Two-Way Two-Tape Automata

Olivier Carton, Léo Exibard, Olivier Serre

Two-way Two-tape Automata

Olivier Carton[1,⋆], Léo Exibard[2], and Olivier Serre[1]

[1] IRIF, Université Paris Diderot & CNRS
[2] Département d'Informatique, ENS de Lyon

Abstract. In this article we consider two-way two-tape (alternating) automata accepting pairs of words and we study some closure properties of this model. Our main result is that such alternating automata are not closed under complementation for non-unary alphabets. This improves a similar result of Kari and Moore for picture languages. We also show that these deterministic, non-deterministic and alternating automata are not closed under composition.

Keywords: Alternating · Multi-tape automata · Complementation

1 Introduction

In this article we consider two-way two-tape (alternating) automata that are designed to recognize binary relations between finite words. Although these automata are quite natural as read-only Turing machines, almost no work has been devoted to this model of computation. We study their properties, with a special focus on their closure properties, in particular under complementation and composition.

Finite states machines with inputs and outputs are widely used in many different areas like coding [11], computer arithmetics [12], natural language processing [13] and program analysis [2]. The simplest model is obtained by adding outputs to a classical finite-state (non)-deterministic one-way automaton to get a machine known as a transducer. In a transducer, the input word is only scanned once by a one-way head and the output is produced by the transitions used along the reading. A transducer can equivalently be seen as a machine with two tapes, one for the input and the other for the output, that are scanned once by two one-way heads. Relations realized by this kind of machines are called rational. They have been intensively studied since the early days of automata theory [3] and they enjoy some nice properties [14]. They are, for instance, closed under composition, but not under complementation.

Rational relations turn out to form a rather small class, hence classes of stronger transducers have been introduced by enriching transducers with extra features like two-wayness and/or alternation. A well studied class is that of two-way transducers in which the input word is scanned by a two-way head and the output is produced (or equivalently scanned) by a one-way head. In that

⋆ funded by the DeLTA project (ANR-16-CE40-0007)

class, deterministic machines are of special interest as they are equivalent to MSO-transductions and turn out to be closed under composition [4].

In this article, we consider machines with two tapes which are both scanned by two-way heads. In this model, it is important to consider the output word as already written, and not produced on some output tape to ensure consistency, as it is scanned several times.

Another key feature, used to increase either the expressive power or the succinctness of finite state machines, is alternation, which allows the machine to spawn several copies of itself. Alternation often provides for free closure under complementation for machines on finite structures like words or trees. Indeed, the dual machine obtained by swapping existential and universal states and complementing the acceptance condition accepts the complement language as long as all computations terminate: if the run of a machine may loop, the closure under complementation of alternating machines may no longer hold.

Picture automata introduced in [1] scan a 2-dimensional array of symbols with moves in the four cardinal directions to either accept or reject it. As in the case of two-way automata, the border of the array is marked by special symbols. A run of such an automaton may loop as it can scan the same position twice with the same control state. These picture automata differ from classical word or tree automata where all variants (deterministic, non-deterministic, alternating) are equivalent. Indeed, it has been shown by Kari and Moore that alternating picture automata are not closed under complementation as soon as the alphabet size is greater than 1 [8]. This means that loops are inherent to the model and that they cannot be removed.

In this article, we show that two-tape two-way alternating automata are not closed under complementation either. Picture automata are actually very close to the model that we consider. In particular they coincide for unary alphabets. Indeed, over a unary alphabet, a pair of words is merely a pair of integers (their lengths) and this is equivalent to a two-dimensional array on a unary alphabet. However, as soon as the alphabets have cardinality at least 2, the models are distinct. Indeed, for an alphabet of size k, the number of $m \times n$-arrays is k^{mn} while the number of pairs of words is only k^{m+n}.

The fact that the two models coincide for unary alphabets allows us to recover immediately some separation and undecidability properties. For instance, it has been shown that such deterministic picture automata are strictly less powerful than non-deterministic ones which are, in turn, less powerful than alternating automata [8]. These results carry over the two-tape two-way automata that we consider.

To prove that alternating automata are not closed under complementation, we use a counter-example that is close to the one in [9] for picture automata. However, since our coding is different, we need some extra arguments to show that it is accepted by an alternating automaton and, to show that its complement cannot, we use a more direct proof.

2 Two-way Two-tape Automata

In this article, Σ is a finite alphabet and Σ^* denotes the set of finite words over Σ. For a word $u \in \Sigma^*$, we denote its length by $|u|$, and for each $1 \leq i \leq |u|$ we denote by u_i its i-th letter. From now on, $\vdash$ (begin) and $\dashv$ (end) are reserved characters not belonging to Σ and marking word boundaries. For simplicity of notation, we let $\Sigma_{\vdash}^{\dashv} = \Sigma \cup \{\vdash, \dashv\}$. For $u \in \Sigma^*$, we let $u_{\vdash}^{\dashv} = \vdash u \dashv$, with $u_{\vdash 0}^{\dashv} = \vdash$, $u_{\vdash |u|+1}^{\dashv} = \dashv$ and $u_{\vdash i}^{\dashv} = u_i$ for each $1 \leq i \leq |u|$.

A *(non-deterministic) two-way two-tape finite automaton* is a tuple $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, where Q is the set of *states*, and $I, F \subseteq Q$ are respectively the sets of *initial* and *final* states. We call $\Delta \subseteq \left(Q \times \Sigma_{\vdash}^{\dashv} \times \Sigma_{\vdash}^{\dashv}\right) \times (Q \times \{\lhd, \nabla, \rhd\} \times \{\lhd, \nabla, \rhd\})$ the *transition relation*. We use the notation $p \xrightarrow{a_1,a_2|d_1,d_2} q$ for $(p, a_1, a_2, q, d_1, d_2) \in \Delta$. We require that the reading heads cannot cross the words boundaries, i.e. for every transition $p \xrightarrow{a_1,a_2|d_1,d_2} q$ and every $i = 1, 2$ if $a_i = \vdash$ (*resp.* $a_i = \dashv$) then $d_i \neq \lhd$ (*resp.* $d_i \neq \rhd$).

An automaton $\mathcal{A}$ is said to be *deterministic* whenever for every state $p \in Q$ and every letters $a_1, a_2 \in \Sigma$, there exists at most one $q \in Q, d_1, d_2 \in \{\lhd, \nabla, \rhd\}$ such that $p \xrightarrow{a_1,a_2|d_1,d_2} q$.

Note that extending the model to more than two tapes is straightforward. Note also that if we restrict the model to a single tape we retrieve the classical notion of two-way automata on finite words.

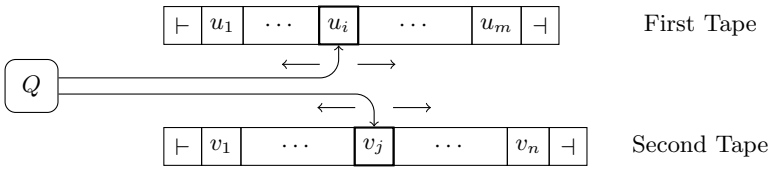

Fig. 1. Schema of a two-way two-tape finite automaton

A *configuration* is a triple $(q, i, j) \in Q \times \mathbb{N} \times \mathbb{N}$, where q is the current state, and i (resp. j) is the position of the reading head on the first (resp. second) tape (recall that the $\vdash$ marker is by convention at position 0).

We say that a configuration (q_2, i_2, j_2) is a *successor* of a configuration (q_1, i_1, j_1) with regard to input (u, v), written $(q_1, i_1, j_1) \overset{(u,v)}{\rightsquigarrow} (q_2, i_2, j_2)$, when automaton $\mathcal{A}$ can go from one configuration to the next in a single step, i.e. if $q_1 \xrightarrow{(u_{\vdash}^{\dashv})_{i_1},(v_{\vdash}^{\dashv})_{j_1}|d,e} q_2$, and if $i_2 = i_1 + \chi(d)$ and $j_2 = j_1 + \chi(e)$, where $\chi(\lhd) = -1$, $\chi(\nabla) = 0$ and $\chi(\rhd) = 1$.

A *run* of $\mathcal{A}$ on input $(u, v) \in \Sigma^* \times \Sigma^*$ is a (possibly infinite) sequence $(p_k, i_k, j_k)_{1 \leq k < n}$, where $n \in \mathbb{N} \cup \{\infty\}$, of successive configurations: for every $1 \leq k < n$, one has $(p_k, i_k, j_k) \overset{(u,v)}{\rightsquigarrow} (p_{k+1}, i_{k+1}, j_{k+1})$.

An *initial run* is a run starting with an initial configuration, i.e. one such that $p_0 \in I$ and $i_0 = j_0 = 0$. It is *accepting* when it is moreover finite and contains a final state $f \in F$, i.e. there exists $k \leq n$ such that $p_k \in F$.

The *relation* $\mathcal{R}(\mathcal{A})$ *accepted by* a two-way two-tape automaton $\mathcal{A}$ is the set of pairs (u, v) such that there exists an accepting run of $\mathcal{A}$ on input (u, v).

We now introduce alternating automata, which generalize non-deterministic automata. An *alternating automaton* is an automaton whose set of control states is partitioned into *existential* ($Q_\exists$) and *universal* ($Q_\forall$) states. A configuration is defined as in the non-deterministic setting and it is existential (*resp.* universal) if the control state is.

Runs of alternating automata are (possibly infinite) trees whose nodes are labeled by configurations, and such that each inner node u labeled by a configuration C satisfies the following conditions:

- If C is existential then u has a single son that is labeled by a successor configuration of C.
- If C is universal and if $\{C_1, \dots, C_k\}$ denotes all successor configurations of C, then u has k sons each of them labeled by a different C_i for $1 \leq i \leq k$.

A run is *accepting* if it is finite, its root is labeled by an initial configuration and all its leaves are labeled either by accepting configurations, or by universal configurations that have no successor configuration. Again, we define the relation accepted by an alternating automaton as those pairs of words over which there is an accepting run. Note that non-deterministic automata correspond to the case where $Q_\forall = \emptyset$.

First we remark that if we restrict the model by forbidding the reading heads to go to the left, we obtain a 1-way model that recognizes the rational relations, i.e. those realized by transducers [14]. Not surprisingly this is a restriction as two-way two-tape automata can, for instance, recognize deterministically the relation $\{(u, \tilde{u}) \mid u \in \Sigma^*\}$, where $\tilde{u}$ denotes the reverse of u. The corresponding automaton is depicted in Figure 2. This relation cannot be recognized by classical one-way transducers.

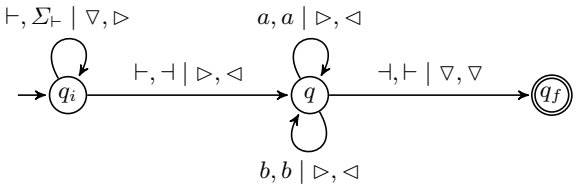

Fig. 2. Automaton recognizing $\{(u, \tilde{u}) \mid u \in \Sigma^*\}$ when $\Sigma = \{a, b\}$

Our model captures much more complex relations. It is shown in the next section that two-way two-tape automata can recognize the relation Coprime $= \{(a^p, a^q) \mid p \wedge q = 1\}$, where $p \wedge q$ denotes the greatest common divisor of p and q, by implementing a variant of the Euclidean algorithm.

3 Picture Languages

For brevity we simply recall here some key notions on picture languages: we refer the reader to [5] for an excellent survey on the topic with complete definitions of all objects discussed below.

A *picture* p of dimensions $m \times n$ is a matrix over a finite alphabet Σ. For every $1 \leq i \leq m$ and $1 \leq j \leq n$, we write $p_{i,j}$ for the content of the cell at position (i, j). To recognize pictures, we add a special marker $\# \notin \Sigma$ all around the picture p, i.e. we adopt the convention that for all $0 \leq i \leq m+1$ and $0 \leq j \leq n+1$, $p_{0,j} = p_{m+1,j} = p_{i,0} = p_{i,n+1} = \#$. We write Σ^{**} for the set of all pictures over Σ. A *picture language* is thus a subset of Σ^{**}.

In order to recognize picture languages, 4-way automata were first introduced in [1]. Such an automaton has a single head which is able to move on a two-dimensional array of symbols (surrounded by markers) in the four directions (up, down, left, right) and accepts when reaching a final state. A schema is provided in Figure 3.

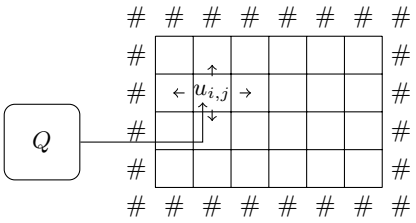

Fig. 3. Schema of a 4-way automaton

The link between two-way two-tape automata and picture automata is provided by special pictures, called *products of words*, that we now define. For $u, v \in \Sigma^*$, we define the picture $u \otimes v = ((u_i, v_j))_{\substack{1 \leq i \leq |u| \\ 1 \leq j \leq |v|}}$ over the product alphabet $\Sigma \times \Sigma$.

Thus, any relation $\mathcal{R} \subseteq \Sigma^* \times \Sigma^*$ is mapped to a picture language $L_{\mathcal{R}}^{\otimes} \subseteq (\Sigma \times \Sigma)^{**}$. Moreover, over unary alphabets, pictures languages and binary relations over words are in one-to-one correspondance as the pair (a^m, a^n) unambiguously represents an image of dimensions $m \times n$ and conversely. Consequently, for $L \subseteq \{a\}^{**}$, L is recognizable by a 4-way deterministic (resp. non-deterministic, alternating) automaton *iff* $\mathcal{R}$ is recognizable by a deterministic (resp. non-deterministic, alternating) two-way two-tape automaton where $\mathcal{R}$ is the unique relation such that $L = L_{\mathcal{R}}^{\otimes}$.

Thus, all the results known for unary picture languages also hold in our model when $|\Sigma| = 1$. In particular, in [8], it is shown that determinism is strictly weaker than non-determinism, the latter being weaker than alternation: DFA $\subsetneq$ NFA $\subsetneq$ AFA where DFA (resp. NFA, AFA) is the class of relations recognized by deterministic (resp. non-deterministic, alternating) two-way two-tape automata.

The relation $\mathcal{R} = \{(a^w, a^h) \mid \exists i, j \in \mathbb{N}, w = ih + j(h+1)\}$ is such that $\mathcal{R} \notin$ DFA, $\mathcal{R} \in$ NFA, $\overline{\mathcal{R}} \notin$ NFA, $\overline{\mathcal{R}} \in$ AFA, where $\overline{\mathcal{R}}$ denotes the complement of $\mathcal{R}$ in $\Sigma^* \times \Sigma^*$. In [10], it is shown that the emptiness problem is undecidable even for a unary alphabet.

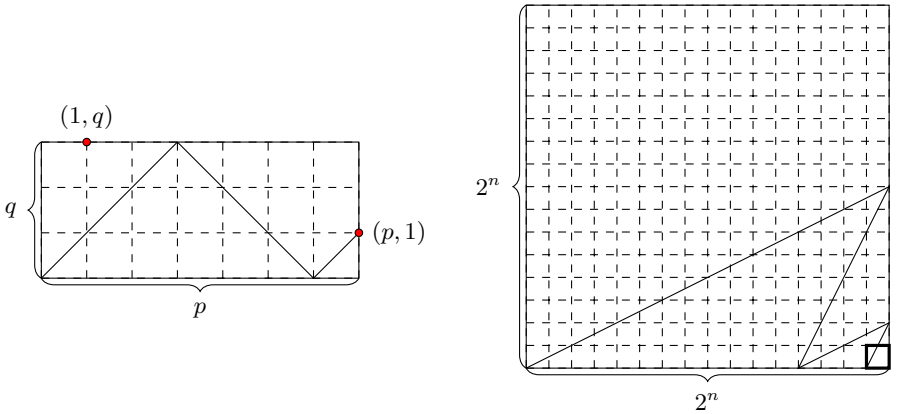

Fig. 4. Euclidean algorithm (left) and Squares of size 2^n (right)

The following examples from [9] show that picture automata and two-way two-tape automata are really expressive. The relation Coprime $= \{(a^p, a^q) \mid p \wedge q = 1\}$ can be recognized by a deterministic automaton implementing a variant of the Euclidean algorithm. The automaton follows diagonals until it reaches either one of $(p, 1)$ or $(1, q)$ or one of $(0, 0)$, $(p, 0)$, $(0, q)$ or (p, q). In the former case, it accepts and in the latter case, it rejects.

A more sophisticated deterministic automaton can recognize the relation $\{(2^n, 2^n) \mid n \in \mathbb{N}\}$ following the schema in Figure 4 (right). The automaton moves with a ratio of $1/2$ as long as it is possible (i.e. while the remaining length is divisible by 2), and it accepts if it reaches the bottom-right square. An even more sophisticated deterministic automaton can accept the relation $\{(2^{2^n}, 2^{2^n}) \mid n \in \mathbb{N}\}$.

4 Alternating Two-way Two-tape Automata Are Not Closed under Complementation

Our main result is the following. Note that the case of unary alphabets is still an open problem.

Theorem 1. *The class of relations recognized by alternating two-way two-tape automata is not closed under complementation as soon as the alphabet is not unary.*

In this proof we denote by AFA the class of relations recognized by alternating two-way two-tape automata, and by co-AFA the class of relations whose complement is recognized by alternating two-way two-tape automata. Hence, we aim to prove that AFA and co-AFA are distinct, and this is achieved by defining a well-chosen relation $\mathcal{P}$ and show that $\mathcal{P} \in$ AFA but $\mathcal{P} \notin$ co-AFA. The first point is proved by giving an explicit alternating two-way two-tape automaton recognizing $\mathcal{P}$ (Lemma 2); the second point is proved by contradiction using a game-theoretic approach combined with a combinatorial argument (Lemma 3).

The proof is inspired by the one in [8] establishing that the set of picture languages recognized by alternating 4-way automata is not closed under complementation. One difference is that the counter-example they exhibit cannot be used in our setting as the images that are considered are not product of words (as defined in Section 3). Hence, even if the general idea — coding a permutation to build a counter-example — is similar to the one in [8], our coding is different and therefore new ideas are needed to prove that we can accept this relation with an alternating two-way two-tape automaton. The proof that the complement is not recognizable by an alternating two-way two-tape automaton, even if it shares ideas with the one in [8], is somehow more direct as it does not appeal to an intermediate class.

In the following, if $c_1, \ldots, c_n$ are n words of length n over alphabet $\{0, 1\}$ we identify the tuple $(c_1, \ldots, c_n)$ with the $n \times n$ matrix whose i-th column is c_i. We define the relation $\mathcal{P}$ by

$$\mathcal{P} = \{(a^n, c_1\#\cdots\#c_n\$c_1\#\cdots\#c_n) \mid n \in \mathbb{N}, (c_1, \ldots, c_n) \in \mathfrak{S}_n\}$$

where $\mathfrak{S}_n$ denotes the set of all ($n \times n$ $\{0,1\}$-matrix coding) permutations over $\{1, \ldots, n\}$. See Figure 5 for an example.

0 0 1 $ 0 0 1	$(a^3, 010\#001\#100\$010\#001\#100)$
1 0 0 $ 1 0 0	
0 1 0 $ 0 1 0	

Fig. 5. Two identical permutations separated by $s and the corresponding encoding

As announced, we start by showing that $\mathcal{P} \in$ AFA.

Lemma 2. *There exists an alternating two-way two-tape automaton recognizing $\mathcal{P}$.*

Proof. We describe below how to check whether a pair $(a^n, c) \in a^* \times \{0, 1, \$, \#\}^*$ is in $\mathcal{P}$. As the a^n word is used only to store position we sometimes refer to the content of the first tape as the *counter*.

To decide if $(a^n, c) \in \mathcal{P}$, we have to check two things. The first one is whether c is of the form $\sigma_1\$\sigma_2$, where σ_1, σ_2 are the encodings of two permutations of dimension n and the second one is whether $\sigma_1 = \sigma_2$.

For the first step, we only explain how to check the property for σ_1 as the case of σ_2 is checked in the same way. Assume $\sigma_1 = c_0\#c_1\#\cdots\#c_n$ (checking that there are n block is easy). We first need to check that each c_i (i.e. each column) has length n and contains exactly one 1: this is easy (the length condition being checked thanks to the first tape). We then need to check that for each $k = 1, \ldots, n$ there is exactly one c_i whose k-th letter is a 1 (i.e. each row contains exactly one 1). This property is checked for each $k = 1, \ldots, n$ in increasing order starting from $k = 1$, and at the beginning of step k the head in the counter in the first tape is at position k and the head in the second tape is at the beginning of σ_1: now going left on the counter and right on the second tape at the same speed the k-th symbol of c_1 is reached (just before the left marker is read on the first tape); then the automaton goes back to the beginning of c_1 and the counter is increased back to k; finally the second head goes to the beginning of c_2 (going to the right until reading a #), then c_2 is processed in the same way, and so on until reading a \$ meaning that c_n was processed and that one can go back to the beginning of the second tape and start again but now for $k + 1$.

We are now left with checking that $\sigma_1 = \sigma_2$ knowing that both σ_1 and σ_2 encode a permutation. For that we use the same approach as the one in [8, Lemma 2, Condition (*)]. More precisely, we use the following property: two permutations are different *iff* there exists an inversion i.e. there exists $i \neq i', \sigma_1(i) < \sigma_1(i')$ but $\sigma_2(i) > \sigma_2(i')$. This can be checked by trying all possible way of moving in the associated picture (as illustrated in Figure 5) in the following fashion (see Figure 6): from a 1 move right to another column but stay on the same side of the column of \$s. Find the 1 on that column. Then move to the other 1 that is on the same row, on the opposite side of the column of \$s, and repeat: then, the machine enters an infinite loop *iff* it finds an inversion [8].

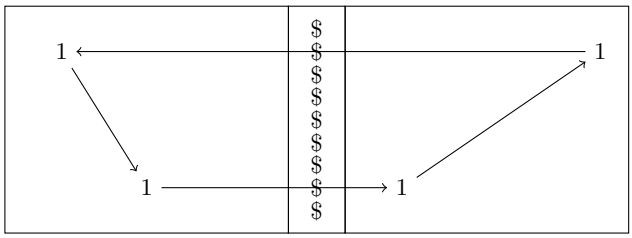

Fig. 6. Loop corresponding to an inversion

The first step (finding a 1) is easy and done using universal states (which ensures that we do the check for all possible 1). Moving to another column on the right is easy: simply use a universal state and keep moving the head to the right passing one or more # but stopping before reading the \$. Finding the 1 on the column is easy (look for it in between the left and right #). Finally, moving to the other 1 on the same row on the opposite side of the \$ is achieved by using the counter to keep in memory the row number j, and then go to the other side

and check every column until finding the one with a 1 on the j-th row (this is done with the same trick used previously to check that each row contains a single 1). If at some point when moving to a column to the right the automaton hit a \$ then it goes to a final state. The automaton accepts the desired language for the following reason: if an inversion exists, it is found (thanks to universal choices) and leads to a looping hence, rejecting, computation; if not, whatever the universal choices are the automaton is getting closer and closer to the \$ and eventually hit it thus reaching a final state.

Therefore, we built an alternating two-way two-tape automaton recognizing $\mathcal{P}$. More generally, the ideas used in the first part of the proof can be used to prove the following: for any picture language L recognizable by a 4-way automaton, the relation $\{(a^{\max(n,m)}, c_1\# \cdots \#c_m) \mid (c_1, \ldots, c_m)$ is an $n \times m$ picture in $L\}$ is recognizable by a two-way two-tape automaton. □

We now show that the complement of $\mathcal{P}$ cannot be recognized by an alternating two-way two-tape automaton.

Lemma 3. *Relation* $\mathcal{P} \notin$ co-AFA.

Proof. The proof is by contradiction assuming the existence of an alternating two-way two-tape automaton $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ such that a pair (u, v) is accepted by $\mathcal{A}$ *iff* $(u, v) \notin \mathcal{P}$. Hence, a pair (a^n, c) is rejected by $\mathcal{A}$ *iff* if belongs to $\mathcal{P}$.

We now rephrase the fact that (a^n, c) is rejected by $\mathcal{A}$, to that end we use a game-theoretic flavoured argument. Indeed, another way of thinking about a run of an alternating automaton is as a two-player games where the existential player is in charge of choosing the transition in existential configurations while the universal player takes care of universal configurations. The winning condition for the existential player is that a final configuration is eventually reached. In particular $\mathcal{A}$ rejects its input *iff* the universal player has a winning strategy, i.e. a playing policy ensuring no final state is ever reached whatever the existential player does. Moreover, this strategy can be chosen to be positional, i.e. only depending on the configuration, not on how it was reached (see *e.g.* [6]).

Fix an input of the form $(a^n, \sigma\$\sigma) \in \mathcal{P}$. A strategy for the universal player and the proof that it is a winning one can be described as follows. For each position $0 \leq j \leq 2n+2$ on the second tape (including the markers) we associate a $n+2$ tuple $\tau_j = (\tau_{0,j}, \ldots, \tau_{n+1,j})$ of functions describing what configurations can appear when the heads are at position (i, j) and for universal ones what the strategy of the universal player is. Hence for each $0 \leq i \leq n+1$, $\tau_{i,j}$ is a map from Q such that:

- if $q \in Q_\exists$, $\tau_{i,j}(q)$ is either $\bot$ (meaning (q, i, j) is not reachable) or $\top$ (reachable)
- if $q \in Q_\forall$, $\tau_{i,j}(q)$ is either $\bot$ (not reachable) or a transition in Δ starting with (q, x, y) where x (*resp.* y) is the letter at position i (*resp.* j) in the first (*resp.* second) tape, including markers. The latter case means that configuration (q, i, j) can appear and gives the corresponding move in the universal player strategy.

Such an object $\tau_0, \ldots, \tau_{2n+2}$ is a *proof of reject* if it satisfies the following four conditions.

1. For each initial state $q \in I$, $\tau_{0,0}(q) \neq \bot$ (all initial configurations are allowed).
2. For each final state $q \in F$ and each i, j, $\tau_{i,j}(q) = \bot$ (no final configuration can be reached).
3. For each existential state q and each i, j such that $\tau_{i,j}(q) = \top$, for each possible successor (q', i', j') of (q, i, j) one has that $\tau_{i',j'}(q') \neq \bot$ (all successors of allowed existential configurations are allowed).
4. For each universal state q and each i, j such that $\tau_{i,j}(q) \neq \bot$, if one denotes by (q', i', j') the configuration reached from (q, i, j) applying transition $\tau_{i,j}(q)$ then $\tau_{i',j'}(q') \neq \bot$ (an allowed universal configurations has its successor described by the strategy allowed).

It is then standard to notice that $\mathcal{A}$ has no accepting run over $(a^n, \sigma\$\sigma)$ *iff* there is a proof of reject for it.

Now, for a fixed n, consider the number of possible values for the central part $(\tau_n, \tau_{n+1}, \tau_{n+2})$ of a proof: it is smaller than $((|\Delta| + 1)^{|Q|})^{3n}$. Hence, for n large enough it is smaller than $n!$. For such n it means that there are two distinct permutations $\sigma \neq \sigma'$ such that the proofs of reject for $\sigma\$\sigma$ and for $\sigma'\$\sigma'$ coincide on their central part: hence, they can be combined (glue the left part of the first proof with the right part of the second), leading a proof of reject for $\sigma\$\sigma'$. But this leads a contradiction as $\sigma\$\sigma' \notin \mathcal{P}$ and therefore is accepted by $\mathcal{A}$. □

5 Union, Intersection and Composition

In this section, we study the closure under union, intersection and composition of two-way two-tape automata.

5.1 Closure under Union and Intersection

Concerning closure under union and intersection we have the following picture.

Lemma 4. *Relations recognized by deterministic (*resp. *non-deterministic,* resp. *alternating) two-way two-tape automata are closed under union and intersection.*

Proof (sketch). For non-deterministic automata union is for free; intersection is simply obtained by simulating the first automaton, and if it reaches an accepting state, simulating the second automaton. For alternating automata both closures are for free. For deterministic automata, intersection can be obtained as in the non-deterministic case. Concerning union, one can no longer use non-determinism and hence, one needs to simulate successively the two automata. However this does not work in general as a two-way two-tape automaton can reject by entering an infinite loop. Hence, one must first get rid of such phenomenon: this can be achieved thanks to a method due to Sipser [15], which ensures that a deterministic two-way two-tape automaton never rejects by looping. □

Remark that preventing deterministic two-way two-tape automata from rejecting by looping can be used to prove that the class of relations they recognize is also closed under complementation, and therefore forms a Boolean algebra.

5.2 Non-closure under Composition

In this section, we prove that the class of relations recognized by two-way two-tape automata is not closed under composition, even in the deterministic case.

Theorem 5. *Relations recognized by deterministic (*resp. *non-deterministic,* resp. *alternating) two-way two-tape automata are not closed under composition.*

Proof. The proof works the same way for all three models. One first establishes (see Lemma 6 below) an elementary bound on the growth of the functions recognized by our model, and then exhibit (see Lemma 7 below) a recognizable relation breaking this bound when self-composed. □

We now give the lemmas used in the proof of Theorem 5. The following lemma bounds the growth of the functions, that is functional relations recognized by two-way two-tape automata (a similar statement can easily be obtained for alternating two-way two-tape automata with a doubly exponential bound instead).

Lemma 6. *If f is a function recognized by a two-way two-tape automaton, then there exists $k \in \mathbb{N}$ such that for all $n \in \mathbb{N}$ and $u \in \Sigma^n$, one has $|f(u)| \leq \binom{2nk}{nk+1}$, where $\binom{n}{k}$ denotes the binomial coefficient.*

Proof. Let $\mathcal{A}$ be a two-way two-tape automaton recognizing a function f. Let $n \in \mathbb{N}$, and $u \in \Sigma^n$. Finally, let k be the size of $\mathcal{A}$. Fixing u, we easily build from $\mathcal{A}$ a two-way automaton with nk states accepting the singleton language $\{f(u)\}$. Now, thanks to a result by Kapoutsis [7] establishing that any m-state two-way automaton can be transformed into an equivalent equivalent non-deterministic finite automaton with $\binom{2m}{m+1}$ states, we have that $\{f(u)\}$ is accepted by a non-deterministic finite automaton with $\binom{2nk}{nk+1}$ states. As the shortest word accepted by an m-state non-deterministic finite automaton has length at most m, it concludes the proof. □

For $i \in \mathbb{N}$ and $n \geq \log_2(i)$ let $\langle i \rangle_n \in \{0,1\}^n$ be the writing of i in base 2 on n bits, with the less significant bit on the left (possibly padded with zeros). For example, $\langle 6 \rangle_5 = 01100$. We are now ready to exhibit a function with an exponential growth and recognized by a deterministic two-way two-tape automaton.

Lemma 7. *The function $\mathcal{E} = \{(w, \#w_0\#w_1\#\ldots\#w_{2^{|w|}-1}\#) \mid w_i = \langle i \rangle_{|w|}\}$ is recognizable by a deterministic two-way two-tape automaton.*

The proof is based on the concept of so-called synchronous relations. A relation is *synchronous* if it is recognized by a synchronous transducer, which can be simulated by a two-way two-tape automaton whose two heads always move simultaneously to the right and which accepts once the two inputs words are

entirely scanned. An example of such a relation is $\mathcal{I} = \{(\langle i \rangle_n, \langle i+1 \rangle_n) \mid n \geq 1 \text{ and } 0 \leq i \leq 2^n - 1\}$ since an integer can deterministically be incremented starting from its less significant digit. Note that a synchronous transducer can always be made deterministic as it is a classical automaton over the product alphabet.

Lemma 8. *Let $\mathcal{R}$ be a synchronous relation. Then $\{(w, \#u\#v\#) \mid (u,v) \in \mathcal{R}, |w| = |u| = |v|\}$ is recognized by a deterministic two-way two-tape automaton.*

The rough idea to prove Lemma 8 is to use the first tape to keep track of the position when alternatively reading the u and v part of the word written on the second tape.

Now, using Lemma 8 with relation $\mathcal{I}$ we conclude that the relation $\mathcal{J} = \{(w, \# \langle i \rangle_{|w|} \# \langle i+1 \rangle_{|w|} \# \mid 0 \leq i \leq 2^{|w|} - 1\}$ is recognizable by a deterministic two-way two-tape automaton.

Finally, $\mathcal{E}$ is recognized by iterating the methods for $\mathcal{J}$ (this is made possible thanks to the presence of #) and stopping once the second word is only composed of 1s.

References

1. Blum, M., Hewitt, C.: Automata on a 2-dimensional tape. In: 8th Annual Symposium on Switching and Automata Theory (SWAT 1967). pp. 155–160. IEEE (1967)
2. Cohen, A., Collard, J.F.: Instance-wise reaching definition analysis for recursive programs using context-free transductions. In: International Conference on Parallel Architectures and Compilation Techniques. pp. 332–339. IEEE (1998)
3. Elgot, C.C., Mezei, J.E.: On relations defined by generalized finite automata. IBM Journal Res. and Dev. 9, 47–68 (1965)
4. Engelfriet, J., Hoogeboom, H.J.: MSO definable string transductions and two-way finite-state transducers. ACM Trans. Comput. Log. 2(2), 216–254 (2001)
5. Giammarresi, D., Restivo, A.: Recognizable picture languages. International Journal of Pattern Recognition and Artificial Intelligence 6, 241–256 (1992)
6. Grädel, E., Thomas, W., Wilke, T. (eds.): Automata, Logics, and Infinite Games: A Guide to Current Research, LNCS, vol. 2500. Springer (2002)
7. Kapoutsis, C.: Removing bidirectionality from nondeterministic finite automata. In: Mathematical Foundations of Computer Science, 30th International Symposium (MFCS 2005). LNCS, vol. 3618, pp. 544–555. Springer (2005)
8. Kari, J., Moore, C.: New results on alternating and non-deterministic two-dimensional finite-state automata. In: 18th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2001). LNCS, vol. 2010, pp. 396–406. Springer (2001)
9. Kari, J., Moore, C.: Rectangles and squares recognized by two-dimensional automata. In: Theory Is Forever. LNCS, vol. 3113, pp. 134–144. Springer (2004)
10. Kari, J., Salo, V.: A survey on picture-walking automata. In: Algebraic Foundations in Computer Science. LNCS, vol. 7020, pp. 183–213. Springer (2011)
11. Lind, D., Marcus, B.: An Introduction to Symbolic Dynamics and Coding. Cambridge University Press (1995)

12. Lothaire, M.: Algebraic Combinatorics on Words, chap. 7, pp. 230–268. Cambridge University Press (2002)
13. Roche, E., Schabes, Y.: Finite-State Language Processing, chap. 7. MIT Press, Cambridge (1997)
14. Sakarovitch, J.: Elements of Automata Theory. Cambridge University Press (2009)
15. Sipser, M.: Halting space-bounded computations. Theoretical Computer Science 10(3), 335–338 (1980)