Two-Way Automata over Locally Finite Semirings

Louis-Marie Dando, Sylvain Lombardy

Two-way Automata over Locally Finite Semirings

Louis-Marie Dando and Sylvain Lombardy

LaBRI UMR 5800, Université de Bordeaux, INP Bordeaux, CNRS
Bordeaux, FRANCE
{louis-marie.dando,sylvain.lombardy}@labri.fr

Abstract. Two-way transducers or weighted automata are in general more powerful than one-way ones. We show that two-way automata over locally finite semirings may have undefined behaviour. We prove that it is decidable whether this behaviour is defined, and, if it is, we show that two-way automata over locally finite semirings are equivalent to one-way automata.

1 Introduction

Weighted two-way automata and transducers have been recently intensively studied for their interest in verification [3]. Their expressiveness is in general larger than the expressiveness of one-way models. We consider in this paper two-way automata over locally finite semirings. Finite or locally finite semirings occur in many models, like distributive lattices or fuzzy logic. One-way automata over these semirings have been studied for many decades. For instance, the first proof of the limitedness problem [4] relies on automata on the idempotent semiring $\{0, 1, \omega, \infty\}$ (where 1 means "something" and ω means "a lot").

It is folklore that every one-way automaton over a locally finite semiring is equivalent to a deterministic finite automaton where the weight of the run only depends on the state where the run stops. For two-way automata over locally finite semirings, the situation is not as simple. For instance, if the weights belong to $\mathbb{Z}/2\mathbb{Z}$, then the weight of an input depends on the parity of the number of accepting runs; since in a two-way automaton, there may exist an infinite number of runs accepting some input, this weight may be not defined.

For every two-way automaton over a locally finite semiring, we build an automaton that describes the potentially infinite family of weights of runs on every input. From this object, knowing which infinite sums are defined in the locally finite semiring is sufficient to decide whether the behaviour of the two-way automaton is defined. We also prove that every two-way automaton over a locally finite semiring with a defined behaviour is equivalent to a one-way automaton.

In Section 2, we consider locally finite semirings. In particular, we study how the *additive order* allows to encode infinite sums. In Section 3, we introduce weighted two-way automata over locally finite semirings and we show that they

can be normalized in such a way that the weight of every run only depends on the final state, and the move of the input head is fully characterized by the current state. In Section 4, we use an extension of crossing sequences [6, 7] to convert a two-way automaton into a one-way automaton. It leads to a deterministic one-way automaton where each final state describes the (potentially infinite) family of weights of runs of the two-way automaton on the input. It is then decidable whether the weight of every input is defined, and, if it is, then the deterministic automaton can be turned into a (deterministic) one-way automaton equivalent to the two-way automaton.

2 Locally Finite Semirings

A semiring $(\mathbb{K}, +, .)$ is a set $\mathbb{K}$ endowed with two associative operations: a commutative *addition* and a *multiplication* that is distributive over the addition. Moreover, the semiring contains at least two distinct elements which are respectively neutral for each of these operations: $0_\mathbb{K}$ for the addition and $1_\mathbb{K}$ for the multiplication; it is also required for $0_\mathbb{K}$ to be an annihilator for the multiplication.

A semiring $\mathbb{K}$ is locally finite if, for every finite subset F of $\mathbb{K}$ containing both $0_\mathbb{K}$ and $1_\mathbb{K}$, the semiring generated by F is finite.

Moreover, we assume that $\mathbb{K}$ is endowed with a partially defined operator for countable sums: $\sum_I x$, where I is a countable set and $x = (x_i)_{i \in I}$ a family of elements of x. If I is finite, then $\sum_I x$ is always defined and equal to the sum of elements of x. We also assume that two families equal up to a permutation have the same sum.

In this paper, we deal with weighted automata where a finite number of elements of some locally finite semiring occur. These elements generate a finite semiring. In the sequel, we assume that the semiring $\mathbb{K}$ is finite.

Example 1. Let $(L, \vee, \wedge)$ be an infinite distributive lattice; L is a semiring with $\vee$ as addition and $\wedge$ as multiplication. 0_L is the minimum element of L and 1_L is the maximum element. The infinite sum of a family of elements of L is the supremum of this family; if it exists. In lattices, the supremum of families with a finite number of distinct elements is always defined. □

Example 2. Let $\mathbb{K}_1 = \{o, i, x\}$ be the finite semiring where o is the zero, i is the unit, $i + i = o$, and $r + x = r.x = x$ for every r in $\{i, x\}$. Intuitively, this semiring allows to count values modulo 2, and x stands for values where the parity information is lost. In $\mathbb{K}_1$, the sum of any family which contains at least one x is defined and equal to x, and a family that does not contain any x is summable if and only if the number of occurrences of i is finite. □

Since $\mathbb{K}$ is finite and families are unordered, a family is totally characterized by the number of occurrences of each element of $\mathbb{K}$. Hence, we represent such a family s by a vector v in $(\mathbb{N} \cup \{\infty\})^\mathbb{K}$ such that, for every x in $\mathbb{K}$, v_x is the number of occurrences of x in s. The *evaluation* of v is the sum of s if it is defined. Notice

that if s and s' are two families respectively represented by vectors v and v', then the union of s and s' is represented by $v + v'$.

In Section 4.3, a one-way deterministic $\mathbb{K}$-automaton equivalent to a two-way $\mathbb{K}$-automaton is built. The states of this automaton store vectors representing families. In order to build a finite number of states, we need to define a finite set of representatives. Two vectors v and v' in $(\mathbb{N} \cup \{\infty\})^{\mathbb{K}}$ are equivalent if for every vector u, $v + u$ and $v' + u$ have the same evaluation. This property will be required during a determinization step.

We show that every vector in $(\mathbb{N} \cup \{\infty\})^{\mathbb{K}}$ is equivalent to a vector where finite entries are bounded. We consider the natural external product : for every (k, x) in $\mathbb{N} \times \mathbb{K}$, $k.x$ is the sum of k elements x. Likewise, if it is defined, the infinite sum of x is denoted $\infty.x$.

Definition 1. *Let x be an element of a locally finite semiring $\mathbb{K}$. The* additive order *of x is the minimal couple (n_x, p_x) of integers (with $p_x > 0$) verifying*

$$\forall k \geqslant n_x,\ (k + p_x).x = k.x\,. \tag{1}$$

The additive order of x is always defined in a locally finite semiring. Actually, the set $\{k.x \mid k \in \mathbb{N}\}$ is finite and the sequence $s = (k.x)_{k\in\mathbb{N}}$ is ultimately periodic: if $s_i = s_j$, then for every k $s_{i+k} = i.x + k.x = j.x + k.x = s_{j+k}$. Thus, p_x is the minimal distance between two occurrences of the same value in s and n_x is the smallest i such that s_i appears infinitely often in s.

If (n_x, p_x) is the order of x, then for every (m, q) with m larger than or equal to k and q multiple of p_x, it also holds $\forall k \geqslant m,\ (k + q).x = k.x$. The additive order of a finite semiring is then the minimal pair which is admissible for every element of the semiring.

Definition 2. *Let $\mathbb{K}$ be a finite semiring. The* additive order *of $\mathbb{K}$ is the couple*

$$(\max\{n_x \mid x \in \mathbb{K}\}, \operatorname{lcm}\{p_x \mid x \in \mathbb{K}\}), \tag{2}$$

where $\operatorname{lcm} X$ *is the* least common multiple *of elements of X. For every k in $\mathbb{N} \cup \{\infty\}$, the value of k* modulo (n, p) *is defined as*

$$k \bmod (n, p) = \begin{cases} k & \text{if } k < n \text{ or } k = \infty \\ n + ((k - n) \bmod p) & \text{otherwise.} \end{cases} \tag{3}$$

Hence, $k \bmod (n, p)$ is in $[\![0; n + p - 1]\!] \cup \{\infty\}$ and if the additive order of a finite semiring $\mathbb{K}$ is (n, p), then for every x in $\mathbb{K}$, $k.x = (k \bmod (n, p)).x$.

If (n, p) is the additive order of $\mathbb{K}$, then every vector v in $(\mathbb{N} \cup \{\infty\})^{\mathbb{K}}$ is equivalent to the vector where each entry is considered modulo (n, p). The number of distinct vectors modulo (n, p) is equal to $(n + p + 1)^{|\mathbb{K}|}$ (some entries may be equal to ∞).

In the sequel, $\mathcal{N}_{n,p}$ is the semiring $[\![0; n + p - 1]\!] \cup \{\infty\}$ where the operations between finite integers are modulo (n, p).

Example 3. The additive order of elements of $\mathbb{K}_1$ (Example 2) is: $\mathsf{order}(o) = (0, 1)$, $\mathsf{order}(i) = (0, 2)$, and $\mathsf{order}(x) = (1, 1)$. Hence, the additive order of $\mathbb{K}_1$ is $(1, 2)$: for every element k in $\mathbb{K}_1$, $k = 3.k$. □

3 Two-way Automata

3.1 Definition and Behaviour

We follow in this paper the definition of two-way weighted automata given in [5]. Contrary to the classical definition of two-way finite automata [7, 6], the move of the reading head does not depend on the transitions but on the states. This model was first used in [1] and was there proved to be equivalent to the classical one.

Definition 3. *Let $\mathbb{K}$ be a locally finite semiring, A an alphabet and $\Box$ a special symbol, called endmarker, which does not belong to A. A two-way $\mathbb{K}$-automaton over A is a tuple (F, B, E, I, T), where*
– F is the set of forward *states, B the set of* backward *states; $Q = F \cup B$ is the set of states;*
– $E : Q \times (A \cup \{\Box\}) \times Q \longrightarrow \mathbb{K}$ is the transition weight function;
– $I : F \longrightarrow \mathbb{K}$ and $T : F \longrightarrow \mathbb{K}$ are the initial and the final weight function.
The set of transitions *is the support of E, the set of* initial states *is the support of I, and the set of* final states *is the support of T. For every transition $e = (p, a, q)$, $\lambda(e) = a$ is the* label *of e, $\sigma(e) = p$ its* source*, and $\tau(e) = q$ its* target*.*

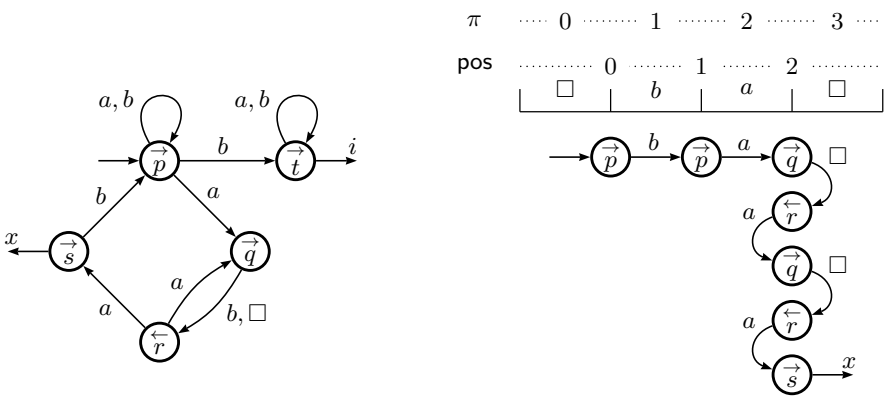
For every state p, we note $\delta(p) = 1$ if p is in F and $\delta(p) = -1$ if p is in B. The head of a two-way automaton can not go beyond endmarkers, hence for every pair (p, q) of states, if $\delta(p)\delta(q) = 1$, then $E(p, \Box, q) = 0_{\mathbb{K}}$.
A run of length k in the automaton is a triple $(p, (e_i)_{i\in[\![1;k]\!]}, q)$, where p and q are in F and $(e_i)_{i\in[\![1;k]\!]}$ is a sequence of transitions such that, for every i in $[\![1; k-1]\!]$, $\tau(e_i) = \sigma(e_{i+1})$; if $k > 0$, then $p = \sigma(e_1)$ and $q = \tau(e_k)$; otherwise $p = q$.
This run admits a label $w = w_1 \dots w_n$ in A^* if there exists a mapping $\pi : [\![1; k]\!] \to [\![0; n+1]\!]$ such that $\pi(1) = 1$, $\pi(k) = n$, and
– for every i in $[\![2; k]\!]$, $\pi(i) = \pi(i-1) + \delta(\sigma(e_i))$,
– for every i in $[\![1; k]\!]$, $\lambda(e_i) = w_{\pi(i)}$ if $\pi(i)$ is in $[\![1; n]\!]$, and $\lambda(e_i) = \Box$ otherwise.
The function π is the *position mapping*, giving the position of the reading head during the computation.

The run $(p, (e_i)_{i\in[\![1;k]\!]}, q)$ meets $k+1$ states. The sequence of these states is defined by $p_0 = p$ and $p_j = \tau(e_j)$ for every j in $[\![1; k]\!]$. The *position* of the j-th state of the run is defined by $\mathsf{pos}(0) = 0$, and

$$\forall j \in [\![1; k]\!],\ \mathsf{pos}(j) = \begin{cases} \mathsf{pos}(j-1) & \text{if } \delta(p_j) \neq \delta(p_{j-1}) \\ \mathsf{pos}(j-1) + \delta(p_j) & \text{otherwise} \end{cases} \tag{4}$$

The positions of states and transitions are related; for every $j \in [\![1; k]\!]$, $\mathsf{pos}(j) = \pi(j)$ if $\delta(p_j) = 1$, and $\mathsf{pos}(j) = \pi(j) - 1$ otherwise.

The weight of a run $(p, (e_i)_{i\in[\![1;k]\!]}, q)$ is $I_p.\big(\prod_{i=1}^{k} E(e_i)\big).T_q$. The weight of a word w in a two-way weighted automaton is defined if the family of the weights of runs with label w is summable.



(a) The two-way $\mathbb{K}_1$-automaton $\mathcal{A}_1$.

(b) A run of $\mathcal{A}_1$ over the word ba

Fig. 1. A two-way $\mathbb{K}_1$-automaton and one of its runs.

Definition 4. *An automaton is* valid *if for every word w the weight of w in the automaton is defined, and two automata are equivalent if, for every word w, either the weight of w is undefined in both automata, or the weight of w is defined in both automata and is equal.*

Remark 1. If $B = \emptyset$, then the two-way $\mathbb{K}$-automaton is actually one-way, and every one-way $\mathbb{K}$-automaton can be described as a two-way $\mathbb{K}$-automaton where $B = \emptyset$.

Example 4. Let $\mathcal{A}_1$ be the two-way $\mathbb{K}_1$-automaton of Figure 1. The final weight of the states is represented by an arrow outgoing from states. States p, q, s and t are forward, r is backward. States p, q and r are not final, which means that their final weight is equal to $0_{\mathbb{K}_1} = o$. The weight of every transition is $1_{\mathbb{K}_1} = i$: it is a *characteristic* automaton.

Figure 1(b) shows a run of $\mathcal{A}_1$. State p appears at position 0 and 1. State r appears twice at the same position; this is an *unmoving circuit*: the part of the run between the two occurrences of r can be repeated in order to get longer runs over the same word. □

Remark 2. In the sequel, we state results for two-way automata over finite semirings. If $\mathcal{A}$ is a two-way automaton over a locally finite semiring $\mathbb{K}$, then it can be considered as an automaton over the finite semiring $\mathbb{K}'$ where $\mathbb{K}'$ is the semiring generated by the weights occurring in $\mathcal{A}$.

3.2 Characteristic and δ-normalized Two-way Automaton

Like for one-way automata, we show here that two-way automata with weights in a finite semiring are equivalent to *characteristic* two-way automata where the weight of the run only depends on the final state. In our model, in such an

automaton, the weights of all transitions as well as initial weights are all equal to $1_{\mathbb{K}}$.

Proposition 1. *Let $\mathcal{A}$ be a two-way automaton over a locally finite semiring. There exists a characteristic two-way automaton $\mathcal{B}$ equivalent to $\mathcal{A}$.*

Actually, through a run, the product of weights spans over a finite set and this can be stored in states. Hence, if $\mathbb{K}$ is the finite semiring generated by the weights of $\mathcal{A}$, then states of $\mathcal{B}$ belong to $Q \times \mathbb{K}$.

Figure 1(b) shows that transitions during a run can be classified into four types: forward transitions, backward transitions, and forward and backward half-turns. We consider now δ-normalized two-way automata, where the type of a transition depends on its source.

Definition 5 ([2]). *A two-way automaton $\mathcal{A} = (F, B, E, I, T)$ is* δ-normalized *if F and B are respectively partitioned into $F = F_+ \cup F_-$ and $B = B_+ \cup B_-$, such that every final state is in F_+ and, for every state p and every transition (p, a, q), q is in F if and only if p is in $F_+ \cup B_+$.*

Example 5. The automaton $\mathcal{A}_1$ of Figure 1 is δ-normalized. Actually, $F_+ = \{p, s, t\}$, $F_- = \{q\}$, $B_+ = \{r\}$ and $B_- = \emptyset$. □

Proposition 2 ([2]). *For every two-way $\mathbb{K}$-automaton $\mathcal{A}$, there exists an equivalent δ-normalized $\mathbb{K}$-automaton $\mathcal{B}$.*

To convert $\mathcal{A}$ into $\mathcal{B}$, every state of $\mathcal{A}$ is split into two copies, the outgoing transitions that go to a forward state are assigned to the first copy, those that go to a backward state are assigned to the second one. This construction applied to a characteristic automaton gives a characteristic automaton.
From now on, we assume that two-way automata are characteristic and δ-normalized.

4 Counting Paths

To convert two-way $\mathbb{K}$-automata into one-way automata, we use a variant of the method of crossing sequences initiated in [6]. Nevertheless, the method must be improved in order to keep records of the number of paths.

More precisely, let (n, p) be the order of $\mathbb{K}$. The algorithm builds a (one-way) deterministic automaton $\mathcal{B}$, where each state is a crossing sequence (that may have states repeating at most once). $\mathcal{B}$ is then used to build a one-way automaton. The idea is to build from $\mathcal{B}$ an automaton that "counts" (modulo (n, p)) the number of runs with a given weight ending in the last state of a crossing sequence.

4.1 Crossing Sequences

Definition 6. *Let ρ be a run with label w in a two-way automata, and let $(p_j)_{j\in[\![0;k]\!]}$ be the sequence of states met by ρ. For every i in $[\![0;|w|]\!]$, the* crossing sequence *of ρ at position i is the subsequence of states p_j such that $\mathsf{pos}(j) = i$.*

Example 6. On Figure 1(b), crossing sequences are (p), (p) and (q, r, s, r, s).

Remark 3. Crossing sequences are sequences of states with odd length. More precisely, the first state is in F, and there is an alternation of states in F and B. Using a regular expression, we can say that crossing sequences are in $(FB)^*F$.

If a crossing sequence of ρ contains (at least) two occurrences of the same state, it means that during the run, the automaton comes back to the same state with the input head at the same position; we call this an *unmoving circuit*. Such a circuit can be removed from ρ in order to obtain a valid run for the same input. On the other hand, it can be iterated to produce an infinite number of valid runs. In a characteristic automaton, all these runs have the same weight.

In the classical crossing sequence construction [6], *reduced crossing sequences*, that are crossing sequences without repetitions, are considered; we have to consider also *crossing sequences with 1 repetition* in which no state appears more than twice, in order to detect unmoving circuits.

Proposition 3 ([2]). *Let ρ be a run of a δ-normalized two-way $\mathbb{K}$-automaton over a word w, and let $(c_i)_{i\in[\![0,|w|]\!]}$ be the list of the crossing sequences of ρ. Then ρ is characterized by w and $(c_i)_{i\in[\![0,|w|]\!]}$.*

Notice that this proposition is independent from the weights on transitions and that it is true whether or not the two-way automaton is characteristic.
We describe briefly how, starting with two consecutive crossing sequences c_{i-1} and c_i, it is possible to build the unique list of transitions in position i which is consistent with these crossing sequences, and the i-th letter of w denoted a.

The algorithm scans sequences c_{i-1} and c_i. If the current state p of c_{i-1} is in F_+, there is a transition from p to the current state q of c_i; if it is in F_-, there is a transition from p to the next state p' of c_{i-1}. The algorithm deals then with the next states. If the current states are in B, the analysis is based on the nature of the current state in c_i. This process produces the list of transitions in position i in the run. It is formally described in [2].

If c_{i-1} and c_i are not successive crossing sequences of a run, the algorithm may fail or build a list of triples which are not transitions of the automaton.

Definition 7. *Let $\mathcal{A} = (F, B, E, I, T)$ be a δ-normalized two-way $\mathbb{K}$-automaton, and let c_1 and c_2 be two potential crossing sequences in $(FB)^*F$.*

- *The sequence c_2 is a* successor *of c_1 by the letter a if the analysis applied to c_1 and c_2 succeeds and returns a list of transitions of $\mathcal{A}$ with label a.*
- *$c_1 = (p_i)_{i\in[\![1;2h+1]\!]}$ is an* initial crossing sequence *if p_1 is initial, and, for every i in $[\![1;h]\!]$, $(p_{2i}, \square, p_{2i+1})$ is a transition of $\mathcal{A}$.*

– $c_1 = (p_i)_{i \in [\![1;2h+1]\!]}$ *is an* final crossing sequence *if* p_{2h+1} *is final, and, for every* i *in* $[\![1;h]\!]$, $(p_{2i-1}, \square, p_{2i})$ *is a transition of* $\mathcal{A}$.

Let $\mathcal{S}$ be the infinite one-way *automaton of crossing sequences* of the δ-normalized two-way $\mathbb{K}$-automaton $\mathcal{A}$. Its states are the elements of $(FB)^*F$. A state c is initial if c is an initial crossing sequence, it is final if c is a final crossing sequence, and there is a transition from c to c' with label a if c' is a successor of c by a. This automaton accepts the same words as $\mathcal{A}$, and there is a bijection between runs of $\mathcal{S}$ and runs of $\mathcal{A}$. $\mathcal{S}$ can be turned into a $\mathbb{K}$-automaton: for every final crossing sequence $c = (p_i)_{i \in [\![1;2h+1]\!]}$, the final weight of c can be set as the final weight of p_{2h+1} in $\mathcal{A}$. This gives an infinite one-way $\mathbb{K}$-automaton equivalent to $\mathcal{A}$. Notice that the $\mathbb{K}$-automaton of crossing sequences of a two-way $\mathbb{K}$-automaton $\mathcal{A}$ is not always equivalent to $\mathcal{A}$, in particular if $\mathbb{K}$ is not commutative. Here, the fact that $\mathcal{A}$ is characteristic is crucial.

To convert two-way automata (without weights) to NFA, it is sufficient to restrict $\mathcal{S}$ to states without any repetition; this is a classic construction of one-way NFA from two-way NFA.

We want to check whether there is an infinite number of runs that end in a given state. As we have seen, as soon as a run has an unmoving circuit, this circuit can be iterated to get an infinite number of runs. Therefore, it is unnecessary to keep track of crossing sequences with more than two occurrences of the same state. If a crossing sequence c of a run ρ contains three times the same state, then the unmoving circuit between the first and the second one (or between the second one and the third one) can be removed; the resulting path has still an unmoving circuit.

Lemma 1. *Let* $\mathcal{A}$ *be a* δ*-normalized characteristic two-way* $\mathbb{K}$*-automaton, and let* ρ *be a run on* $\mathcal{A}$*. If* ρ *admits a crossing sequence at position* i *in wich a state* p *appears more than two times, then there exists a run* ρ' *in which* p *appears only twice in the crossing sequence at position* i*, and* ρ' *and* ρ *end in the same state.*

Each crossing sequence with 1 repetition is the witness of an infinite number of runs with the same weight. Lemma 1 tells that every run of $\mathcal{A}$ is either a run without any unmoving circuit, or a run which admits such a witness.

4.2 Automaton of Crossing Sequences with One Repetition

Let $\mathcal{A} = (F, B, E, I, T)$ be a two-way $\mathbb{K}$-automaton. Let C_1 (*resp.* C_2) be the sequences of $(FB)^*F$ where each state appears at most once (*resp.* twice). Let (n, p) be the order of the finite semiring generated by the coefficients of $\mathcal{A}$. Let $\mathcal{B}$ be the one-way automaton over $\mathcal{N}_{n,p}$ with states $R = C_1 \cup C_2$ defined by:

– if c is an initial crossing sequence, then c is an initial state of $\mathcal{B}$, with weight 1 if c is in C_1 and ∞ if c is in C_2;
– if c' is a successor of c by a, then (c, a, c') is a transition of $\mathcal{B}$, with weight ∞ if c is in C_1 and c' in C_2, and weight 1 otherwise;

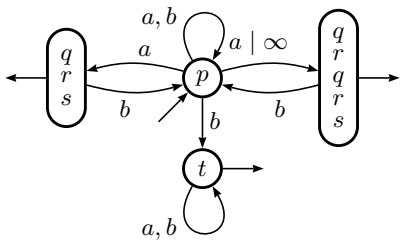


Fig. 2. The $\mathcal{N}_{n,p}$-automaton $\mathcal{B}_1$ of crossing sequences of $\mathcal{A}_1$.

– if c is a final crossing sequence, then c is final in $\mathcal{B}$, with weight 1.

The only interesting part of $\mathcal{B}$ is its trim part formed with states that can actually occur in some runs. Thus, our construction only requires to build the accessible part of $\mathcal{B}$.

Automaton $\mathcal{B}$ is a variant of the crossing sequence automaton.

On the one hand, there is a bijection between runs with weight 1 in $\mathcal{B}$ and runs without unmoving circuits in $\mathcal{A}$. On the other hand, every run in $\mathcal{B}$ with weight ∞ is the witness of an infinite number of runs in $\mathcal{A}$ ending in the same state. Notice that some runs of $\mathcal{A}$ may admit several witnesses, but all these witnesses show that there are infinitely many runs for the same final state. A run of $\mathcal{B}$ ends in a final crossing sequence; the weight of the corresponding run in $\mathcal{A}$ is given by the last state of this crossing sequence.

Example 7. Four different crossing sequences may appear in the runs of $\mathcal{A}_1$; this leads to the automaton $\mathcal{B}_1$ on Figure 2. The weight of every transition in this automaton is equal to 1, except the transition that goes to the crossing sequence with repetitions where the weight is equal to ∞. For instance, this automaton shows that there is an infinite number of runs over a in $\mathcal{A}_1$ that start in state p and stop in state s. □

4.3 Gathering Runs with the Same Label

On some locally finite semirings, the automaton $\mathcal{B}$ is sufficient to decide whether the two-way automaton is valid and to build an equivalent one-way automaton. In these cases (Case 1 and 2), the last step of the algorithm can be avoided; it leads to a simpler construction.

Case 1: Every infinite sum is defined. A non deterministic characteristic one-way $\mathbb{K}$-automaton $\mathcal{C}$ equivalent to the two-way $\mathbb{K}$-automaton can be built from the automaton $\mathcal{B}$ of crossing sequences with 1 repetition. The weight of a run is ∞ as soon as the weight ∞ is met (otherwise it is 1); this information can be stored in the states of $\mathcal{C}$ that belong to $R \times \{1, \infty\}$, where R is the set of states of $\mathcal{B}$. The accessible part of $\mathcal{C}$ is inductively defined as:

– if p is initial in $\mathcal{B}$ with weight k, then (p, k) is initial in $\mathcal{C}$;

– if (p, a, q) is a transition in $\mathcal{B}$ with weight k and (p, r) is a state of $\mathcal{C}$, then $((p, k), a, (q, k.r))$ is a transition of $\mathcal{C}$;
– if p is final in $\mathcal{B}$, then p is a final crossing sequence whose last state is final with weight x in $\mathbb{K}$; every state (p, k) of $\mathcal{C}$ is final with weight $k.x$.

Case 2: No infinite sum is defined. The two-way $\mathbb{K}$-automaton is valid if and only if no transition with weight ∞ appears in any run of $\mathcal{B}$. If there is no such transition, then the previous construction applies.

Case 3: Some infinite sums are defined, some are not. To decide whether the family of weights of all runs labelled by a given word is defined, all these runs must be gathered. We use a determinization-like algorithm to build an automaton that computes, for each word w, the number (modulo (n, p)) of runs that end in each final crossing sequence. The determinization $\mathcal{D}$ of $\mathcal{B}$ gathers the vectors corresponding to all runs on every input. Each state of $\mathcal{D}$ is a vector in $\mathcal{N}_{n,p}^{R}$, where R is the set of states of $\mathcal{B}$.

– the initial vector of $\mathcal{D}$ is I, where I_c is the initial weight of c in $\mathcal{B}$;
– if X is a state of $\mathcal{D}$, then the succesor of X by letter a is the state Y defined by:
$$Y_{c'} = \sum_{c \in R} X_c.E(c, a, c'). \tag{5}$$
– A state X is final if there exists a final crossing sequence c such that X_c is different from 0.

We consider the projection π from the final crossing sequences of R onto F which maps c onto the last state of c; this projection allows to map every final state X of $\mathcal{D}$ onto a vector v in $\mathcal{N}_{n,p}^{F}$:

$$\forall f \in F,\ v_f = \sum_{c \in R, c \text{ final}, \pi(c)=p} X_c. \tag{6}$$

For every word w, if the run on w in $\mathcal{D}$ ends in state X, then for every c, X_c is the number (modulo (n, p)) of runs on w in $\mathcal{A}$ whose last crossing sequence is c. Hence, v_f is the number (modulo (n, p)) of runs on w in $\mathcal{A}$ that end in state f.

Therefore, the weight of w in $\mathcal{A}$ is equal to the evaluation of v; if this evaluation is not defined, then the weight of w in $\mathcal{A}$ is not defined. If, for every final state X of $\mathcal{D}$, the evaluation of the corresponding vector is defined, this weight can be assigned to X as a final weight. This turns $\mathcal{D}$ to a characteristic and deterministic one-way $\mathbb{K}$-automaton equivalent to $\mathcal{A}$.

Theorem 1. *Over locally finite semirings, the validity of two-way automata is decidable, and every valid two-way automaton is equivalent to a one-way automaton.*

The construction follows the algorithm outlined in this paper. Given a two-way automaton $\mathcal{A}$ over a locally finite semiring semiring, we can first consider it as an automaton over $\mathbb{K}$, the finite semiring generated by the weights of the

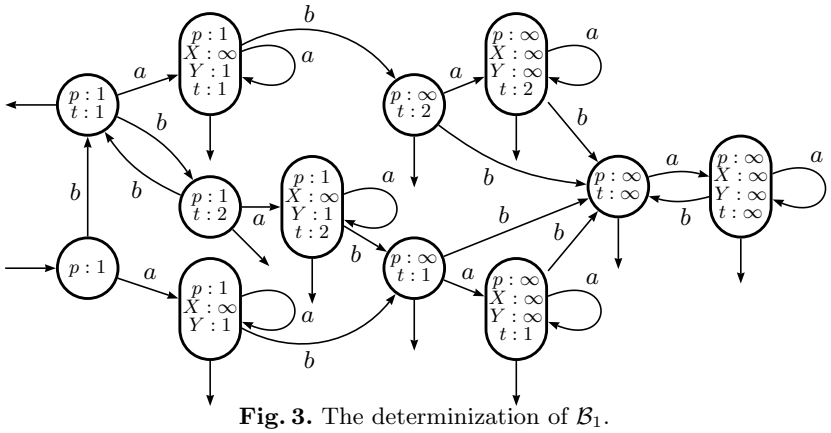


Fig. 3. The determinization of $\mathcal{B}_1$.

automaton. A δ-normalized characteristic two-way $\mathbb{K}$-automaton $\mathcal{A}'$ equivalent to $\mathcal{A}$ is then constructed. The automaton of crossing sequences with 1 repetition of $\mathcal{A}'$, $\mathcal{B}$, allows to compute the number of runs that stop in each state. The end of the algorithm depends on the cases described in Section 4.3. In Cases 1 and 2 the validity is directly decidable on $\mathcal{B}$ and a one-way $\mathbb{K}$-automaton equivalent to $\mathcal{A}$ can also be derived from $\mathcal{B}$. In Case 3, a determinization step is required. As explained above, this deterministic automaton allows both to decide the validity and to build a one-way $\mathbb{K}$-automaton equivalent to $\mathcal{A}$, if there exists one.

Remark 4. The fact that the weight of a word w is defined in $\mathcal{A}$ only depends on the state ending the run over w in $\mathcal{D}$. Hence the language of words with an undefined weight is regular.

Example 8. We denote $X = (q, r, q, r, s)$ and $Y = (q, r, s)$; hence the states of automaton $\mathcal{B}_1$ are p, t, X and Y. The states of the determinization of $\mathcal{B}_1$ are vectors in $\mathcal{N}_{n,p}^{\{p,t,X,Y\}}$; the determinization is drawn on Figure 3. Only non zero entries are written in each state. The initial state is not final since it does not contain any final crossing sequence.

We replace now each vector in $\mathcal{N}_{n,p}^{\{p,t,X,Y\}}$ by the corresponding vector in $\mathcal{N}_{n,p}^{\mathbb{K}_1}$ (the entry corresponding to o is not shown since it does not influence the evaluation). Consider for instance the vector $[p : 1, X : \infty, Y : 1, t : 1]$. p is not a final crossing sequence and does not contribute; the last state of X and Y is s with final weight x, hence, x appears with multiplicity $\infty + 1 = \infty$ and the final weight of t is i; finally, the family of weights for this state is $[x : \infty, i : 1]$ and the final weight is $\infty.x + 1.i = x$. We obtain the automaton of Figure 4. Notice that states with vector $2.i$ are not final since $i + i = o = 0_{\mathbb{K}}$. The construction fails because one state corresponds to the vector $\infty.i$ and the infinite sum of i is not defined, thus no final weight can be defined for this state. For every word leading to this state (for instance abb), there is in $\mathcal{A}_1$ an infinite number of runs and the weight of each of these runs is equal to i. Therefore, the two-way automaton $\mathcal{A}_1$ is not valid. □

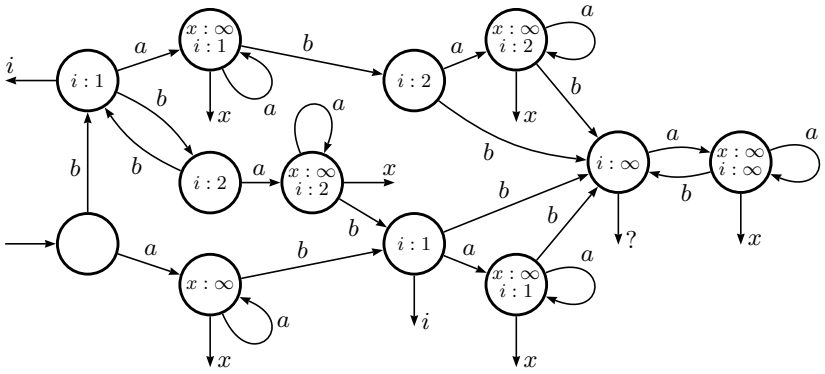


Fig. 4. The automaton $\mathcal{A}_1$ is not valid.

5 Conclusion

This paper describes an algorithm to decide whether a two-way $\mathbb{K}$-automaton is valid, and, if it is, to build an equivalent one-way $\mathbb{K}$-automaton. The construction involves several steps. The first two – getting a characteristic $\mathbb{K}$-automaton and then a δ-normalized $\mathbb{K}$-automaton – have low state complexity: respectively $|\mathbb{K}||Q|$ and $2|Q|$. In contrast, the last two have huge complexity. The state complexity of the automaton of crossing sequences with 1 repetition is in $2^{O(|Q|\log|Q|)}$ and the determinization is in $(n+p)^{|Q|}$, where Q is the set of states of the automaton on which each construction is applied and (n,p) is the order of the finite semiring of weights. Finally the state complexity of the construction is a double exponential.

This work encompasses all locally finite semirings. It is an open question to know whether there exist in general a more efficient construction. For particular classes of semirings, this construction can certainly be improved.

References

1. Birget, J.C.: Concatenation of inputs in a two-way automaton. Theor. Comput. Sci. **63**(2), 141–156 (1989). https://doi.org/10.1016/0304-3975(89)90075-3
2. Carnino, V., Lombardy, S.: On Determinism and Unambiguity of Weighted Two-way Automata. In: AFL'14. EPTCS, vol. 151, pp. 188–200 (2014)
3. Engelfriet, J., Hoogeboom, H.J.: MSO definable string transductions and two-way finite-state transducers. ACM Trans. Comput. Log. **2**(2), 216–254 (2001)
4. Leung, H.: Limitedness theorem on finite automata with distance functions: an algebraic proof. Theoret. Comp. Sci. **81**(1, (Part A)), 137–145 (1991)
5. Lombardy, S.: Weighted two-way automata. In: NCMA'15. books@ocg.at, vol. 318, pp. 37–47. Österreichische Computer Gesellschaft (2015)
6. Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM J. Res. Dev. **3**(2), 114–125 (1959). https://doi.org/10.1147/rd.32.0114
7. Shepherdson, J.C.: The reduction of two-way automata to one-way automata. IBM J. Res. Dev. **3**(2), 198–200 (1959). https://doi.org/10.1147/rd.32.0198