time complexity* of terminating SCA instances is decidable:

Theorem 1. *Let $\mathcal{S}$ be a terminating SCS. The length of the longest trace of $\mathcal{S}$ is of order $\Theta(N^\alpha)$, where $\alpha \geq 1$ is a rational number; moreover, α is computable.*

We highlight that our result provides a *complete characterization* of the complexity bounds arising from SCA and gives means for determining the exact asymptotic bound of a given abstract program. Our investigation was motivated by previous work [11], where we introduced a practical program analysis based on SCA for computing resource bounds of imperative programs; in contrast to this paper, [11] does not study the completeness of the proposed algorithms and does not contain any result on the expressivity of SCA.

Organization of the Paper In Section 2, we give the automata definitions and sketch the proof of Theorem 2. In Section 3 we provide a reduction from

size-change systems to max-plus automata that allows to prove Theorem 1 from Theorem 2.

2 Max-Plus Automata

In this section, we first define max-plus automata (section 2.1), and then sketch the proof of Theorem 2 (section 2.2).

2.1 Definition of max-plus automata

A **semigroup** $(S, \cdot)$ is a set S equipped with an associative binary operation '$\cdot$'. If the product has furthermore a neutral element 1, $(S, \cdot, 1)$ is called a **monoid**. The monoid is said to be **commutative** if $\cdot$ is commutative. An **idempotent** in a semigroup is an element e such that $e \cdot e = e$. Given a subset A of a semigroup, $\langle A \rangle$ denotes the closure of A under product, *i.e.,* the least sub-semigroup that contains A. Given $X, Y \subseteq S$, $X \cdot Y$ denotes $\{a \cdot b \ : \ a \in X, \ b \in Y\}$.

A **semiring** $(S, \oplus, \otimes, 0_S, 1_S)$ is a set S equipped with two binary operations $\oplus$ and $\otimes$ such that $(S, \oplus, 0_S)$ is a commutative monoid, $(S, \otimes, 1_S)$ is a monoid, 0_S is absorbing for $\otimes$ (for all $x \in S$, $x \otimes 0_S = 0_S \otimes x = 0_S$) and $\otimes$ distributes over $\oplus$. We shall use the **max-plus semiring** $(\{-\infty\} \cup \mathbb{N}, \max, +, -\infty, 0)$, denoted $\overline{\mathbb{N}}$, and its extension $\overline{\mathbb{R}^+} = \{-\infty, 0\} \cup \{x \ : \ x \in \mathbb{R}, \ x \geq 1\}$, that we name the **real semiring**. This semiring will be used instead of $\overline{\mathbb{N}}$ during the computations. The operation over matrices induced by this semiring is denoted $\otimes$. Remark that $0_{\overline{\mathbb{N}}} = -\infty$, and $1_{\overline{\mathbb{N}}} = 0$.

Let S be a semiring. The set of matrices with m rows and n columns over S is denoted $\mathcal{M}_{m,n}(S)$, or simply $\mathcal{M}_n(S)$ if $m = n$. As usual, $A \otimes B$ for two matrices A, B (provided the width of A and the height of B coincide) is defined as:

$$(A \otimes B)_{i,j} = \bigoplus_{0<k\leq n} (A_{i,k} \otimes B_{k,j}) \quad \left(= \max_{0<k\leq n} (A_{i,k} + B_{k,j}) \text{ for } S = \overline{\mathbb{N}} \text{ or } \overline{\mathbb{R}^+} \right) .$$

It is standard that $(\mathcal{M}_n(S), \otimes, I_n)$ is a monoid, whose neutral element is the diagonal matrix I_n with 1_S (i.e., 0 for $\overline{\mathbb{N}}$) on the diagonal, and 0_S (i.e., $-\infty$ for $\overline{\mathbb{N}}$) elsewhere. For a positive integer k, we set $M^0 = I_n$, and $M^k = M^{k-1} \otimes M$. For $\lambda \in \mathbb{R}^+$, we denote by λA the matrix such that $(\lambda A)_{i,j} = \lambda A_{i,j}$ for all i, j (this matrix has non-negative real coefficients, which might not be over $\overline{\mathbb{R}^+}$ if $\lambda \leq 1$). Finally, we write $A \leq B$ if for all i, j, $A_{i,j} \leq B_{i,j}$.

A **max-plus automaton** over the alphabet $\mathbb{A}$ (with k states) is a map δ from $\mathbb{A}$ to $\mathcal{M}_k(\overline{\mathbb{N}})$ together with initial and final vectors $I, F \in \mathcal{M}_{1,k}(\{0, -\infty\})$. The map δ is uniquely extended into a morphism from $\mathbb{A}^*$ to $\mathcal{M}_k(\overline{\mathbb{N}})$, that we also denote δ. The **function computed by the automaton** maps each word $u \in \mathbb{A}^*$ to ${}^tI \otimes \delta(u) \otimes F \in \overline{\mathbb{N}}$ where ${}^tI$ denotes the transpose of I.

Example 2. We consider the following automaton, over the alphabet $\{a, b, c\}$, for $k = 6$ and defined by (where $-\infty$ is not written for readability):

$$\delta(a) = \begin{pmatrix} 0&0&0&0&0&0 \\ 1&&&&&0 \\ &0&&&&0 \\ &&1&&&0 \\ &&&0&0&0 \\ &&&&&0 \end{pmatrix}, \qquad \delta(b) = \begin{pmatrix} 0&0&0&0&0&0 \\ 1&&&&&0 \\ &0&0&1&0&0 \\ &&&&1&0 \\ &&&&1&0 \\ &&&&&0 \end{pmatrix},$$

$$\delta(c) = \begin{pmatrix} 0&0&0&0&0&0 \\ &&&&&0 \\ &1&1&1&0&0 \\ &1&1&1&0&0 \\ &1&1&1&0&0 \\ &&&&&0 \end{pmatrix}, \qquad \text{and } I = F = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

It is sometimes convenient to see such matrices as a weighted automaton [8]. Such a presentation is provided in Figure 1. The states of the automaton are $q_1, \ldots, q_6$ and correspond respectively to the lines and the columns 1 to 6 of the matrices. There is a transition from q_i to q_j corresponding to letter $x = a, b, c$ if the entry i, j of the matrix $\delta(x)$ is $z \neq -\infty$. In this case, the transition is weighted by z. The initial states are the states q_i such that $I_i = 0$. The final states are the states q_j such that $F_j = 0$. A run over the word w is a path (a sequence of compatible transitions) in the graph labelled by w. Its weight is the sum of the weights of the transitions. Finally the weight of a given word w is the maximum of the weights of the runs labelled by w and going from an initial state to a final state. The weight of w, given by the graph representation is exactly the value ${}^tI \otimes \delta(w) \otimes F$, given by the matrix presentation.

More details about weighted automata can be found in [3].

2.2 Main theorem

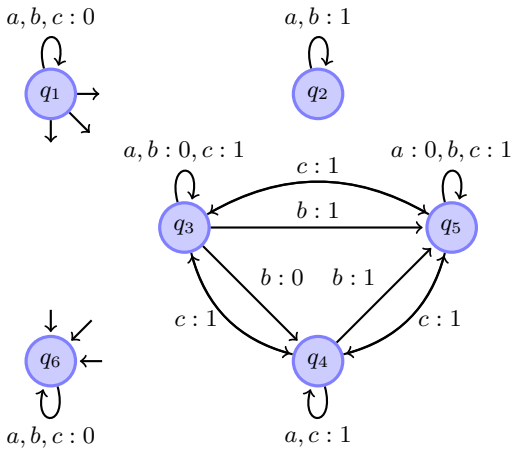
Theorem 2. *Given a max-plus automaton computing $f : \mathbb{A}^* \to \mathbb{N} \cup \{-\infty\}$, there exists an algorithm that computes the value $\alpha \in \{+\infty\} \cup \{\beta \in \mathbb{Q} \ : \ \beta \geq 1\}$ such that*

$$g(n) = \Theta(n^\alpha)$$

where $g(n) = \sup\{|w| \ : \ f(w) \leq n\}$, with the convention that $n^{+\infty} = +\infty$.

Example 3. The algorithm applied on the automaton given in exemple 2 outputs value 2/3. A sequence of words that witness this growth is $(((a^n b)^n)c^n)_{n\in\mathbb{N}}$.

The semigroup of weighted matrices Our goal is to analyse the relationship between the output of the automaton and the length of the input. Thus we



where:
– there are edges from state q_1 to every state labelled by every letter with weight 0,
– there are edges from every state to state q_6 labelled by every letter with weight 0,
– every state is initial and final.

Fig. 1. A weighted automaton over the semiring $(\overline{\mathbb{N}}, \max, +)$.

use weighted matrices that are pairs of a matrix representing the behaviour of the automaton with a value standing for the length of the input. Formally, a **weighted matrix** is an ordered pair (M, x) where $M \in \mathcal{M}_k(\overline{\mathbb{R}^+})$ and $x \geq 1$ is a real number called the **weight** of the weighted matrix. They are useful to represent pairs $(\delta(w), |w|)$. The set of weighted matrices is denoted by $\mathcal{W}_k$. Weighted matrices have a semigroup structure $(\mathcal{W}_k, \otimes)$, where $(M, x) \otimes (N, y)$ stands for $(M \otimes N, x + y)$. By definition, the function $w \mapsto (\delta(w), |w|)$ is a morphism of semigroups. As in the general case, we use $\otimes$ over subsets of $\mathcal{W}_k$. Given $A \subseteq \mathcal{W}_k$, $\langle A \rangle$ is the closure under $\otimes$ of A. Our goal is to study the set

$$\{(\delta(w), |w|) \mid w \in \mathbb{A}^*\} = \langle \{(\delta(a), 1) \mid a \in \mathbb{A}\} \rangle$$

and more precisely to give a finite representation of it up to some approximation. The key to our algorithm is the ability to (a) finitely represent infinite sets of weighted matrices and (b) define a notion of approximation between such sets. Then our algorithm computes using such sets, and guarantees that, up to the approximation, it is consistent with the behaviour of the automaton. We present these notions below. From now we fix a max-plus automaton with k states computing a function f and defined by the morphism δ. Let us first introduce another semiring useful for defining finite representation.

The $\overline{\mathbb{R}^+_\odot}$ and small semirings, and the semigroup of weighted matrices
We have seen the semirings $\overline{\mathbb{N}}$ and $\overline{\mathbb{R}^+}$. We use another semiring over the same ground set $\overline{\mathbb{R}^+}$ but with a different product, $\odot$. For all $x, y \in \overline{\mathbb{R}^+}$ set $x \odot y$ to be:

$$x \odot y = \begin{cases} -\infty & \text{if either } x = -\infty \text{ or } y = -\infty, \\ \max(x, y) & \text{otherwise.} \end{cases}$$

Again, $(\overline{\mathbb{R}^+}, \max, \odot, -\infty, 0)$ is a semiring, denoted $\overline{\mathbb{R}^+_\odot}$. As before, this induces a product operation $\odot$ for matrices. The product operation $\odot$ is a good approximation of $\otimes$ as shown by the following key lemma that follows from the similar property for real number and monotonicity of max and plus.

Lemma 1. *Given matrices $M_1, \ldots, M_q$, $q \geq 1$ over $\overline{\mathbb{R}^+}$, then*

$$M_1 \odot \cdots \odot M_q \leq M_1 \otimes \cdots \otimes M_q \leq q(M_1 \odot \cdots \odot M_q) \ .$$

The last semiring we use is the **small semiring** $(\mathbb{S}, \max, \odot, -\infty, 0)$, simply denoted $\mathbb{S}$, which is the restriction of $\overline{\mathbb{R}^+_\odot}$ to $\{-\infty, 0, 1\}$. There is a natural map φ from $\overline{\mathbb{R}^+}$ to $\mathbb{S}$ obtained by collapsing all elements above or equal to 1 to 1. It happens that φ is at the same time a **morphism** of semirings from $\overline{\mathbb{R}^+}$ to $\mathbb{S}$ and from $\overline{\mathbb{R}^+_\odot}$ to $\mathbb{S}$. Matrices over the small semiring are called **small matrices**.

The morphism φ is also extended to weighted matrices by $\varphi((M, x)) = \varphi(M)$.

Our goal is, given a finite set of weighted matrices A, to compute a presentation of $\langle A \rangle$ up to approximation (Lemma 7). The notion of presentation of sets of weighted matrices and the notion of approximation are the subject of the two subsequent sections.

Presentable Sets of Weighted Matrices We introduce now the notion of presentable sets of matrices, i.e., sets of matrices that we can manipulate via their finite presentation. Our sets of weighted matrices are presented in 'exponential form', i.e., given a weight $x \geq 1$, an entry of the matrix will be of the form x^α. In fact, some special cases have to be treated, that results in the use of $\alpha = \perp$ or $-\infty$.

Exponents and exponentiations The semiring of **exponents** (the choice of this name will be explained when defining exponentiation in the next paragraph) is $(\mathit{Exps}, \max, \max_\odot, \perp, -\infty)$ where

$$\mathit{Exps} = \{\perp, -\infty\} \cup [0, 1] \ ,$$

where max is defined with respect to the order $\perp < -\infty < x < y$ for all $x < y \in [0, 1]$, and where $\max_\odot(\alpha, \beta)$ for $\alpha, \beta \in \mathit{Exps}$ is defined by:

$$\max_\odot(\alpha, \beta) = \begin{cases} \perp & \text{if } \alpha = \perp \text{ or } \beta = \perp, \\ \max(\alpha, \beta) & \text{otherwise.} \end{cases}$$

This semiring will be simply denoted *Exps*, and the induced operation over matrices $\odot$ (we will see that this notation is not ambiguous). We take the convention to denote by α, β exponents, and by X, Y, Z vectors and matrices of exponents.

We define now the exponentiation operation. For $x \geq 1$ and $\alpha \in \mathit{Exps}$, set

$$x^\alpha = \begin{cases} -\infty & \text{if } \alpha = \perp, \\ 0 & \text{if } \alpha = -\infty, \\ x^\alpha & \text{otherwise, i.e., if } \alpha \in [0, 1], \text{ for the usual exponent.} \end{cases}$$

Lemma 2. *For all $x \geq 1$, $\alpha \mapsto x^\alpha$ is a semiring morphism from Exps to $\overline{\mathbb{R}^+_\odot}$.*

Note that this morphism can be applied to vectors (or matrices). In this case, given a matrix $Y \subseteq \mathit{Exps}^{k\times k}$, and some $x \geq 1$, we denote by $Y[x] \in \overline{\mathbb{R}^+}^{k\times k}$ the matrix such that $(Y[x])_{i,j} = x^{Y_{i,j}}$ for all $i, j = 1 \dots k$. According to the previous lemma, the map $Y \mapsto Y[x]$ is a morphism from matrices over *Exps* to matrices over $\overline{\mathbb{R}^+_\odot}$.

It is also sometimes convenient to send the small semiring to the exponent semiring. It is done using the following straightforward lemma.

Lemma 3. *The function γ that maps $-\infty$ to $\bot$, 0 to $-\infty$, and 1 to 0 is a semiring morphism from $\mathbb{S}$ to Exps such that $x^{\gamma(a)} = a$ for all $a \in \mathbb{S}$ and $x \geq 1$.*

Polytopes and presentable sets. Our goal it to describe finitely some infinite sets of matrices over $\overline{\mathbb{R}^+}$. We start from the notion of polytope. For this, we rely on the definition of polytopes in $\mathbb{R}^k$: a **polytope** (in $\mathbb{R}^k$) is a convex hull of finitely many points of $\mathbb{R}^k$. We would like to use this definition for subsets of Exps^k. For that we send *Exps* to $\mathbb{R}$ by $t(\bot) = -2$, $t(-\infty) = -1$ and $t(s) = s$ if s is real.

A subset of Exps^k is called a polytope if its image under t is a polytope in $\mathbb{R}^k$. In particular, we can use this definition for matrices of exponents, yielding polytopes of matrices.

We can now define presentable sets of matrices over $\overline{\mathbb{R}^+}$. Essentially, a set of matrices over $\overline{\mathbb{R}^+}$ is presentable if it is the image under exponentiation of a finite union of polytopes of exponent matrices. Let us define precisely how this is defined. A set of weighted matrices $A \subseteq \mathcal{W}_k$ is **presentable** if it is of the form:

$$A = \{(M, 1) \ : \ M \in S\} \cup \{(Y[x], x) \ : \ Y \in P, \ x \geq 1\} \ ,$$

where S is a set of small matrices of dimension $k \times k$, and P is a finite union of polytopes of $\mathit{Exps}^{k\times k}$. The pair (S, P) is called the **presentation** of A. A presentation is said **small** if $P = \emptyset$. It is said **asymptotic** if $S = \emptyset$. Obviously, any presentable set is the union of a set of small presentation with a set of asymptotic presentation. Of course presentable sets are closed under union.

The approximation and simulation scheme We describe now the notion of approximation that we use. Indeed, our goal is to compute the set of weighted matrices $\{(\delta(w), |w|)\}$. We cannot expect to do it in general, and, at any rate, presentable sets of matrices cannot capture exactly the behaviour of the automaton. That is why we reason about sets of matrices up to some approximation relation that is sufficiently precise for our purpose, and at the same time is sufficiently relaxed for allowing to approximate the behaviour of the automaton by a presentable set of weighted matrices.

Given some $a \geq 1$ and two weighted matrices (M, x) and (N, y), we write

$$(M, x) \preccurlyeq_a (N, y) \quad \text{if} \quad M \leq aN, \ y \leq ax \ \text{ and } \varphi(M) = \varphi(N) \ .$$

This definition extends to sets of weighted matrices as follows. Given two such sets A, B, $A \preccurlyeq_a B$ if for all $(N, y) \in B$, there exists $(M, x) \in A$ such that

$(M,x) \preccurlyeq_a (N,y)$. We write $A \approx_a B$ if $A \preccurlyeq_a B$ and $B \preccurlyeq_a A$ and say that A is a-**equivalent** to B. We drop the a parameter when not necessary, and simply write $A \approx B$ if $A \approx_a B$ for some a.

A first consequence of this definition is that every weighted matrix (M,x) is a-equivalent to the weighted matrix $(\varphi(M),1)$ where a is the maximum of the entries of M and x. This justifies that, in the definition of a presentable set, the weighted matrices of the finite part are of this form.

Let us give some intuition why this approximation may help. For instance consider some exponent matrix M, and let us show:

$$\{(M[x],x) \ : \ x \geq 1\} \approx_2 \{(M[y],y) \ : \ y \in \mathbb{N},\ y \geq 1\} \ .$$

Indeed, one inclusion is obvious, yielding $\preccurlyeq_1$. For the other direction, consider some $x \geq 1$, and take $y = \lfloor x \rfloor$, then $2y \geq x$ and $M[y] \leq M[x]$, thus $(M[y],y) \preccurlyeq_2 (M[x],x)$. More generally imagine the y's would be further constrained to be multiples of some value, say 2, then the same arguments would work. Hence this equivalence relation allows to absorb a certain number of phenomena that can occur in an automaton and are irrelevant for our specific problem. In particular, if the least growing rate is achieved for words of length n for n even only, then this 'computing modulo 2' can be 'hidden' thanks to the $\approx$-approximation.

The following lemma establishes some essential properties of the $\preccurlyeq_a$ relations (as a consequence, the same properties hold for $\approx_a$).

Lemma 4. *Given A, A', B, B', C sets of weighted matrices and $a, b \geq 1$,*

1. *if $A \preccurlyeq_a B$ and $b \geq a$, then $A \preccurlyeq_b B$,*
2. *if $A \preccurlyeq_a A'$ and $B \preccurlyeq_a B'$, then $A \cup B \preccurlyeq_a A' \cup B'$,*
3. *if $A \preccurlyeq_a B$ and $B \preccurlyeq_b C$ then $A \preccurlyeq_{ab} C$,*
4. *if $A \preccurlyeq_a A'$ and $B \preccurlyeq_a B'$ then $A \otimes B \preccurlyeq_a A' \otimes B'$,*
5. *if $A \preccurlyeq_a B$ then $\langle A \rangle \preccurlyeq_a \langle B \rangle$.*

The main induction: the forest factorization theorem of Simon The forest factorization theorem of Simon [9] is a powerful combinatorial tool for understanding the structure of finite semigroups. In this short abstract, we will not describe the original statement of this theorem, in terms of trees of factorizations, but rather a direct consequence of it which is central in our proof (the presentation of the theorem was used in a similar way in [2]).

Theorem 3 (equivalent to the forest factorization theorem [9]). *Given a semigroup morphism φ from $(S,\otimes)$ (possibly infinite) to a finite semigroup $(T,\odot)$, and some $A \subseteq S$, set $B_0 = A$ and for all $n \geq 0$,*

$$B_{n+1} = B_n \cup B_n \otimes B_n \cup \bigcup_{\substack{e \in T \\ \text{is idempotent}}} \langle B_n \cap \varphi^{-1}(e) \rangle \ ,$$

then $\langle A \rangle = B_N$ for $N = 3|T| - 1$.

This theorem teaches us that, for computing the closure under product in the semigroup S, it is sufficient to be able to know how to compute (a) the union of sets, (b) the product of sets, and (c) the restriction of a set to the inverse image of an idempotent by φ, and (d) the closure under product of sets of elements that all have the same idempotent image under φ. Of course, this proposition is only interesting when the semigroup T is cleverly chosen.

In our case, we are going to use the above proposition with $(S, \otimes) = (\mathcal{W}_k, \otimes)$, and $(T, \odot) = (\mathcal{M}_k(\mathbb{S}), \odot)$, and φ the morphism which maps each weighted matrix (M, x) to $\varphi(M)$. Our algorithm will compute, given a presentation of a set of weighted matrices A, an approximation of $\langle A \rangle$ using the inductive principle of the factorization forest theorem. This is justified by the two following lemmas.

Lemma 5. *For all presentable sets of weighted matrices A, A', there exists effectively a presentable set of weighted matrices* $\texttt{product}(A, A')$ *such that*

$$A \otimes A' \approx \texttt{product}(A, A') \ .$$

Lemma 6. *For all presentable sets A such that $\varphi(A) = \{E\}$ for E an idempotent, there is effectively a presentable set* $\texttt{idempotent}(A)$ *such that*

$$\langle A \rangle \approx \texttt{idempotent}(A) \ .$$

Assuming that Lemmas 5 and 6 hold, it is easy to provide an algorithm which, given a presentable set A computes a presentable set $\texttt{closure}(A)$ as follows:

- Set $A_0 = A$ and for all $n = 0 \dots N-1$ (N taken from Theorem 3), set

$$A_{n+1} = A_n \cup \texttt{product}(A_n, A_n) \cup \bigcup_{\substack{E \in \mathcal{M}_k(\mathbb{S}) \\ \text{idempotent}}} \texttt{idempotent}(A_n \cap \varphi^{-1}(E)) \ .$$

- Output $\texttt{closure}(A) = A_N$.

The correctness of this algorithm is given by the following lemma. It derives from the good properties of $\approx$ given in Lemma 4.

Lemma 7. *For all presentable sets of weighted matrices* $\texttt{closure}(A) \approx \langle A \rangle$.

This allows us to conclude the proof of Theorem 2. The algorithm takes an automaton δ, I, F as input, then it computes thanks to the above Lemma 7 a presentable set B that is $\approx$-equivalent to $\langle A \rangle$ where A is the set of weighted matrices corresponding to basic letters (i.e., $\{(\delta(a), 1) \ : \ a \text{ letter}\}$). Set (S, P) a presentation of B. Then the algorithm outputs $\inf\{{}^t I \odot X \odot F \mid X \in P\}$ that is computable since P is a finite union of polytopes. This coefficient is the answer of the algorithm: the minimal exponent such that the presentable set witnesses the existence of a behaviour of the automaton that has this growth-rate.

3 From Size-Change Systems to Max-Plus Automata

For proving Theorem 1, we define a translation of SCSs to max-plus automata. Let $\mathcal{S}$ be an SCS with k variables, which we assume to be numbered $x_1, \ldots, x_k$. We define an max-plus automaton $\phi(\mathcal{S})$ with $k+2$ states as follows: The alphabet $A_\mathcal{S}$ of $\phi(\mathcal{S})$ contains a letter a_T for every SCT $T \in \mathcal{S}$. We define the mapping δ of $A_\mathcal{S}$ to $\mathcal{M}_{k+2}(\overline{\mathbb{N}})$ as follows:

$$\delta(a_T)_{i,j} = \begin{cases} 0, & i = 1 \text{ or } j = k+2 \\ 1, & x_{i-1} > x'_{j-1} \in T \\ 0, & x_{i-1} \geq x'_{j-1} \in T \\ -\infty, \text{ otherwise} \end{cases}$$

Further, $\phi(\mathcal{S})$ has the initial and final vector $I = F = \mathbf{0} \in \mathcal{M}_{1,k+2}(\overline{\mathbb{N}})$. For example, the SCS from Example 1 is translated to the max-plus-automaton in Example 2.

The following lemmata relate SCSs and their translations; they allow us to derive Theorem 1 from Theorem 2.

Lemma 8. *Let u be a word of $\phi(\mathcal{S})$ with ${}^tI \otimes \delta(u) \otimes F = N$. Then $\mathcal{S}$ has a trace with valuations over $[0, N]$ of length $|u|$.*

Lemma 9. *Assume $\mathcal{S}$ has a trace with valuations over $[0, N]$ of length l. Then there is a word u of $\phi(\mathcal{S})$ with ${}^tI \otimes \delta(u) \otimes F \leq N$ and $|u| = l$.*

References

1. Amir M. Ben-Amram and Michael Vainer. Bounded termination of monotonicity-constraint transition systems. *CoRR*, abs/1202.4281, 2012.
2. Thomas Colcombet and Laure Daviaud. Approximate comparison of distance automata. In *STACS*, pages 574–585, 2013.
3. Manfred Droste, Werner Kuich, and Heiko Vogler, editors. *Handbook of Weighted Automata*. Springer-Verlag, 2009.
4. Kosaburo Hashiguchi. Limitedness theorem on finite automata with distance functions. *J. Comput. Syst. Sci.*, 24(2):233–244, 1982.
5. Daniel Krob. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *Internat. J. Algebra Comput.*, 4(3):405–425, 1994.
6. Chin Soon Lee, Neil D. Jones, and Amir M. Ben-Amram. The size-change principle for program termination. In *POPL*, pages 81–92, 2001.
7. Panagiotis Manolios and Daron Vroon. Termination analysis with calling context graphs. In *CAV*, pages 401–414, 2006.
8. Marcel Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4:245–270, 1961.
9. Imre Simon. Factorization forests of finite height. *Theoretical Computer Science*, 72:65–94, 1990.
10. Imre Simon. The nondeterministic complexity of a finite automaton. In *Mots*, Lang. Raison. Calc., pages 384–400. Hermès, Paris, 1990.
11. Florian Zuleger, Sumit Gulwani, Moritz Sinn, and Helmut Veith. Bound analysis of imperative programs with the size-change abstraction. In *SAS*, pages 280–297, 2011.