Tropical Two-Way Automata

Vincent Carnino[1] and Sylvain Lombardy[2]

[1] LIGM - Laboratoire d'informatique Gaspard-Monge
Université Paris-Est Marne-la-Vallée, France
Vincent.Carnino@univ-mlv.fr
[2] LaBRI - Laboratoire Bordelais de Recherche en Informatique
Institut Polytechnique de Bordeaux, France
Sylvain.Lombardy@labri.fr

Abstract. In this paper we study two-way min-plus automata. We prove that two-way distance automata are equivalent to one-way distance automata. In the second part of the paper we show that, with general min-plus semirings, it is decidable whether every accepted word has a weight different from $-\infty$ and that, in contrast, it is undecidable whether there exists a word accepted with a weight different from $-\infty$.

1 Introduction

Min-plus automata have attracted much attention for three decades. $\mathbb{N}$-min-plus is one of the simplest extensions of the Boolean semiring and min-plus automata are therefore a very natural extension of automata with various applications in natural language processing or optimization. They are indeed very powerful tools and take part in some very important results like star height.

We study here two-way min-plus automata. When the weights are non negative, we extend the classical result [8,7] that states that a two-way finite automaton is equivalent to a one-way finite automaton.

In the second part, we show that in general some words may be accepted in some two-way automata by a family of runs whose the infimum over weights is $-\infty$. In this case, the behaviour of the automata may be not rational. We prove that the existence of such accepted words is decidable. In contrast, we prove that given a $\mathbb{Z}$-min-plus automaton, it is undecidable whether there exists a word accepted with a finite weight.

2 Tropical Two-Way Automata

2.1 Automata and Runs

An alphabet is a finite set of letters; for every alphabet A, we assume that there exist two fresh symbols $\vdash$ and $\dashv$ that are marks at the beginning and the end of the tapes of automata. We denote $A_{\vdash\!\dashv}$ the alphabet $A \cup \{\vdash, \dashv\}$. For every word w in A, $w_{\vdash\!\dashv}$ is the word in $A_{\vdash\!\dashv}$ equal to $\vdash w \dashv$.

Tropical automata are instances of automata weighted by a semiring. For every additive submonoid $\mathbb{K}$ of $\mathbb{R}$, we can define the min-plus semiring $\mathcal{K} = (\mathbb{K} \cup \{\infty\}, \min, +)$. For instance, from $\mathbb{N}$, $\mathbb{Z}$ and $\mathbb{R}_+$, we can respectively define the min-plus semirings $\mathcal{N}$, $\mathcal{Z}$ and $\mathcal{R}_+$. Notice that in a $\mathcal{K}$-automaton, only weights in $\mathbb{K}$ appear (∞ is an algebraic way to specify the absence of transition).

In the sequel, we call *distance automaton* every min-plus automaton with non negative weights. Hence, $\mathcal{N}$-automata and $\mathcal{R}_+$-automata are distance automata.

In some applications, the semiring $\mathbb{P} = ([0;1], \max, \cdot)$ is used (the weight of a path is in this case the product of the probabilities of the path and the weight of a word is the weight of the run on this word with the highest probability). The application $x \longmapsto -\log x$ is actually an isomorphism from $\mathbb{P}$ onto $\mathcal{R}_+$. Every result on distance automata is therefore valid for $\mathbb{P}$-automata.

One-way and two-way $\mathcal{K}$-automata share a part of their definition. A $\mathcal{K}$-automaton is a tuple $\mathcal{A} = (Q, A, E, I, T)$ where Q is a finite set of state, A is a finite alphabet, and I and T are partial functions from Q to $\mathbb{K}$. The support of I, $\underline{I}$, is the set of initial states of $\mathcal{A}$, and the support of T, $\underline{T}$, is the set of final states of $\mathcal{A}$.

The definition of transitions differ. In a two-way $\mathbb{K}$-automaton, E is a partial function from $Q \times (A_{\vdash\dashv} \times \{-1, +1\}) \times Q$ into $\mathbb{K}$ and the support of E, $\underline{E}$, is the set of transitions of $\mathcal{A}$. Moreover, the intersection of $\underline{E}$ and $Q \times (\{\vdash\} \times \{-1\} \cup \{\dashv\} \times \{1\}) \times Q$ must be empty.

Let t be a transition in $\underline{E}$; if $t = (p, a, d, q)$, we denote $\sigma(t) = p$, $\tau(t) = q$, $\lambda(t) = a$, $\delta(t) = d$. On figures, the value of δ is represented by a left (-1) or right (+1) arrow. For instance, if $t = (p, a, -1, q)$ and $E_t = k$, we draw $p \xrightarrow{a,\leftarrow|k} q$.

In a one-way $\mathbb{K}$-automaton, E is a partial function from $Q \times A \times Q$ into $\mathbb{K}$, and the support of E, $\underline{E}$, is the set of transitions of $\mathcal{A}$.

Let t be a transition in $\underline{E}$; if $t = (p, a, q)$, we denote $\sigma(t) = p$, $\tau(t) = q$, $\lambda(t) = a$.

Definition 1. *Let $w = w_1 \dots w_n$ be a word of A^*, we set $w_0 = \vdash$ and $w_{n+1} = \dashv$. A* configuration *of $\mathcal{A}$ on w is a pair (t, i) where i is in $[0; n+1]$ and t is a transition of $\mathcal{A}$ with $\lambda(t) = w_i$. A computation (or run) ρ of $\mathcal{A}$ on w is a finite sequence of configurations $((t_1, i_1), \dots, (t_k, i_k))$ such that :*

- *$i_1 = 1$, $i_k = n$, $\delta(t_1) = \delta(t_k) = 1$, $\sigma(t_1)$ is in $\underline{I}$ and $\tau(t_k)$ is in $\underline{T}$;*
- *for every j in $[0; k-1]$, $\sigma(t_{j+1}) = \tau(t_j)$ and $i_{j+1} = i_j + \delta(t_j)$.*

Example 1. Let $\mathcal{A}_1$ be the two-way $\mathcal{N}$-automaton of Figure 1. The automaton checks by a left-right reading the parity of the length of each subsequence of repetitions of $'a'$s; if it is odd, a right-left reading computes the length of the block; otherwise the automaton goes to the next block of $'a'$s.

A run of the $\mathcal{N}$-automaton $\mathcal{A}_1$ over the word *abaaba* is represented on Figure 2. The weight of this run is equal to 2, since there are two odd subsequences of $'a'$s in the string, and each id of length 1.

The weight of such a computation, denoted by $|\rho|$, is $I(\sigma(t_1)) + \sum_{j=1}^{k} E(t_j) + T(\tau(t_k))$. The weight of w in $\mathcal{A}$, denoted by $\langle |\mathcal{A}|, w \rangle$, is the infimum of the weights

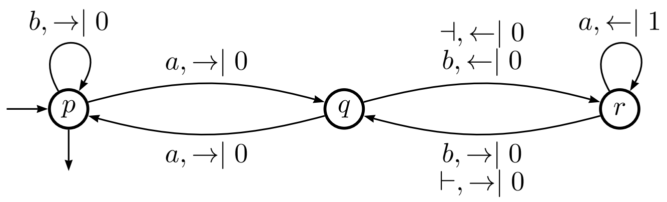


Fig. 1. The two-way $\mathcal{N}$-automaton $\mathcal{A}_1$

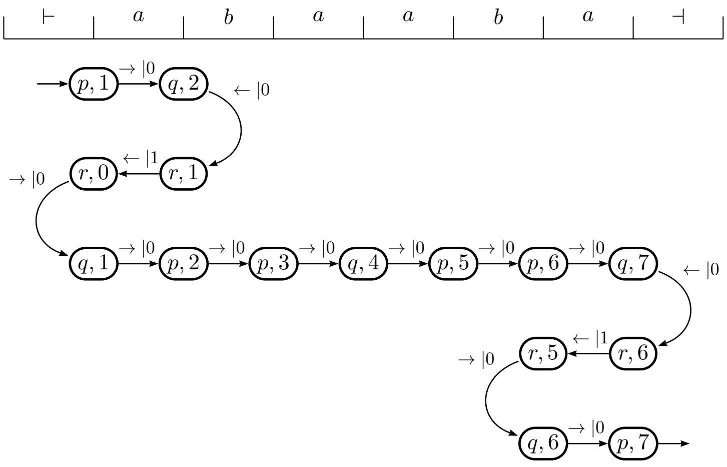

$\vdash$ a b a a b a $\dashv$
$p,1$ $q,2$ $r,1$ $r,0$ $q,1$ $p,2$ $p,3$ $q,4$ $p,5$ $p,6$ $q,7$ $r,6$ $r,5$ $q,6$ $p,7$
$\rightarrow |0$
$\leftarrow |0$
$\leftarrow |1$

Fig. 2. A run of $\mathcal{A}_1$ over the word *abaaba*

of all the runs with label w in $\mathcal{A}$. Notice that there may be an infinite number of computations with the same label w; in this case the infimum may not belong to $\mathcal{K}$; actually, $\mathcal{K}$ can always be embedded into the semiring $(\mathbb{R} \cup \{-\infty, \infty\}, \min, +)$ (with $-\infty + \infty = \infty$) where the infimum of every family is always defined.

2.2 δ-normalization

Definition 2. *Let $\mathcal{A}$ be a two-way $\mathbb{K}$-automaton.*
If, for each state p of $\mathcal{A}$, every outgoing transition from p has the same direction, then $\mathcal{A}$ is δ-local.
If $\mathcal{A}$ is δ-local and, for each state p of $\mathcal{A}$, every transition arriving at p has the same direction, then $\mathcal{A}$ is δ-normalized.

If $\mathcal{A}$ is a δ-local automaton, for every state p in Q, we set $\delta_{\mathbf{O}}(p) = \delta(t)$, where t is any transition outgoing from p; if it is normalized, we also set $\delta_{\mathbf{I}}(p) = \delta(t)$, where t is any transition incoming to p.

The following proposition is proved in [2].

Proposition 1. *For every two-way $\mathbb{K}$-automaton, there exists an equivalent δ-local two-way $\mathbb{K}$-automaton.*

To make a two-way automaton $\mathcal{A}$ δ-local, a *covering* of $\mathcal{A}$ is built: every state p with outgoing transitions with different directions is split into two states p_+ and p_- that have the same incoming transitions as p, transitions outgoing from p_+ (*resp.* p_-) are the transitions outgoing from p with direction $+1$ (*resp.* -1).

The dual construction consists in splitting the states to separate incoming transitions with different directions. Applied to a δ-local two-way automaton, it results in a δ-normalized automaton.

Example 2. Figure 3 shows the conversion of a two-way automaton into a δ-local automaton, and then into a δ-normalized automaton.

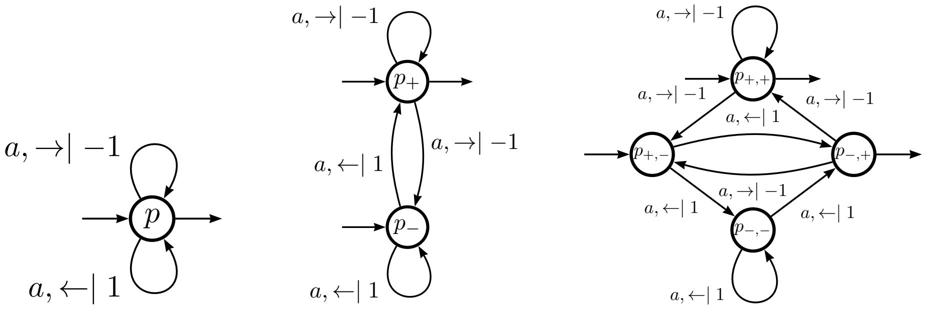


Fig. 3. The two steps of the δ-normalization

Example 3. The automaton $\mathcal{A}_1$ of Figure 1 is not δ-normalized: in states q and r, there are outgoing transitions with direction -1 and others with direction $+1$. The automaton $\mathcal{A}'_1$ of Figure 4 is a δ-normalized equivalent automaton.

2.3 The Slice Automaton

The slice automaton is a one-way automaton that (non-deterministicly) emulates the runs of a two-way automaton. On a given run, for each position of the input, the *slice* of the run is the vector made of the states visited at this position. Every state of the slice automaton is such a vector and there is a transition between two states if the corresponding slices can successively appear in a run of the two-way automaton.

We give here the formal definition of the slices. A more complete description of the slice automaton is given in [2].

Definition 3. *Let $\mathcal{A} = (Q, A, E, I, T)$ be a two-way $\mathbb{K}$-automaton and let $w = w_1 \dots w_k$ be a word. Let $\rho = ((p_0, i_0), \dots (t_n, i_n))$ be a run over w, and j in $[1; k+1]$. Let h be the subsequence of all pairs (p_k, i_k) such that $(i_k, i_{k+1}) =$*

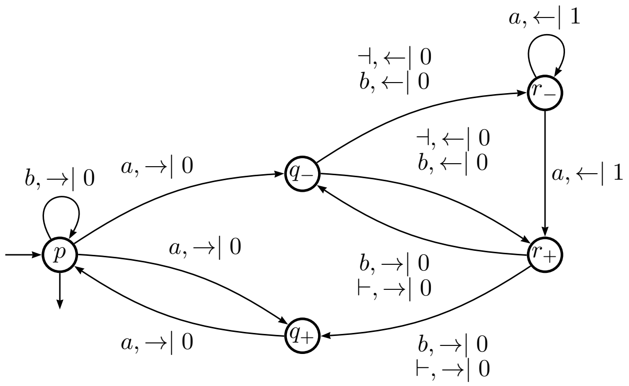


Fig. 4. The δ-normalized two-way $\mathcal{N}$-automaton $\mathcal{A}'_1$

$(j, j+1)$ *or* $(i_{k-1}, i_k) = (j, j-1)$. *The j-th* slice *of ρ is the vector $s^{(j)}$ of states obtained by the projection of the first component of each pair of h.*
The signature $S(\rho)$ *of ρ is the sequence of its slices.*

The slices we define here are not exactly the *crossing sequences* defined in [8].

Example 4. The vector $\begin{bmatrix} q \\ r \\ p \end{bmatrix}$ is the second (and the seventh) slice of the run of Figure 2. The signature of this run is:

$$\left(\begin{matrix} p \\ r \\ q \end{matrix}, \begin{matrix} q \\ r \\ p \end{matrix}, p, q, p, \begin{matrix} p \\ r \\ q \end{matrix}, \begin{matrix} q \\ r \\ p \end{matrix} \right). \tag{1}$$

Let $\mathcal{A} = (Q, A, E, I, T)$ be a δ-local two-way $\mathbb{K}$-automaton. In order to define a one-way $\mathbb{K}$-automaton from slices we consider the set X of subvectors of slices, that are vectors in Q^* with an odd length; let Y be the vectors in Q^* with an even length.

We define inductively two partial functions $\theta : X \times A \times X \to \mathbb{K}$ and $\eta : Y \times A \times Y \to \mathbb{K}$ by:

$$\begin{aligned} && \eta(\varepsilon, a, \varepsilon) &= 0_{\mathbb{K}}, \\ \forall p, q \in Q, \quad & \delta_{\mathbf{O}}(p) = 1 \Rightarrow \forall u, v \in Y, & \theta(pu, a, qv) &= E(p, a, 1, q) + \eta(u, a, v), \\ && \eta(u, a, pqv) &= E(p, a, 1, q) + \eta(u, a, v), \\ & \delta_{\mathbf{O}}(p) = -1 \Rightarrow \forall u, v \in X, & \theta(pqu, a, v) &= E(p, a, -1, q) + \theta(u, a, v), \\ && \eta(qu, a, pv) &= E(p, a, -1, q) + \theta(u, a, yv). \end{aligned} \tag{2}$$

Since $\mathcal{A}$ is δ-local, if θ is defined on a triple (u, a, v), it is uniquely defined.

For every vector pu in X, pu is initial if p is in $\underline{I}$ and $(\varepsilon, \vdash, u)$ is in $\underline{\eta}$; in this case, we set $\mathcal{I}(pu) = I(p) + \eta(\varepsilon, \vdash, u)$. Likewise, every vector up in X is final if p is in $\underline{T}$ and $(u, \dashv, \varepsilon)$ is in $\underline{\eta}$; in this case, we set $\mathcal{T}(up) = \eta(u, \dashv, \varepsilon) + T(p)$.

Definition 4. *With the above notations, the* slice automaton *of the two-way* $\mathbb{K}$*-automaton* $\mathcal{A} = (Q, A, E, I, T)$ *is the infinite one-way* $\mathbb{K}$*-automaton* $\mathcal{C} = (X, A, \theta, \mathcal{I}, \mathcal{T})$.

3 Two-Way Distance Automata

In two-way automata, in the same computation, there may be two steps where the automaton is in the same state and reads the same letter of the input. In this case we say that the computation contains an *unmoving circuit.*

Definition 5. *Let* $\rho = ((t_1, i_1), \ldots, (t_k, i_k))$ *be a run. If there exists* m, n *in* $[1, k]$*, with* $m < n$ *such that* $i_m = i_n$ *and* $\sigma(t_m) = \sigma(t_n)$*, then we say that* $((t_m, i_m), \ldots, (t_{n-1}, i_{n-1}))$ *is an* unmoving circuit *of* ρ*. If* ρ *does not contain any unmoving circuit, it is* reduced.

If a run contains unmoving circuits, they can all be removed with a finite number of iterations since the removing of such a circuit leads to a shorter run.

Lemma 1. *If a two-way* $\mathbb{K}$*-automaton admits a run* ρ *which is not reduced, it admits a reduced run with the same label.*

Lemma 2. *Let* $\mathcal{A}$ *be a two-way distance automaton on an alphabet* A*. For each* w *in* A^**,* $\langle |\mathcal{A}|, w\rangle$ *is the weight of a reduced run of* w*.*

Proof. By contradiction, let us suppose that, for a word w, there is no reduced run in $\mathcal{A}$ labeled by w with a minimal weight. Then let $\rho = ((t_1, i_1), \ldots, (t_l, i_l))$ be one of the shortest non reduced run labeled by w with a minimal weight. Since it is not reduced, then there exist j and k, with $j < k$, such that $i_j = i_k$ and $\sigma(t_j) = \sigma(t_k)$. Then there exists a run $\rho' = ((t_1, i_1), \ldots, (t_{j-1}, i_{j-1}), (t_k, i_k), \ldots, (t_l, i_l))$ labeled by w with $|\rho'| \leq |\rho|$ which is a contradiction. ☐

By Lemma 2 to simulate a two-way distance automaton by a one-way automaton, we only need to simulate reduced runs.

Actually, if a run of a two-way automaton contains an unmoving circuit, the signature of this run contains a vector where two entries with an index with the same parity are equals. In [2], we prove that the restriction of the slice automaton of $\mathcal{A}$ to states labelled by vectors that do not contain this kind of entry results in a finite one-way automaton where every computation corresponds to a reduced computation of $\mathcal{A}$ with the same weight and that every reduced computation of $\mathcal{A}$ has a representative in this finite one-way automaton.

Example 5. From the δ-normalization of $\mathcal{A}_1$, we can build an equivalent one-way $\mathcal{N}$-automaton.

Finally, by Lemma 2,

Proposition 2. *Every two-way distance automaton is equivalent to a one-way distance automaton.*

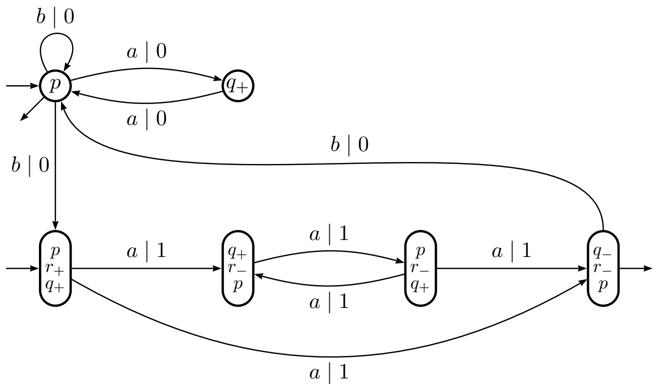


Fig. 5. A one-way $\mathcal{N}$-automaton equivalent to $\mathcal{A}_1$

4 Two-Way Min-plus Automata

In this part, we study two-way automata on min-plus semirings based on non postive submonoids of $\mathbb{R}$. In this case, a word may label an infinite number of paths with an increasingly smaller weight.

We say that a two-way min-plus automaton is *valid* if the weight of every accepted word is finite. We address the problem of deciding whether a two-way min-plus automaton is valid.

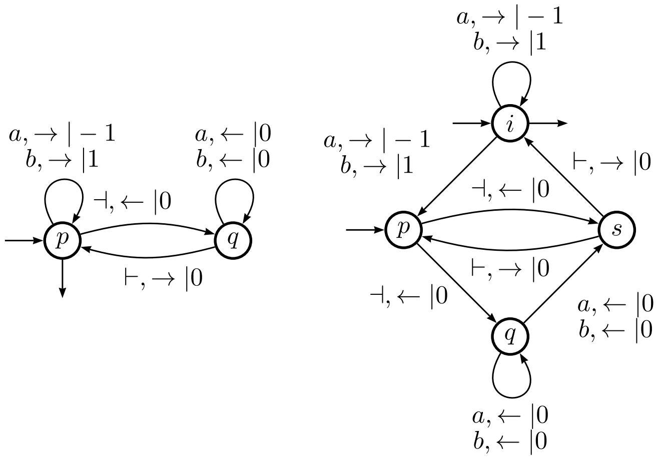


Fig. 6. The two-way $\mathcal{Z}$-automata $\mathcal{A}_2$ and $\mathcal{A}'_2$

Example 6. Let $\mathcal{A}_2$ be the two-way $\mathcal{Z}$-automaton of Figure 6 (left). Every time this automaton reads a word from left to right it computes the difference between the number of 'b's and the number of 'a's. Since for each accepted word, there can be an unbounded number of left-right reading, if there are more 'a's than 'b's, the weight of runs is not lowerly bounded. Thus, the behaviour of this automaton is only defined for words where the number of 'a's is at most equal to the number of 'b's.

The automaton $\mathcal{A}_2'$ of Figure 6 (right) is the δ-normalization of $\mathcal{A}_2$.

This example shows the following fact.

Proposition 3. *There exist two-way min-plus automata such that the language of words accepted with a finite weight is not rational (or regular).*

Theorem 1. *It is decidable whether a two-way min-plus automaton is valid.*

To prove this theorem, we need to consider another restriction of the slice automaton. Unlike the case of distance automata where we want that unmoving circuits do not appear at all, we want here to detect when unmoving circuits appear, but we want to deal with a finite automaton. So, we allow that each unmoving circuit appears at most once.

To this purpose, we consider the slices that belong to $W = \bigcup_k W_k$ with W_k defined for all k in $\mathbb{N}$ as follows :

$$W_k = \{v \in Q^{2k+1} | \forall p \in Q, \forall s \in [0;1], |\{i \mid v_i = p \text{ and } i \mod 2 = s\}| \leqslant 2\}. \quad (3)$$

We consider the restriction of the slice automaton to W.

Proposition 4. *Let $\mathcal{A}$ be a two-way $\mathbb{K}$-automaton and let $\mathcal{C}$ be the restriction of the slice automaton of $\mathcal{A}$ to W. If $\mathcal{A}$ accepts a run that contains an unmoving circuit with a negative weight, then there exists a run in $\mathcal{A}$ that contains an unmoving circuit with a negative weight and that is mapped into $\mathcal{C}$.*

Proof. Assume that there exist runs of $\mathcal{A}$ that contain at least one unmoving circuit with a negative weight. We chose ρ among these runs with a minimal number of transitions. Let q be the end of the unmoving circuit with negative weight and let q_1 and q_2 be the both occurences of q. If ρ is not mapped into $\mathcal{C}$, there exists a state p that appears (at least) three times in a slice v_1 of ρ (let p_1, p_2 and p_3 be these three occurences); p is the end of two consecutive unmoving circuits.
Different cases occur. If one of the two consecutive unmoving circuits has a negative weight, the other one can be removed to simplify the run. (This case may occur if $p = q$.)
Likewise, if one of the two consecutive unmoving circuits does not intersect the unmoving circuit with negative weight, it can be removed.
The only case that remains is when the run ρ can be decomposed as:

$$\to i \xrightarrow{w_1|k_1} p_1 \xrightarrow{w_2|k_2} q_1 \xrightarrow{w_3|k_3} p_2 \xrightarrow{w_4|k_4} q_2 \xrightarrow{w_5|k_5} p_3 \xrightarrow{w_6|k_6} t \to . \quad (4)$$

In this case we have $k_3 + k_4 < 0$ and the shorter run

$$\to i \xrightarrow{w_1|k_1} p_1 = p_2 \xrightarrow{w_4|k_4} q_2 = q_1 \xrightarrow{w_3|k_3} p_2 = p_3 \xrightarrow{w_6|k_6} t \to \quad (5)$$

contains an unmoving circuit with a negative weight. □

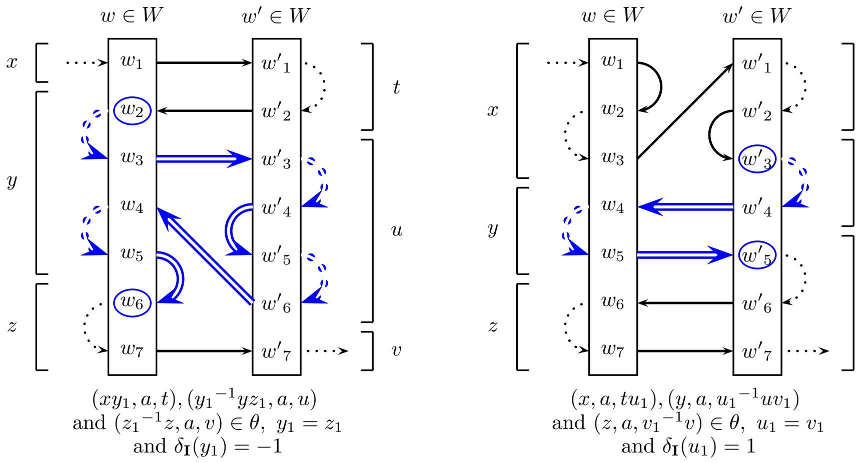
In the automaton $\mathcal{C}$, every run that meets a state in $W \setminus V$ does contain an unmoving circuit. The problem is to detect whether such an unmoving circuit has a negative weight. The solution consists in comparing the weight of this run with the weight of the run without the unmoving circuit. To this purpose, we define an automaton which is a kind of *square* of the automaton $\mathcal{C}$ (*cf.* [1]): it compares paths of $\mathcal{C}$ that differ by unmoving circuits.

We consider first the set $X = \{(x,y,z) \in (Q^*)^3 \mid xz, xyz \in W\}$. An element (x,y,z) in X is *special* if $y_1 = z_1$. From the function θ, we define the (partial) function $\tilde{\theta} : X \times A \times X \longrightarrow \mathbb{K}$ as

$$\tilde{\theta}((x,y,z),a,(t,u,v)) = \theta(xyz,a,tuv) - \theta(xz,a,tv), \tag{6}$$

for every triple $((x,y,z),a,(t,u,v))$ that fulfils one of the three following conditions:

$$\begin{aligned} &(x,a,t),(y,a,u) \text{ and } (z,a,v) \in \underline{\theta};\\ &(xy_1,a,t),(y_1^{-1}yz_1,a,u) \text{ and } (z_1^{-1}z,a,v) \in \underline{\theta}, y_1 = z_1 \text{ and } \delta_{\mathbf{I}}(y_1) = -1;\\ &(x,a,tu_1),(y,a,u_1^{-1}uv_1) \text{ and } (z,a,v_1^{-1}v) \in \underline{\theta}, u_1 = v_1 \text{ and } \delta_{\mathbf{I}}(u_1) = 1. \end{aligned} \tag{7}$$



$w \in W$
$w' \in W$
x
y
z
t
u
v
w_1
w_2
w_3
w_4
w_5
w_6
w_7
w'_1
w'_2
w'_3
w'_4
w'_5
w'_6
w'_7
$(xy_1,a,t),(y_1{}^{-1}yz_1,a,u)$ and $(z_1{}^{-1}z,a,v) \in \theta,\ y_1 = z_1$ and $\delta_{\mathbf{I}}(y_1) = -1$
$(x,a,tu_1),(y,a,u_1{}^{-1}uv_1)$ and $(z,a,v_1{}^{-1}v) \in \theta,\ u_1 = v_1$ and $\delta_{\mathbf{I}}(u_1) = 1$

Fig. 7. A valid run over the word $w = w_1 \dots w_k$

Let $X_0 = \{(x,y,z) \in X \mid y = 1\}$. We define on X the relation $(x,y,z) \equiv (x',y',z')$ if and only if $y = y' = 1$ and $xz = x'z'$. In the quotient $\tilde{X}$ of X by $\equiv$, every element which is not in X_0 is the only element of its class, while the quotient of X_0 is isomorphic to W. Moreover, this equivalence is compliant with the definition of $\tilde{\theta}$. Let $\mathcal{P} = (\tilde{X}, A, \tilde{\theta}, J, U)$ be a one-way automaton defined as follows. The transition function is $\tilde{\theta}$; every transition that corresponds to one of the two last lines of (7) is called a special transition. We set $J(x,y,z) = \theta(0,\vdash,xyz) - \theta(0,\vdash,xz)$ if x is non empty or if x is empty and $y_1 = z_1$ (special initial state). Likewise, $U(x,y,z) = \theta(xyz,\dashv,0) - \theta(xz,\dashv,0)$ if z is non empty or if z is empty and $x_1 = y_1$ (special final state).

Every computation in the automaton $\mathcal{P}$ that meets one (and only one) special transition (or special initial or final state) corresponds to two computations in the slice automaton. Each state of the first computation is obtained from every state (x,y,z) of the computation in $\mathcal{P}$ by concatenating x, y and z, while each state of the second computation is given by the concatenation of x and z. This two computations correspond to two runs in $\mathcal{A}$, one with an unmoving circuit, the second one where the unmoving circuit has been removed.

Example 7. From the automaton $\mathcal{A}'_2$ of Figure 6 (right), we can build the automaton $\mathcal{P}_2$ of Figure 8. The states in X_0 are labelled by one vector, and the other ones by three vectors. The special transitions and special initial states are red (there is no special final state). Each run in this automaton that contains a special transition (or initial state) corresponds two runs in $\mathcal{A}'_2$, one with an unmoving circuit, the other one without this circuit. Such a pair of paths in $\mathcal{A}'_2$ may correspond to several paths in $\mathcal{P}_2$, depending where the path with the unmoving circuit is cut. For instance, consider the path $\left(i, \begin{smallmatrix} s \\ i \end{smallmatrix}, \text{-}\right) \xrightarrow{a,-1} \left(\text{-}, \begin{smallmatrix} i \\ q \end{smallmatrix}, i\right) \xrightarrow{a,-1} \left(\text{-}, \begin{smallmatrix} p \\ q \end{smallmatrix}, i\right)$; it corresponds to the path of Figure 9 and to the cut between the two red states. The weight of the path with the unmoving circuit is -4, without the unmoving circuit, it is -2; the difference is -2 which is equal to the weight of the path in $\mathcal{P}_2$.

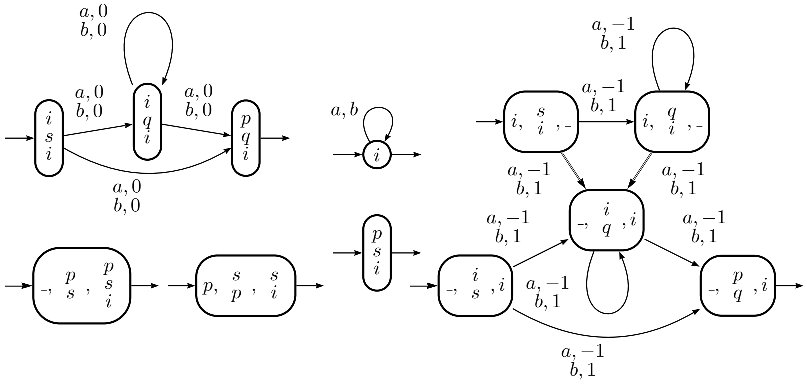


Fig. 8. The one-way automaton $\mathcal{P}_2$

Proposition 5. *Let $\mathcal{A}$ be a two-way min-plus automaton and let $\mathcal{P}$ be the automaton built above. If $\mathcal{A}$ accepts a run that contains an unmoving circuit with a negative weight, then there is a run in $\mathcal{P}$ that meets one special transition (or special initial/final state) with a negative weight.*

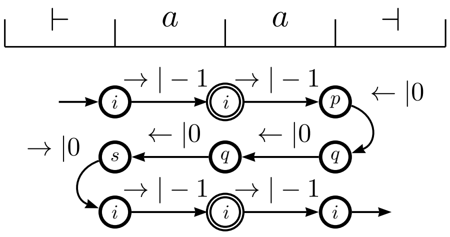


Fig. 9. A path in the automaton $\mathcal{A}_2$ with a negative unmoving circuit

This property can be checked on the automaton $\mathcal{P}$ in polynomial time (see for instance [5]), and this implies Theorem 1.

If the two-way automaton is not valid, it could be interesting to compute an effective description of the language on which the behaviour is defined. By Proposition 3 this language must be non rational; worst, it is undecidable to know whether it is empty.

Theorem 2. *Let $\mathcal{A}$ be a two-way $\mathcal{Z}$-automaton. It is undecidable whether there exists a word w accepted by $\mathcal{A}$ with a finite weight.*

Proof. In [4], it is prove that it is undecidable, given a one-way $\mathcal{Z}$-automaton $\mathcal{B} = (Q, A, E, I, T)$, to know whether there exists a word w whose weight in $\mathcal{B}$ is non negative. Let r be an element which is not in Q and let $\mathcal{A} = (Q \cup \{r\}, A, F, I, T)$ be the two-way $\mathcal{Z}$-automaton defined as follow:

$$\begin{aligned} F =& \{p \xrightarrow{a,\rightarrow|k} q \mid p \xrightarrow{a|k} q \in E\} \\ =& \{p \xrightarrow{\dashv,\leftarrow|k} r \mid p \in \underline{T}, T(p) = k\} \\ =& \{r \xrightarrow{\vdash,\rightarrow|k} p \mid p \in \underline{I}, I(p) = k\} \\ =& \{r \xrightarrow{a,\leftarrow|0} r \mid a \in A\} \end{aligned} \qquad (8)$$

For every word w in $\mathcal{A}$, every computation on w is any sequence of computations on w in $\mathcal{B}$. Therefore, the weight of a word w in $\mathcal{A}$ is defined if and only if it has no computation with a negative weight in $\mathcal{B}$, that is if its weight in $\mathcal{B}$ is non negative. □

5 Conclusion

The problem tackled in this paper raises a more general problem on two-way automata. Actually, since the number of computations for a given input can be infinite, a proper definition of the behaviour of a weighted automaton must be forged. It meets some works on the behaviour of one-way weighted automata with ε-transitions (*cf.* [3,6]).

The last part of the paper also introduces some open questions. Despite the fact that the emptiness of the domain of a tropical two-way automaton is undecidable, is it possible to give a usable characterization of this domain?

References

1. Béal, M.-P., Carton, O., Prieur, C., Sakarovitch, J.: Squaring transducers: an efficient procedure for deciding functionality and sequentiality. Theor. Comput. Sci. 292(1), 45–63 (2003)
2. Carnino, V., Lombardy, S.: On determinism and unambiguity of weighted two-way automata. In: AFL 2014 (accepted, 2014)
3. Ésik, Z., Kuich, W.: Finite automata. In: Droste, M., et al. (eds.) Handbook of Weighted Automata, pp. 69–104. Springer (2009)
4. Krob, D.: The equality problem for rational series with multiplicities in the tropical semiring is undecidable. Internat. J. Algebra Comput. 4(3), 405–425 (1994)
5. Lombardy, S., Mairesse, J.: Series which are both max-plus and min-plus rational are unambiguous. RAIRO - Theor. Inf. and Appl. 40(1), 1–14 (2006)
6. Lombardy, S., Sakarovitch, J.: The validity of weighted automata. Internat. J. Algebra Comput. 23, 863–913 (2013)
7. Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM J. Res. Dev. 3(2), 114–125 (1959)
8. Shepherdson, J.C.: The reduction of two-way automata to one-way automata. IBM J. Res. Dev. 3(2), 198–200 (1959)