City Research Online

City, University of London Institutional Repository

Citation: Daviaud, L., Jurdzinski, M., Lazic, R., Mazowiecki, F., Perez, G. A. & Worrell, J. (2021). When are Emptiness and Containment Decidable for Probabilistic Automata?. Journal of Computer and System Sciences, 119, pp. 78-96. doi: 10.1016/j.jcss.2021.01.006

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/25690/

Link to published version: https://doi.org/10.1016/j.jcss.2021.01.006

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

When are Emptiness and Containment Decidable for Probabilistic Automata?

Laure Daviaud[a], Marcin Jurdziński[b], Ranko Lazić[b], Filip Mazowiecki[c], Guillermo A. Pérez[d], James Worrell[e]

[a] *CitAI, City University of London, UK*
[b] *University of Warwick, Coventry, UK*
[c] *Université de Bordeaux, Bordeaux, France*
[d] *University of Antwerp, Antwerp, Belgium*
[e] *University of Oxford, Oxford, UK*

Abstract

The emptiness and containment problems for probabilistic automata are natural quantitative generalisations of the classical language emptiness and inclusion problems for Boolean automata. It is well known that both problems are undecidable. In this paper we provide a more refined view of these problems in terms of the degree of ambiguity of probabilistic automata. We show that a gap version of the emptiness problem (that is known to be undecidable in general) becomes decidable for automata of polynomial ambiguity. We complement this positive result by showing that the emptiness problem remains undecidable even when restricted to automata of linear ambiguity. We then turn to finitely ambiguous automata. Here we give a conditional decidability proof for containment in case one of the automata is assumed to be unambiguous while the other one is allowed to be finitely ambiguous. Part of our proof relies on the decidability of the theory of real exponentiation, which has been shown, subject to Schanuel's Conjecture, by Macintyre and Wilkie.

Keywords: Probabilistic automata, Emptiness, Containment, Ambiguity

1. Introduction

Probabilistic automata (PA) are a quantitative extension of classical Boolean automata that were first introduced by Rabin [1]. In this model, non-deterministic choices are replaced by probabilities: each transition carries a rational number which gives its probability to be chosen amongst all the other transitions going out of the same state and labelled by the same letter. Then, instead of simply accepting or rejecting a word, such an automaton measures the probability of it being accepted.

PA can be seen as (blind) partially observable Markov decision processes [2]. PA are also closely related to hidden Markov models, which are finite-state models for generating probability distributions over strings [3]. Such probabilistic

Email addresses: laure.daviaud@city.ac.uk (Laure Daviaud), Marcin.Jurdzinski@warwick.ac.uk (Marcin Jurdziński), R.S.Lazic@warwick.ac.uk (Ranko Lazić), filipm@mpi-sws.org (Filip Mazowiecki), guillermoalberto.perez@uantwerpen.be (Guillermo A. Pérez), James.Worrell@cs.ox.ac.uk (James Worrell)

finite-state machines have numerous applications in the field of artificial intelligence [4, 5]. Further applications for PA include, amongst others, verification of probabilistic systems [6, 7, 8], reasoning about inexact hardware [9], quantum complexity theory [10], uncertainty in runtime modelling [11], as well as text and speech processing [12]. PA are very expressive, as witnessed by the applications described above, and it is thus not surprising that most natural verification-related decision problems for them are undecidable. (However, equivalence and minimisation do admit efficient algorithms [13, 14].) Due to these negative results, many sub-classes of probabilistic automata have been studied. These include, for example, hierarchical [15], leaktight [16] and bounded-ambiguity automata [17] (see [18] for a survey).

We focus on the emptiness and containment problems for PA, which are natural analogs of the like-named problems for Boolean automata. The emptiness problem asks: *given an automaton $\mathcal{A}$, determine whether there exists a word w such that the probability $[\![\mathcal{A}]\!](w)$ of it being accepted is strictly greater than $\frac{1}{2}$.*[1] The containment problem asks: *given two automata $\mathcal{A}$ and $\mathcal{B}$, determine whether for all words w it holds that $[\![\mathcal{A}]\!](w) \leq [\![\mathcal{B}]\!](w)$.* The emptiness problem (also called the threshold problem) has long been known to be undecidable [19]. It directly follows that containment, being a generalisation of emptiness, is also undecidable. Undecidability of both problems holds even for automata with a fixed number of states [20], while in case of a unary alphabet decidability of emptiness is equivalent to longstanding open problems in number theory [21].

Remarkably, even the following gap version of the emptiness problem is also undecidable [22]. The *gap emptiness problem* takes as input a rational $\varepsilon \in \mathbb{Q} \cap (0, 1)$ and a probabilistic automaton $\mathcal{A}$ such that either there is a word w satisfying $[\![\mathcal{A}]\!](w) > \frac{1}{2} + \varepsilon$ or $[\![\mathcal{A}]\!](w) \leq \frac{1}{2}$ for all words w. The task consists in determining whether $[\![\mathcal{A}]\!](w) \leq \frac{1}{2}$ for all words w. Thus the gap problem is a promise problem [23] that can be seen as the decision analog of the problem of approximating $\sup_{w \in \Sigma^*} [\![\mathcal{A}]\!](w)$ to within given precision ε. In contrast to the situation with probabilistic automata, the results of [24] imply that the analogous gap problem for quantum automata is decidable.

In this paper we undertake a systematic analysis of the decidability of emptiness and containment for PA in terms of their *ambiguity*. The latter is a well-studied structural restriction on automata (a definition follows) which usually lowers the complexity of decision problems. For instance, in [25] it is shown that the sequentiality problem for polynomially-ambiguous min-plus automata is decidable. In contrast, the decidability of the general problem is still open. As a second example, in [26], the containment problem for register automata is shown to be considerably easier when one focuses on unambiguous such automata. For these reasons, recently, restrictions on the ambiguity were studied in the context of probabilistic automata [17]. There, the authors show that the emptiness problem is decidable for finitely ambiguous probabilistic automata, in contrast to the general class.

We define the ambiguity of a PA to be that of the underlying non-deterministic

[1] In our formulation of the emptiness problem we fix the probability threshold to be 1/2. Many authors allow the threshold to be any rational number between 0 and 1 and consider it as an additional input to the problem. However it is straightforward to reduce the general case to the fixed-threshold formulation that we adopt here.

finite automaton: thus a PA is *finitely ambiguous* if there exists a finite bound on the number of accepting runs of any word and *polynomially ambiguous* if there exists a polynomial function $f : \mathbb{N} \to \mathbb{N}$ such that every word of length n has at most $f(n)$ accepting runs. The classes of finitely ambiguous and polynomially ambiguous automata respectively admit characterisations in terms of the structure of their transition graphs and, based on these characterisations, it is decidable in polynomial time whether a PA is finitely ambiguous and whether it is polynomially ambiguous [27].

Our main results are as follows:

Gap emptiness problem. We show that the gap emptiness problem, which is undecidable in general, becomes decidable when we restrict to the class of polynomially ambiguous automata.

Theorem 1. *The gap emptiness problem is decidable for the class of polynomially ambiguous probabilistic automata.*

To prove Theorem 1 we approximate polynomially ambiguous automata by finitely ambiguous automata and rely on the decidability of emptiness in the finitely ambiguous case [17].

Emptiness problem. While polynomial ambiguity suffices for decidability of the gap emptiness problem, we show that the emptiness problem itself is undecidable even for linearly ambiguous automata.

Theorem 2. *The emptiness and containment problems are undecidable for the class of linearly ambiguous probabilistic automata.*

Theorem 2 strengthens a previous result that emptiness and containment are undecidable for quadratically ambiguous PA [17]. Recall that emptiness is decidable for finitely ambiguous PA [17] and hence this result is optimal.

Containment problem. Theorem 2 leads us to consider the decidability of containment between finitely ambiguous automata. Here we have the following result:

Theorem 3. *If Schanuel's conjecture holds then the containment problem is decidable if at least one of the input automata is unambiguous while the other is finitely ambiguous.*

We prove Theorem 3 by reduction to an arithmetical decision problem involving integer exponentiation, which we solve by a process of relaxation and rounding. The dependence on Schanuel's conjecture in this theorem is due to our use of the result of Macintyre and Wilkie [28] that the theory of real exponentiation is decidable subject to Schanuel's conjecture. The decidability of containment between two finitely ambiguous automata remains open and appears to involve difficult number-theoretic considerations.

Organisation of the paper. In Section 2, we give the formal definition of PA, the notion of ambiguity and classical results that will be useful in the paper. In Section 3, we recall the problems under consideration. In Section 4 we show that the gap emptiness problem for polynomially ambiguous PA is decidable (Theorem 1). Then, in Sections 6 and 7, we prove Theorem 3. Before that, in

Section 5, we explain how to translate the containment problem into a problem about the existence of integral exponents for certain exponential inequalities. Using this formalism, we prove that the containment problem for $\mathcal{A}$ and $\mathcal{B}$, as stated above, is decidable if $\mathcal{A}$ is finitely ambiguous and $\mathcal{B}$ is unambiguous. In Section 7, we tackle the more challenging direction and prove that the containment problem is also decidable if $\mathcal{A}$ is unambiguous and $\mathcal{B}$ is finitely ambiguous. Finally, in Section 8, we prove that the emptiness and the containment problems are undecidable provided that one of the automata is allowed to be linearly ambiguous (Theorem 2).

2. Preliminaries

In this section, we define probabilistic automata and recall some classical results.

Notation. We use boldface lower-case letters, e.g., $\boldsymbol{a}, \boldsymbol{b}, \ldots$, to denote vectors and upper-case letters, e.g., $M, N, \ldots$, for matrices. For a vector $\boldsymbol{a}$, we write a_i for its i-th component, and $\boldsymbol{a}^\top$ for its transpose.

2.1. Probabilistic automata and ambiguity

For a finite set S, we say that a function $f : S \to \mathbb{Q}_{\geq 0}$ is a *distribution over* S if $\sum_{s \in S} f(s) \leq 1$. We write $\mathcal{D}(S)$ for the set of all distributions over S. We also say that a vector $\boldsymbol{d} = (d_1, d_2, \ldots, d_n) \in \mathbb{Q}^n_{\geq 0}$ of non-negative rationals is a distribution if $\sum_{i=1}^n d_i \leq 1$.

A *probabilistic automaton (PA)* $\mathcal{A}$ is a tuple $(\Sigma, Q, \delta, \iota, F)$, where:

- Σ is the finite alphabet,
- Q is the finite set of states,
- $\delta : Q \times \Sigma \to \mathcal{D}(Q)$ is the (probabilistic) transition function,
- $\iota \in \mathcal{D}(Q)$ is the initial distribution, and
- $F \subseteq Q$ is the set of final states.

We write $\delta(q, a, p)$ instead of $\delta(q, a)(p)$ for the *probability of moving from q to p reading a*. Consider the word $w = a_1 \ldots a_n \in \Sigma^*$. A *run ρ of $\mathcal{A}$ over w* is a sequence of transitions $(q_0, a_1, q_1), (q_1, a_2, q_2), \ldots, (q_{n-1}, a_n, q_n)$ where $\delta(q_{i-1}, a_i, q_i) > 0$ for all $1 \leq i \leq n$. It is an *accepting run* if $\iota(q_0) > 0$ and $q_n \in F$. The *probability of the run* ρ is $\Pr_{\mathcal{A}}(\rho) \stackrel{\text{def}}{=} \iota(q_0) \cdot \prod_{i=1}^n \delta(q_{i-1}, a_i, q_i)$.

The automaton $\mathcal{A}$ realizes a function $[\![\mathcal{A}]\!]$ mapping words over the alphabet Σ to values in $[0, 1]$. Formally, for all $w \in \Sigma^*$, we set: $[\![\mathcal{A}]\!](w) \stackrel{\text{def}}{=} \sum_{\rho \in \mathrm{Acc}_{\mathcal{A}}(w)} \Pr_{\mathcal{A}}(\rho)$ where $\mathrm{Acc}_{\mathcal{A}}(w)$ is the set of all accepting runs of $\mathcal{A}$ over w.

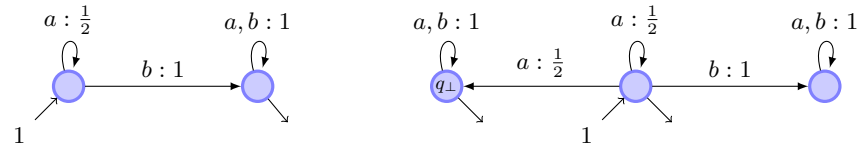

Figure 1: Two PA over the alphabet $\Sigma = \{a, b\}$ are depicted. On the left hand side, automaton $\mathcal{A}$ induces the function $a^n b\Sigma^* \mapsto \frac{1}{2^n}$ and $a^* \mapsto 0$. On the right hand side, the automaton $\overline{\mathcal{A}}$ induces the function $a^n b\Sigma^* \mapsto 1 - \frac{1}{2^n}$ and $a^* \mapsto 1$. Observe that $\mathcal{A}$ is unambiguous and $\overline{\mathcal{A}}$ is linearly ambiguous.

Ambiguity. The notion of ambiguity depends only on the structure of the underlying automaton (i.e., whether a probability is null or not, but not on its actual value). An automaton $\mathcal{A}$ is said to be *unambiguous* (resp. *k-ambiguous*) if for all words w, there is at most one accepting run (resp. k accepting runs) over w in $\mathcal{A}$. If an automaton is k-ambiguous for some k, then it is said to be *finitely ambiguous*. If there exists a polynomial P, such that for every word w, the number of accepting runs of $\mathcal{A}$ on w is bounded by $P(|w|)$ (where $|w|$ is the length of w), then $\mathcal{A}$ is said to be *polynomially ambiguous*, and *linearly ambiguous* whenever the degree of P is at most 1.

It is well-known that if an automaton is not finitely ambiguous then it is at least linearly ambiguous (see, for example, the criterion in [29, Section 3]). The same paper shows that if an automaton is finitely ambiguous then it is k-ambiguous for k bounded exponentially in the number of states of that automaton.

We give two examples of PA and discuss their ambiguity in Figure 1. As usual, they are depicted as graphs. The initial distribution is denoted by ingoing arrows associated with their probability (when there is no such arrow, the initial probability is 0) and the final states are denoted by outgoing arrows.

2.2. Classical results

Weighted-sum automaton. For PA $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$ over the same alphabet, and for a discrete distribution $\boldsymbol{d} = (d_1, d_2, \ldots, d_n)$, the *weighted-sum automaton* (of $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$ with weights $\boldsymbol{d}$) is defined to be the disjoint union of the n automata with the initial distribution $\iota(q) \stackrel{\text{def}}{=} d_i \cdot \iota_i(q)$ if q is a state of $\mathcal{A}_i$, where ι_i is the initial distribution of $\mathcal{A}_i$. Note that if $\mathcal{B}$ is the weighted sum of $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n$ with weights $\boldsymbol{d}$ then it is also a probabilistic automaton and $[\![\mathcal{B}]\!] = \sum_{i=1}^n d_i \cdot [\![\mathcal{A}_i]\!]$.

Complement automaton. For a PA $\mathcal{A}$, we define its *complement automaton* $\overline{\mathcal{A}}$ in the following way. First, define the PA $\mathcal{A}'$ by modifying $\mathcal{A}$ as follows:

- add a new sink state $q_\perp$;
- obtain the transition function δ' from δ by adding transitions:
  - $\delta'(q_\perp, a, q_\perp) = 1$ for all $a \in \Sigma$,
  - $\delta'(q, a, q_\perp) = 1 - \sum_{r \in Q} \delta(q, a, r)$ for all $(q, a) \in Q \times \Sigma$;
- obtain the initial distribution ι' from ι by adding $\iota'(q_\perp) = 1 - \sum_{q \in Q} \iota(q)$.

Observe that $[\![\mathcal{A}']\!] = [\![\mathcal{A}]\!]$, that $\sum_{r\in Q} \delta'(q,a,r) = 1$ for all $(q,a) \in Q \times \Sigma$, and that $\sum_{q\in Q} \iota'(q) = 1$. We obtain $\overline{\mathcal{A}}$ from $\mathcal{A}'$ by swapping its final and non-final states. As expected, it is the case that $[\![\overline{\mathcal{A}}]\!] = 1 - [\![\mathcal{A}]\!]$.

Remark 1 (Preserving ambiguity). The ambiguity of a weighted-sum automaton is at most the sum of the ambiguities of the individual automata, and the ambiguity of a complement automaton may be larger than the ambiguity of the original one (see Figure 1).

3. Decision problems

In this work, we are interested in comparing the functions computed by PA. We write $[\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]$ if "$\mathcal{A}$ is contained in $\mathcal{B}$", that is if $[\![\mathcal{A}]\!](w) \leq [\![\mathcal{B}]\!](w)$ for all $w \in \Sigma^*$; and we write $[\![\mathcal{A}]\!] \leq \frac{1}{2}$ if $[\![\mathcal{A}]\!](w) \leq \frac{1}{2}$ for all $w \in \Sigma^*$. We are interested in the following decision problems for PA.

- *Containment problem:* Given probabilistic automata $\mathcal{A}$ and $\mathcal{B}$, does $[\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]$ hold?
- *Emptiness problem:* Given a probabilistic automaton $\mathcal{A}$, does $[\![\mathcal{A}]\!] \leq \frac{1}{2}$ hold?
- *Gap emptiness problem:*
  - **Input**: $\varepsilon \in \mathbb{Q} \cap (0,1)$ and a probabilistic automaton $\mathcal{A}$ such that either there is a word w satisfying $[\![\mathcal{A}]\!](w) > \frac{1}{2} + \varepsilon$ or $[\![\mathcal{A}]\!](w) \leq \frac{1}{2}$ for all words w.
  - **Output**: does $[\![\mathcal{A}]\!](w) \leq \frac{1}{2}$ hold for all words w?

We will argue that the containment and emptiness problems are both undecidable when considered for the class of linearly ambiguous automata (Section 8). The emptiness problem is known to be decidable for the class of finitely ambiguous automata [17]. We tackle here the more difficult containment problem (Sections 6 and 7).

As for the gap emptiness problem, recall that for general PA, it is known to be undecidable [22]. We know that it is decidable for finitely-ambiguous PA (because the emptiness problem is decidable). Section 4 is devoted to clarifying the decidability frontier for polynomially-ambiguous PA.

4. Decidability of gap emptiness for polynomially ambiguous automata

This section is devoted to proving that the gap emptiness problem is decidable for the class of polynomially ambiguous PA.

For the rest of the section, fix a rational $\varepsilon \in (0,1)$ and a PA $\mathcal{A} = (\Sigma, Q, \delta, \iota, F)$ that is polynomially ambiguous. We also assume, without loss of generality, that $\mathcal{A}$ is trimmed (i.e., all states are reachable from some initial state and can reach some final state).

The key ingredient of this section is to show that we can compute a PA $\mathcal{A}'$ such that:

- $\mathcal{A}'$ is finitely ambiguous,

- for all words w, $[\![\mathcal{A}']\!](w) \leq [\![\mathcal{A}]\!](w) \leq [\![\mathcal{A}']\!](w) + \varepsilon$.

Using such a construction we can easily prove Theorem 1 reducing the question to the emptiness problem of $\mathcal{A}'$ (which is decidable since $\mathcal{A}'$ is finitely ambiguous). Indeed, suppose that for all w we have $\mathcal{A}'(w) \leq \frac{1}{2}$. Then for all w, $\mathcal{A}(w) \leq \frac{1}{2} + \varepsilon$. Hence there does not exist a word w such that $\mathcal{A}(w) > \frac{1}{2} + \varepsilon$. Conversely, if there exists w such that $\mathcal{A}'(w) > \frac{1}{2}$ then $\mathcal{A}(w) > \frac{1}{2}$. But then it is not the case that for all w we have $\mathcal{A}(w) \leq \frac{1}{2}$.

Construction of $\mathcal{A}'$. Let N be a positive integer (N will be fixed later in the proof, depending only on ε and $\mathcal{A}$). Let $\mathcal{A}'$ be the same as $\mathcal{A}$ except that on every run, we are only allowed to make the first N non-deterministic choices (including the choice of the initial state). In other words, we can only take at most N times a transition (p, a, q) of non-zero probability such that there exists another transition (p, a, q') of non-zero probability. After seeing the $(N+1)$-th non-deterministic choice, the automaton rejects the run. This can be achieved easily by making N copies of $\mathcal{A}$. Clearly, $\mathcal{A}'$ is finitely ambiguous: for every word, there are at most $|Q|^N$ accepting runs. Moreover, since the runs of $\mathcal{A}'$ can be embedded in the runs of $\mathcal{A}$, $[\![\mathcal{A}']\!](w) \leq [\![\mathcal{A}]\!](w)$ for all words. It remains to prove that $[\![\mathcal{A}]\!](w) \leq [\![\mathcal{A}']\!](w) + \varepsilon$ for all words w.

Proof that $[\![\mathcal{A}]\!](w) \leq [\![\mathcal{A}']\!](w) + \varepsilon$ for all words w. The combination of the two following lemmas will prove $[\![\mathcal{A}]\!](w) \leq [\![\mathcal{A}']\!](w) + \varepsilon$ for all words w.

Lemma 4. *Any run in $\mathcal{A}$ using exactly m non-deterministic choices has probability at most α^m, where α is the maximal transition probability not equal to* 1 *in $\mathcal{A}$.*

Lemma 5. *There exists a polynomial $P_{\mathcal{A}}$ such that for all words w, there are at most $P_{\mathcal{A}}(m)$ accepting runs of $\mathcal{A}$ over w using exactly m non-deterministic choices.*

Let us first prove that these two lemmas lead to the result. By definition of $\mathcal{A}'$, Lemma 4, and Lemma 5, we have:

$$[\![\mathcal{A}]\!](w) - [\![\mathcal{A}']\!](w) \leq \sum_{m=N+1}^{\infty} \alpha^m P_{\mathcal{A}}(m)$$

for some polynomial function $P_{\mathcal{A}}$, where α is the maximal transition probability not equal to 1 in $\mathcal{A}$.

Since $\alpha < 1$, it is easily verified that the series $\sum_{m=0}^{\infty} \alpha^m P_{\mathcal{A}}(m)$ converges (e.g. by the d'Alembert's ratio test). This is equivalent to

$$\lim_{N \to \infty} \sum_{m=N+1}^{\infty} \alpha^m P_{\mathcal{A}}(m) = 0.$$

Hence it suffices to take N such that $\sum_{m=N+1}^{\infty} \alpha^m P_{\mathcal{A}}(m) \leq \varepsilon$. Such an N is computable from ε and $\mathcal{A}$. Hence, it suffices to prove Lemma 4 and Lemma 5.

Lemma 4 is immediate: by definition, on a run using m non-deterministic choices, there are at least m transitions with probability smaller than 1, and thus with probability at most α (while the other transitions have probability at most 1).

Let us now turn to the proof of Lemma 5.

Lemma 6. *For all words w, there are at most $2^{|Q|}((m+1)|Q|^2)^{|Q|^3}$ runs of $\mathcal{A}$ over w using exactly m non-deterministic choices.*

The rest of this section is devoted to proving this lemma, which will conclude the proof.

We define the strongly connected components (SCC) of $\mathcal{A}$ as the SCCs of the underlying graphs when ignoring the labels of the transitions (only considering transitions with positive probability).

Lemma 7. *For all states p, q in a same SCC of $\mathcal{A}$ and for all words w, there is at most one run from p to q on w.*

Note that by definition of SCC, any such run remains in the SCC.

Proof. We use a criterion from [29] that characterizes ambiguity of automata. The EDA criterion states that an automaton is not polynomially ambiguous if and only if there is a word $w \in \Sigma^*$ and a state $s \in Q$ such that there are at least two different runs from s to s on w.

Suppose now that there is a word w such that there are two distinct runs from p to q as defined in the lemma. Since p and q belong to the same SCC, there is a word w' and a run from q to p on w'. Thus there are two distinct runs from p to p on ww'. Using the EDA criterion, $\mathcal{A}$ cannot be polynomially ambiguous and we get a contradiction. □

As a direct consequence, we have:

Corollary 8. *For all words w, there are at most $|Q|^2$ runs on w that start and end in any given SCC.*

The next lemma is the last ingredient needed to prove Lemma 6.

Lemma 9. *For all words w, the sum of all runs of $\mathcal{A}$ over all prefixes of w starting and ending in a given SCC and using at most m non-deterministic choices is bounded by $((m+1)|Q|^2)^{|Q|^2}$.*

Proof. The lemma can be rephrased in terms of counting the number of leaves in a tree. More precisely: runs correspond to branches (which are artificially rooted together); internal nodes to states in runs in the given SCC; and leaves correspond to states outside of the given SCC (so when a run leaves the given SCC) or states after reading the whole w. Non-deterministic choices correspond to nodes having several children.

We consider finite trees. The width of the tree is the maximal number of non-leaf nodes at a same depth (the root has depth 0) and we say that a node is a *split* if it has more than one child. Recall that by definition of $\mathcal{A}$ the outdegree of every node is bounded by $|Q|$. By Corollary 8, the lemma boils down to proving that the number of leaves in a tree which: (1) has width at most $|Q|^2$, (2) has outdegree at most $|Q|$, (3) has at most m split nodes on each branch, is at most $((m+1)|Q|^2)^{|Q|^2}$. Notice that the bound on the outdegree does not follow from the width bound because children can be leaves. Figure 2 visualises the runs on a SCC as trees on some example automaton.

We will show a more general claim, which also proves Lemma 9.

Claim 10. *For all non-negative integers k and m, the number of leaves in a tree which has*

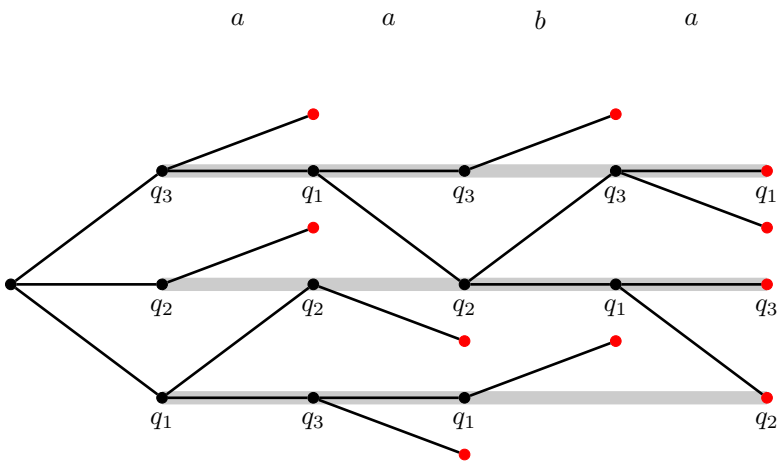

Figure 2: In this example the set of states is $Q = Q_1 \cup P$, where $Q_1 = \{q_1, q_2, q_3\}$ are states in the given SCC and P are the remaining states. We consider runs over the word $w = aaba$. The transitions are depicted only on the picture. The width is 3 and this bound is highlighted with grey shadows. The inner nodes are coloured black and the leaves are coloured red. The unlabelled leaves have states from P.

(1) width and outdegree at most k and

(2) at most m split nodes on each branch

is at most $((m+1)k)^k$.

The proof of the claim is by induction on m and k. The base case $m = 0$ is immediate, since a tree with no split is a path and has one leaf. Moreover the case $k = 0$ corresponds to a tree consisting of only one node and the induction hypothesis is also satisfied. Now, consider a tree with positive parameters m and k. We will focus on the first split node, that is the node where the single branch from the root splits for the first time. Let us call it v. By assumption, v has at most k children. Furthermore, it is not hard to see that there is at most one subtree of v with width k. Hence, the number of leaves in the tree is bounded by the sum of the leaves of

- one subtree of v with width k and at most $m - 1$ split notes, and
- k subtrees of v with width at most $k - 1$ and at most $m - 1$ split nodes.

Overall, by the induction hypothesis, the number of leaves in the tree is thus bounded by $(mk)^k + k(m(k-1))^{k-1}$ which is smaller than $((m+1)k)^k$. □

Proof of Lemma 6. We prove the lemma by induction on the number of states in $\mathcal{A}$. The case for one state is immediate. Let w be a word and consider now a source-SCC in $\mathcal{A}$ that is to say an SCC such that there is no transition from any other SCC to this one. Such a source-SCC always exists. By Lemma 9, there are at most $((m+1)|Q|^2)^{|Q|^2}$ runs on a prefix of w in the source-SCC, using at most m non-deterministic choices. If any run is continued after leaving this source-SCC we can bound the number of runs that arise from it by the induction hypothesis because the number of states has decreased. Since every run either

starts in this source-SCC (and if it leaves it, it does not go back to it) or never reaches it, by induction hypothesis, there are at most

$$((m+1)|Q|^2)^{|Q|^2}2^{|Q|-1}((m+1)(|Q|-1)^2)^{(|Q|-1)^3} + 2^{|Q|-1}((m+1)(|Q|-1)^2)^{(|Q|-1)^3}$$

runs on w in $\mathcal{A}$ using at most m non-deterministic choices. This concludes the proof since:

$$\begin{aligned}
&((m+1)|Q|^2)^{|Q|^2}2^{|Q|-1}((m+1)(|Q|-1)^2)^{(|Q|-1)^3} \\
&\qquad + 2^{|Q|-1}((m+1)(|Q|-1)^2)^{(|Q|-1)^3} \\
&\leq \quad 2^{|Q|-1}((m+1)|Q|^2)^{|Q|^2(1+|Q|-1)} + 2^{|Q|-1}((m+1)|Q|^2)^{|Q|^3} \\
&= \quad 2^{|Q|}((m+1)|Q|^2)^{|Q|^3}
\end{aligned}$$

□

5. From containment to exponential inequalities

Our aim is to decide whether $[\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]$. We first give a translation of the problem into a problem about the existence of integral exponents for certain exponential inequalities.

Notation. In the rest of the paper, we write $\exp(x)$ to denote the exponential function $x \mapsto e^x$, and $\log(y)$ for the natural logarithm function $y \mapsto \log_e(y)$. For a real number x and a positive real number y, we write y^x for $\exp(x \log(y))$.

5.1. The translation into exponential inequalities

We are going to translate the negation of the containment problem: Given two finitely ambiguous PA $\mathcal{A}$ and $\mathcal{B}$, does there exist a word w, such that $[\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w)$? Consider two positive integers k and n, and vectors $\boldsymbol{p} \in \mathbb{Q}_{>0}^k$ and $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k \in \mathbb{Q}_{>0}^n$. We denote by $S(\boldsymbol{p}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_k) : \mathbb{N}^n \to \mathbb{R}$ the function associating a vector $\boldsymbol{x} \in \mathbb{N}^n$ to $\sum_{i=1}^k p_i q_{i,1}^{x_1} \cdots q_{i,n}^{x_n}$, where $q_{i,j}$ is the j-th component of vector $\boldsymbol{q}_i$.

Proposition 11. *Given a k-ambiguous automaton $\mathcal{A}$ and an ℓ-ambiguous automaton $\mathcal{B}$, one can compute a positive integer n and a finite set Δ of tuples $(\boldsymbol{p}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_{k'}, \boldsymbol{r}, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{\ell'})$ of vectors $\boldsymbol{p} \in \mathbb{Q}_{>0}^{k'}, \boldsymbol{r} \in \mathbb{Q}_{>0}^{\ell'}$, for some $k' \leq k$ and $\ell' \leq \ell$; and $\boldsymbol{q}_i \in \mathbb{Q}_{>0}^n, \boldsymbol{s}_j \in \mathbb{Q}_{>0}^n$, for all i and j; such that the following two conditions are equivalent:*

- *there exists $w \in \Sigma^*$ such that $[\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w)$,*
- *there exist $(\boldsymbol{p}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_{k'}, \boldsymbol{r}, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{\ell'}) \in \Delta$ and $\boldsymbol{x} \in \mathbb{N}^n$ such that*

$$S(\boldsymbol{p}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_{k'})(\boldsymbol{x}) > S(\boldsymbol{r}, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{\ell'})(\boldsymbol{x}).$$

It thus follows that to prove Theorem 3, it suffices to show decidability of the second item of Proposition 11 for a given element of Δ in the cases where either k or ℓ are equal to 1. The proof of Proposition 11 is technical and is postponed to Section 5.2.

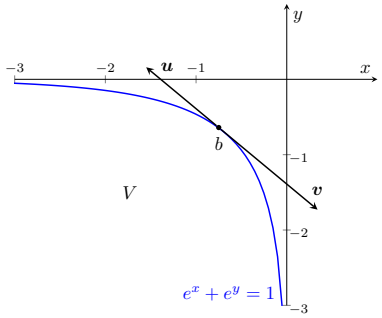

Figure 3: The set V is bounded by the plot $e^x + e^y = 1$ and the point b is on that plot.

Example 1. Consider the following instance of the problem, where $k = n = 2$, $\ell = 1$, and p is a fixed rational number $0 \leq p \leq 1$: Do there exist $x, y \in \mathbb{N}$ such that $p \cdot \left(\frac{1}{12}\right)^x \cdot \left(\frac{1}{2}\right)^y + (1-p) \cdot \left(\frac{1}{3}\right)^x \cdot \left(\frac{1}{18}\right)^y < \left(\frac{1}{6}\right)^x \cdot \left(\frac{1}{6}\right)^y$. One can see that is an instance of the equivalent problem in Proposition 11, where: $\boldsymbol{p} = (p, 1-p)$, $\boldsymbol{q}_1 = (\frac{1}{12}, \frac{1}{2})$, $\boldsymbol{q}_2 = (\frac{1}{3}, \frac{1}{18})$, $\boldsymbol{r} = 1$, and $\boldsymbol{s}_1 = (\frac{1}{6}, \frac{1}{6})$. This can be rewritten as

$$p \cdot \left(\frac{1}{2}\right)^x \cdot 3^y + (1-p) \cdot 2^x \cdot \left(\frac{1}{3}\right)^y < 1$$

or equivalently, using the exponential function, as follows

$$\exp(\log(p) - x\log(2) + y\log(3)) + \exp(\log(1-p) + x\log(2) - y\log(3)) < 1.$$

Consider the set $V = \{(x, y) \in \mathbb{R}^2 \mid e^x + e^y < 1\}$ and denote by b the point $(\log(p), \log(1-p))$. Let $\boldsymbol{u} = (-\log(2), \log(2))$ and $\boldsymbol{v} = (\log(3), -\log(3))$ be two vectors. See Figure 3 for a geometric representation. The question is now: do there exist $x, y \in \mathbb{N}$ such that $b + x\boldsymbol{u} + y\boldsymbol{v} \in V$. We will show that the answer is yes if and only if $p \neq \frac{1}{2}$.

Let $C = \{(x, -x) \mid x \in \mathbb{R}\}$. For $p = \frac{1}{2}$, the affine line $C + p$ is tangent to the blue plot and so, whatever the values of x and y, $b + x\boldsymbol{u} + y\boldsymbol{v}$ cannot be in V. For $p \neq \frac{1}{2}$, there is a value δ such that the whole interval strictly between b and $b + (\delta, -\delta)$ is in V. Since $\log(2)$ and $\log(3)$ are rationally independent, the set $D = \{x\boldsymbol{u} + y\boldsymbol{v} \mid x, y \in \mathbb{N}\}$ is a dense subset of C, so in particular, there is a point of $D + b$ in the interval between b and $b + (\delta, -\delta)$ and thus there exist $x, y \in \mathbb{N}$ such that $b + x\boldsymbol{u} + y\boldsymbol{v} \in V$.

5.2. Proof of Proposition 11

We write that the first state of a run is the first state of the first transition, and the last state of a run is the last state of the last transition. A run is *short* if every state appearing in the sequence of transitions composing the run, appears at most twice. A *cycle* is a run in which the first and the last states coincide. A *short cycle* is a cycle which is a short run.

For a state q, we say that a cycle is a *q-cycle* if the first (and hence also the last) state is q. For a run $\rho = \rho' \cdot \rho''$, such that the last state of ρ' (and hence also the first state of ρ'') is q, and for a q-cycle ω, the result of *injecting ω (after ρ') into ρ* is the run $\rho' \cdot \omega \cdot \rho''$.

For a run ρ, we write $Q(\rho)$ for the set of states that occur in it. For a set of states P, we write Periods(P) for the set of short cycles in which only

states in P occur. A *short cycle decomposition* is a pair (γ, σ), where γ is a run of length less than $|Q(\gamma)|^2$ and $\sigma : \text{Periods}(Q(\gamma)) \to \mathbb{N}$. We say that a short cycle decomposition (γ, σ) is a *short cycle decomposition of a run* ρ if the run ρ can be obtained from γ by injecting $\sigma(\omega)$ cycles ω, for every short cycle $\omega \in \text{Periods}(Q(\gamma))$, in some order.

Proposition 12. *Every run has a short cycle decomposition.*

Proof. The above result is classic, cf. e.g. [30, proof of Lemma 4.5]. It follows by repeatedly removing from a run ρ, as long as its length is at least $|Q(\rho)|^2$, some short cycle whose removal does not decrease the set of states $Q(\rho)$. To see that such a short cycle must exist, observe that if the length of ρ is at least $|Q(\rho)|^2$, then it contains $|Q(\rho)|$ non-overlapping short cycles $\omega_1, \ldots, \omega_{|Q(\rho)|}$; and if W_i is the set of all states that occur strictly inside ω_i but nowhere else in ρ, then the sets $W_1, \ldots, W_{|Q(\rho)|}$ are mutually disjoint and their union has size less than $|Q(\rho)|$, so some W_i must be empty. □

If (γ, σ) is a short cycle decomposition, then we refer to γ as its *spine* and to σ as its *short cycle count*. Observe that the number of distinct spines is finite; more specifically, it is at most exponential in the size of the automaton, as is the set of short cycles $\text{Periods}(Q(\gamma))$ for every spine γ.

We say that a short cycle decomposition (γ, σ) is *accepting* if the run γ is. By Proposition **??**, every accepting run has an accepting short cycle decomposition. Moreover, for every accepting spine γ, and for every function $\sigma : \text{Periods}(Q(\gamma)) \to \mathbb{N}$, there is at least one accepting run ρ, such that (γ, σ) is its short cycle decomposition.

Proposition 13. *There is an algorithm that given a finitely ambiguous probabilistic automaton $\mathcal{A}$ and a nonnegative integer i, outputs a finite automaton that accepts the language of words on which $\mathcal{A}$ has exactly i accepting runs.*

Proof. We can assume that $\mathcal{A}$ is trimmed (i.e., all states are reachable from some initial state and can reach some final state). It is known that the number of all active runs in a trimmed finitely ambiguous automaton is bounded exponentially in the number of states in $\mathcal{A}$ [29]. We can therefore add an extra component to $\mathcal{A}$ that, using the powerset construction, keeps track of all active runs in the automaton. Using this component, the automaton $\mathcal{A}$ can extract the number of all accepting runs. □

We are ready to prove Proposition 11.

Proof of Proposition 11. First, use Proposition 13 to compute finite automata $\mathcal{A}_{k'}$, $0 \leq k' \leq k$, and $\mathcal{B}_{\ell'}$, $0 \leq \ell' \leq \ell$, that accept the languages of words on which $\mathcal{A}$ has exactly k' accepting runs and $\mathcal{B}$ has exactly ℓ' accepting runs, respectively.

For all k', $0 \leq k' \leq k$, and for all ℓ', $0 \leq \ell' \leq \ell$, we perform the following. Consider the product automaton of $\mathcal{A}_{k'}$, $\mathcal{B}_{\ell'}$, k' copies of $\mathcal{A}$, and ℓ' copies of $\mathcal{B}$. Moreover, equip the product automaton with another component, a finite automaton that maintains (in its state space) the partition of the k' components corresponding to copies of $\mathcal{A}$, and of the partition of the ℓ' components corresponding to copies of $\mathcal{B}$, that reflects which of the k' runs of $\mathcal{A}$, and which of the ℓ' runs of $\mathcal{B}$, respectively, have been identical so far. Consider as final

the states of this additional component in which all sets in both partitions are singletons. The purpose of the last component is to be able to only consider runs of the product automaton in which the k' components corresponding to copies of $\mathcal{A}$, and the ℓ' components corresponding to copies of $\mathcal{B}$, have all distinct runs. Similarly, the purpose of the copies of $\mathcal{A}_{k'}$ and $\mathcal{B}_{\ell'}$ is to be able to only consider runs of the product automaton which record all the k' distinct accepting runs of $\mathcal{A}$ and all of the ℓ' distinct accepting runs of $\mathcal{B}$, respectively, on the underlying words. Let $\mathcal{C}_{k',\ell'}$ be the resulting finite automaton with $k' + \ell' + 3$ components.

The following proposition follows by the construction of automaton $\mathcal{C}_{k',\ell'}$.

Proposition 14. *There are exactly k' distinct runs of $\mathcal{A}$ on w and exactly ℓ' runs of $\mathcal{B}$ on w, if and only if there is an accepting run of $\mathcal{C}_{k',\ell'}$ on w.*

Consider the set of spines of $\mathcal{C}_{k',\ell'}$ in which all $k' + \ell' + 3$ components of the last state are accepting states; let m be the size of this set of *accepting spines*. For every such accepting spine γ, we define an instance of vectors $\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma, \boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma$. If we set $n = |\text{Periods}(Q(\gamma))|$ and (arbitrarily) enumerate all short cycles in $\text{Periods}(Q(\gamma))$ from 1 to n, then

- $\boldsymbol{p}^\gamma$ has k' components: for every i, such that $1 \leq i \leq k'$, we set p_i^γ to be the product of the probabilities of the transitions in the i-th copy of $\mathcal{A}$ in spine γ;
- $\boldsymbol{q}_i^\gamma$ has n components: for every $1 \leq j \leq n$, we set the j-th component of $\boldsymbol{q}_i^\gamma$ to be the product of the probabilities of the transitions in i-th copy of $\mathcal{A}$ in the j-th cycle in the set $\text{Periods}(Q(\gamma))$;
- $\boldsymbol{r}^\gamma$ has ℓ' components: for every i, such that $1 \leq i \leq \ell'$, we set r_i^γ to be the product of the probabilities of the transitions in the i-th copy of $\mathcal{B}$ in spine γ;
- $\boldsymbol{s}_i^\gamma$ has n components: for $1 \leq j \leq n$, we set the j-th component of $\boldsymbol{s}_i^\gamma$ to be the product of the probabilities of the transitions, in i-th copy of $\mathcal{B}$, in the j-th cycle in the set $\text{Periods}(Q(\gamma))$.

In the special case when $k' = 0$ or $l' = 0$ we put 0 everywhere (which can be understood as a 0-dimensional vector).

For an arithmetic expression E over n variables $\boldsymbol{x}$ indexed by elements of a set I, and for a function $\sigma : I \to \mathbb{N}$, we write $E[\sigma/\boldsymbol{x}]$ for the numerical value of the expression E in which every occurrence of variable x_i was replaced by $\sigma(i)$, for every $i \in I$. The following proposition follows again by the construction of automaton $\mathcal{C}_{k',\ell'}$, taking into account the following observations:

- the probability of a run of a probabilistic automaton can be determined from its short cycle decomposition (γ, σ), by taking the product of the following:
  - the product of the probabilities of the transitions in spine γ,
  - for every short cycle $\omega \in \text{Periods}(Q(\gamma))$, the $\sigma(\omega)$-th power of the product of the probabilities of the transitions in ω;
- in an accepting run of $\mathcal{C}_{k',\ell'}$ on a word $w \in \Sigma^*$, the k' components that correspond to k' copies of $\mathcal{A}$ all follow a distinct run of $\mathcal{A}$ on w, and hence

by Proposition 14, $[\![\mathcal{A}]\!](w)$ is the sum of the probabilities of the k' distinct runs followed by the k' copies of $\mathcal{A}$;

- in an accepting run of $\mathcal{C}_{k',\ell'}$ on a word $w \in \Sigma^*$, the ℓ' components that correspond to ℓ' copies of $\mathcal{B}$ all follow a distinct run of $\mathcal{B}$ on w, and hence by Proposition 14, $[\![\mathcal{B}]\!](w)$ is the sum of the probabilities of the ℓ' distinct runs followed by the ℓ' copies of $\mathcal{B}$.

Proposition 15. *If there is an accepting run ρ of $\mathcal{C}_{k',\ell'}$ on word $w \in \Sigma^*$, then for every short cycle decomposition (γ, σ) of ρ, we have*

- $[\![\mathcal{A}]\!](w) = S(\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma)[\sigma/\boldsymbol{x}]$ *and*
- $[\![\mathcal{B}]\!](w) = S(\boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma)[\sigma/\boldsymbol{x}]$.

Let Δ be the set of all tuples $(\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma, \boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma)$ given by all accepting spines, in particular $|\Delta|$ is finite. We now argue that the two conditions in the statement of Proposition 11 are indeed equivalent. Let $w \in \Sigma^*$ be a word such that $[\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w)$. Let the numbers of distinct accepting runs of $\mathcal{A}$ and $\mathcal{B}$, respectively, on w be k' and ℓ', respectively. Then, by Proposition 14, there is an accepting run ρ of $\mathcal{C}_{k',\ell'}$ on w. Let (γ, σ) be a short cycle decomposition of ρ; note that since γ is an accepting spine, we have $(\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma, \boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma) \in \Delta$. It then follows by Proposition 15, that σ is a non-negative integer solution of $S(\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma)[\sigma/\boldsymbol{x}] > S(\boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma)[\sigma/\boldsymbol{x}]$.

Conversely, suppose that there is a non-negative integer solution σ of the inequality

$$S(\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma)[\sigma/\boldsymbol{x}] > S(\boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma)[\sigma/\boldsymbol{x}],$$

for some quadruple

$$(\boldsymbol{p}^\gamma, \boldsymbol{q}_1^\gamma, \ldots, \boldsymbol{q}_{k'}^\gamma, \boldsymbol{r}^\gamma, \boldsymbol{s}_1^\gamma, \ldots, \boldsymbol{s}_{l'}^\gamma) \in \Delta.$$

This quadruple is in Δ because γ is an accepting spine of the automaton $\mathcal{C}_{k',\ell'}$ for some k' and ℓ', such that $0 \leq k' \leq k$ and $0 \leq \ell' \leq \ell$. Let ρ be an accepting run of $\mathcal{C}_{k',\ell'}$ that is obtained by injecting into γ, in some order, $\sigma(\omega)$ copies of the short cycle ω, for all $\omega \in \mathrm{Periods}(Q(\gamma))$; let $w \in \Sigma^*$ be the word underlying the run ρ. By Proposition 15, it follows that $[\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w)$. □

6. Decidability of containment of a finitely ambiguous automaton in an ambiguous automaton

We prove here the decidability of the containment problem when $\mathcal{A}$ is finitely ambiguous and $\mathcal{B}$ is unambiguous. The converse situation is tackled in Section 7.

Proposition 16. *Determining whether $[\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]$ is decidable when $\mathcal{A}$ is finitely ambiguous and $\mathcal{B}$ is unambiguous.*

Proof. Let $\mathcal{A}$ be k-ambiguous. Proposition 11 shows that it is sufficient to decide, given an integer n and positive rational numbers $p, q_j, r_i, s_{i,j}$ for $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$, whether there exists $x_1, \ldots, x_n \in \mathbb{N}$ such that

$$\sum_{i=1}^{k} p_i q_{i,1}^{x_1} \cdots q_{i,n}^{x_n} > r s_1^{x_1} \cdots s_n^{x_n}. \tag{1}$$

We consider two cases. First, assume that there exist i and j such that $q_{i,j} > s_j$. Then in that case, for a large enough $m \in \mathbb{N}$ condition (1) will be satisfied for $(x_1, \ldots, x_j, \ldots, x_n) = (0, \ldots, m, \ldots, 0)$. Otherwise, assume that $\max\{q_{i,j} \mid 1 \leq i \leq k\} \leq s_j$ for all $1 \leq j \leq n$. In this case, if there exists a valuation of the x_i satisfying (1) then $(x_1, \ldots, x_n) = (0, \ldots, 0)$ also satisfies it. It is then sufficient to test condition (1) for $x_1 = \cdots = x_n = 0$ to conclude. □

7. Decidability of containment of an unambiguous automaton in a finitely ambiguous automaton

In this section we will show the more challenging part of Theorem 3, i.e., that the containment problem is decidable for $\mathcal{A}$ unambiguous and $\mathcal{B}$ finitely ambiguous. Our proof is conditional on the first-order theory of the reals with the exponential function being decidable. In [28], the authors show that this is the case if a conjecture due to Schanuel and regarding transcendental number theory is true.

Theorem 17. *Determining whether $[\![\mathcal{A}]\!] \leq [\![\mathcal{B}]\!]$ is decidable when $\mathcal{A}$ is unambiguous and $\mathcal{B}$ is finitely ambiguous, assuming Schanuel's conjecture is true.*

To prove this, we will argue that the containment problem can be phrased as an *integer programming* problem with exponentiation. Some definitions follow before we present the proof of the claim.

7.1. Integer programming problem with exponentiation

Given two positive integers n and ℓ, we define $\mathcal{F}_{n,\ell}$ to be the set of all the functions $f : \mathbb{R}^n \to \mathbb{R}$ such that there exist $\boldsymbol{r} \in \mathbb{Q}^{\ell}_{>0}$ and $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\ell \in \mathbb{Q}^n_{>0}$ such that $f(\boldsymbol{x}) = \sum_{i=1}^{\ell} r_i s_{i,1}^{x_1} \ldots s_{i,n}^{x_n}$. Observe that this is just a lifting of the $S(\cdot)$ function, defined in the previous section, to real-valued parameters. Consider the following integer programming problem with exponentiation.

Problem 18 (IP+EXP)**.**

- ***Input:*** *Three positive integers n, ℓ and m, a function $f \in \mathcal{F}_{n,\ell}$, a matrix $M \in \mathbb{Z}^{m \times n}$, and a vector $\boldsymbol{c} \in \mathbb{Z}^m$.*
- ***Question:*** *Does there exist $\boldsymbol{x} \in \mathbb{Z}^n$ such that $f(\boldsymbol{x}) < 1$ and $M\boldsymbol{x} < \boldsymbol{c}$?*

In the sequel, we will show that the above problem is decidable.

Theorem 19. *The IP+EXP problem is decidable, assuming Schanuel's conjecture is true.*

Theorem 17 is a direct corollary of Theorem 19.

Proof of Theorem 17. Proposition 11 shows that, in order to prove Theorem 17, it is sufficient to decide, given an integer n and positive rational numbers $p, r_i, q_j, s_{i,j}$ for $i \in \{1, \ldots, \ell\}$, $j \in \{1, \ldots, n\}$, whether there exist $x_1, \ldots, x_n \in \mathbb{N}$ such that $pq_1^{x_1} \cdots q_n^{x_n} > \sum_{i=1}^{\ell} r_i s_{i,1}^{x_1} \cdots s_{i,n}^{x_n}$ or equivalently, whether there exist $x_1, \ldots, x_n \in \mathbb{N}$ such that:

$$\sum_{i=1}^{\ell} r_i p^{-1} (s_{i,1} q_1^{-1})^{x_1} \cdots (s_{i,n} q_n^{-1})^{x_n} < 1. \tag{2}$$

Define $f : \mathbb{R}^n \to \mathbb{R}$ such that $f(\boldsymbol{x}) = \sum_{i=1}^{\ell} r_i p^{-1}(s_{i,1}q_1^{-1})^{x_1} \cdots (s_{i,n}q_n^{-1})^{x_n}$. Then, inequality (2) becomes $f(\boldsymbol{x}) < 1$. We can now apply Theorem 19 with m set to be n; M, to be $-Id$, where Id is the identity matrix; and $\boldsymbol{c}$ to be the null vector. □

Since the IP+EXP problem is semi-decidable (indeed, we can enumerate the vectors $\boldsymbol{x}$ in $\mathbb{Z}^n$ to find one satisfying the conditions), it will suffice to give a semi-decision procedure to determine whether the inequalities $f(\boldsymbol{x}) < 1 \wedge M\boldsymbol{x} < \boldsymbol{c}$ have no integer solution. We give now such a procedure.

7.2. Semi-decision procedure for the complement of IP+EXP

Consider as input for the IP+EXP problem three positive integers n, ℓ, m, a function $f \in \mathcal{F}_{n,\ell}$, a matrix $M \in \mathbb{Z}^{m\times n}$, and a vector $\boldsymbol{c} \in \mathbb{Z}^m$. Denote by X the set of *real* solutions of the problem, i.e., the set of vectors $X = \{\boldsymbol{x} \in \mathbb{R}^n \mid f(\boldsymbol{x}) < 1 \wedge M\boldsymbol{x} < \boldsymbol{c}\}$.

Proc(n, ℓ, m, f, M, $\boldsymbol{c}$).

1. Search for a non-zero vector $\boldsymbol{d} \in \mathbb{Z}^n$ and $a, b \in \mathbb{Z}$ such that $\{\boldsymbol{d}^\top \boldsymbol{x} \mid \boldsymbol{x} \in X\} \subseteq [a, b]$. Set $i = a$.
2. If $i > b$, then stop and return `YES`. Otherwise, let Y_i be the set of vectors $\boldsymbol{x} \in \mathbb{Z}^n$ satisfying $d_1x_1 + \cdots + d_nx_n = i$. If Y_i is empty, then increment i and start again from step 2. Otherwise:
   (a) Compute $N \in \mathbb{Z}^{n\times(n-1)}$ and $\boldsymbol{h} \in \mathbb{Z}^n$ such that $Y_i = \{N\boldsymbol{y} + \boldsymbol{h} \mid \boldsymbol{y} \in \mathbb{Z}^{n-1}\}$.
   (b) If $n - 1 = 0$ and $f(\boldsymbol{h}) < 1 \wedge M\boldsymbol{h} < \boldsymbol{c}$ then return `NO`, otherwise increment i and start again from step 2.
   (c) If $n - 1 > 0$ then recursively call **Proc($n-1$, ℓ, m, f', M', $\boldsymbol{c}'$)**, where $f' \in \mathcal{F}_{n-1,\ell}$ is defined as $f'(\boldsymbol{y}) = f(N\boldsymbol{y} + \boldsymbol{h})$; $M' \in \mathbb{Z}^{m\times(n-1)}$, as $M' = MN$; and $\boldsymbol{c}' \in \mathbb{Z}^m$, as $\boldsymbol{c} - M\boldsymbol{h}$. If the procedure stops and returns `YES` then increment i and start again from step 2. If the procedure stops and returns `NO` then return `NO`.

Lemma 20. *The above semi-decision procedure stops and outputs `YES` if and only if there is no integer valuation of $\boldsymbol{x}$ that satisfies the constraints, i.e. $X \cap \mathbb{Z}^n$ is empty.*

We prove this lemma in Section 7.3. Before, let us shortly comment on both steps of the procedure.

Step 1 of the procedure. First, notice that the only step which might not terminate in a call to our procedure is step 1. Indeed, once $\boldsymbol{d}$, a, and b are fixed, there are only finitely many integers $i \in [a, b]$ that have to be considered in step 2.

Moreover, for each integer vector $\boldsymbol{d} \in \mathbb{Z}^n$ and $a, b \in \mathbb{Z}$, the inclusion $\{\boldsymbol{d}^\top \boldsymbol{x} \mid \boldsymbol{x} \in X\} \subseteq [a, b]$ that needs to be checked in step 1 can be formulated as a decision problem in the first-order logic over the structure $(\mathbb{R}, +, \times, \exp)$. Since this structure has a decidable first-order theory subject to Schanuel's conjecture [28], the inclusion can be decided for each fixed $\boldsymbol{d}$, a, and b.

Step 2 of the procedure. For fixed $\boldsymbol{d}$, a, and b, one can compute in a standard way the set of all integer solutions Y_i (see, e.g., [31]), as we now explain. By performing elementary column operations, find a $n \times n$ unimodular (i.e. with determinant equal to 1 or -1) integer matrix U such that

$$(d_1, \ldots, d_n)U = (g, 0, \ldots, 0),$$

where $g = \gcd(d_1, \ldots, d_n)$. Recall that Y_i is the set of integer solutions of $\boldsymbol{d}^\top \boldsymbol{x} = i$. We apply the change of variables $U\boldsymbol{y} = \boldsymbol{x}$ to it, where $\boldsymbol{y} = (y_1, \ldots, y_n)$, to obtain $\boldsymbol{d}^\top U\boldsymbol{y} = i$. Since $\boldsymbol{d}^\top U\boldsymbol{y} = (g, 0, \ldots, 0)\boldsymbol{y} = gy_1$, the transformed equation is $gy_1 = i$ and the matrix U gives a one-to-one correspondence between integer solutions $\boldsymbol{y}$ of the transformed equation and solutions $\boldsymbol{x} \in Y_i$. Now the transformed equation has a solution if and only if g divides i, in which case $y_1 = i/g$. Furthermore, in this case a general solution from Y_i can be written in the form $\boldsymbol{x} = N\boldsymbol{y}' + \boldsymbol{h}$ for N a $n \times (n-1)$ integer matrix and $\boldsymbol{h} \in \mathbb{Z}^n$ (both derived from U) and $\boldsymbol{y}' = (y_2, \ldots, y_n)$.

7.3. Proof of Lemma 20

The proof of Lemma 20 relies on the two following lemmas. The first one is the most technical contribution of the paper and is proved in Section 7.4. It ensures termination of step 1 in the procedure when there is no integer solution.

Lemma 21. *If the set X contains no integer point then there must exist a non-zero integer vector $\boldsymbol{d} \in \mathbb{Z}^n$ and $a, b \in \mathbb{Z}$ such that $\{\boldsymbol{d}^\top \boldsymbol{x} \mid \boldsymbol{x} \in X\} \subseteq [a, b]$.*

This second lemma guarantees that the recursive calls in step 2 guarantee the correct output.

Lemma 22. *Given a non-zero vector $\boldsymbol{d} \in \mathbb{Z}^n$ and an integer i, there exists $\boldsymbol{x} \in \mathbb{Z}^n$ such that $f(\boldsymbol{x}) < 1 \wedge M\boldsymbol{x} < \boldsymbol{c} \wedge \boldsymbol{d}^\top \boldsymbol{x} = i$ if and only if there exists $\boldsymbol{y} \in \mathbb{Z}^{n-1}$ such that $f'(\boldsymbol{y}) < 1 \wedge M'\boldsymbol{y} < \boldsymbol{c}'$ where f', M' and $\boldsymbol{c}'$ are as defined in the procedure.*

Proof. We want to prove that given a non-zero vector $\boldsymbol{d} \in \mathbb{Z}^n$ and an integer i, there exists $\boldsymbol{x} \in \mathbb{Z}^n$ such that $f(\boldsymbol{x}) < 1 \wedge M\boldsymbol{x} < \boldsymbol{c} \wedge \boldsymbol{d}^\top \boldsymbol{x} = i$ if and only if there exists $\boldsymbol{y} \in \mathbb{Z}^{n-1}$ such that $f'(\boldsymbol{y}) < 1 \wedge M'\boldsymbol{y} < \boldsymbol{c}'$ where f', M' and $\boldsymbol{c}'$ are as defined in the procedure. Recall that Y_i is the set of vectors $\boldsymbol{x}$ such that $\boldsymbol{d}^\top \boldsymbol{x} = i$ and that $Y_i = \{N\boldsymbol{y} + \boldsymbol{h} \mid \boldsymbol{y} \in \mathbb{Z}^{n-1}\}$ for some $N \in \mathbb{Z}^{n \times (n-1)}$ and $\boldsymbol{h} \in \mathbb{Z}^n$.

Let $\boldsymbol{x} \in \mathbb{Z}^n$ such that $f(\boldsymbol{x}) < 1 \wedge M\boldsymbol{x} < \boldsymbol{c} \wedge \boldsymbol{d}^\top \boldsymbol{x} = i$.

Then $\boldsymbol{x} \in Y_i$ and thus there is $\boldsymbol{y} \in \mathbb{Z}^{n-1}$ such that $\boldsymbol{x} = N\boldsymbol{y} + \boldsymbol{h}$. We have: $f'(\boldsymbol{y}) = f(N\boldsymbol{y} + \boldsymbol{h}) = f(\boldsymbol{x}) < 1$ and $M'\boldsymbol{y} = MN\boldsymbol{y} = M(\boldsymbol{x} - \boldsymbol{h}) = M\boldsymbol{x} - M\boldsymbol{h} < \boldsymbol{c} - M\boldsymbol{h} = \boldsymbol{c}'$.

Conversely, consider $\boldsymbol{y} \in \mathbb{Z}^{n-1} \cap Y_i$ such that $f'(\boldsymbol{y}) < 1 \wedge M'\boldsymbol{y} < \boldsymbol{c}'$. Let $\boldsymbol{x} = N\boldsymbol{y} + \boldsymbol{h}$. Then $\boldsymbol{x} \in Y_i$ and thus $\boldsymbol{d}^\top \boldsymbol{x} = i$. Moreover, $f(\boldsymbol{x}) = f(N\boldsymbol{y} + \boldsymbol{h}) = f'(\boldsymbol{y}) < 1$ and $M\boldsymbol{x} = M(N\boldsymbol{y} + \boldsymbol{h}) = MN\boldsymbol{y} + M\boldsymbol{h} = M'\boldsymbol{y} + M\boldsymbol{h} < \boldsymbol{c}' + M\boldsymbol{h} = \boldsymbol{c}$. □

We prove Lemma 20.

First direction: when the procedure returns `YES`. Suppose first that the semi-decision procedure stops and outputs `YES`. Then there exist a non-zero vector $\boldsymbol{d} \in \mathbb{Z}^n$ and $a, b \in \mathbb{Z}$ such that $\{\boldsymbol{d}^\top \boldsymbol{x} \mid \boldsymbol{x} \in X\} \subseteq [a, b]$ as in step 1, and for all integers $i \in [a, b]$, one of the following situations occurs:

1. Y_i is empty,
2. $n - 1 = 0$, $Y_i = \{\boldsymbol{h}\}$ as defined in step 2.*a* but $\boldsymbol{h}$ is not an integer solution of the problem,
3. $n - 1 > 0$ and the recursive call stops and outputs `YES`.

By definition of $\boldsymbol{d}$, in order to prove that there is no integer solution of the problem, we need to show that in all those cases, and for all $i \in [a, b]$, $Y_i \cap X = \emptyset$. It is clear for items 1 and 2 and we use Lemma 22 and an induction for item 3.

Second direction: when $X \cap \mathbb{Z}^n = \emptyset$. If there is no integer solution then by Lemma 21, there must exist a non-zero vector $\boldsymbol{d} \in \mathbb{Z}^n$ and $a, b \in \mathbb{Z}$ such that $\{\boldsymbol{d}^\top \boldsymbol{x} \mid \boldsymbol{x} \in X\} \subseteq [a, b]$ as in step 1. Moreover, for any of those choices, if for an integer $i \in [a, b]$, the set Y_i of vectors $\boldsymbol{x} \in \mathbb{Z}^n$ satisfying $d_1 x_1 + \cdots + d_n x_n = i$ is non-empty, then,

1. if $n = 1$, then $\boldsymbol{h}$ as defined in step 2.*a*, is not a solution of the problem (by hypothesis) and thus the procedure stops and returns `YES`,
2. if $n > 1$, we use Lemma 22 and, by induction, the recursive call must return `YES`.

7.4. Proof of Lemma 21

Fix three positive integers n, ℓ, m, a function $f \in \mathcal{F}_{n,\ell}$, a matrix $M \in \mathbb{Z}^{m \times n}$, and a vector $\boldsymbol{c} \in \mathbb{Z}^m$. Recall that we denote by X the set of vectors

$$X = \{\boldsymbol{x} \in \mathbb{R}^n \mid f(\boldsymbol{x}) < 1 \wedge M\boldsymbol{x} < \boldsymbol{c}\}.$$

We want to prove that if the set X contains no integer point then there must exist a non-zero integer vector $\boldsymbol{d} \in \mathbb{Z}^n$ and $a, b \in \mathbb{Z}$ such that $\{\boldsymbol{d}^\top \boldsymbol{x} \mid \boldsymbol{x} \in X\} \subseteq [a, b]$.

We will use the following corollary of Kronecker's theorem on simultaneous Diophantine approximation. It generalises the fact that any line in the plane with irrational slope passes arbitrarily close to integer points in the plane.

Proposition 23. *[32, Corollary 2.8]. Let $\boldsymbol{u}, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_s$ be vectors in $\mathbb{R}^n$. Suppose that for all $\boldsymbol{d} \in \mathbb{Z}^n$ we have $\boldsymbol{d}^\top \boldsymbol{u} = 0$ whenever $\boldsymbol{d}^\top \boldsymbol{u}_1 = \cdots = \boldsymbol{d}^\top \boldsymbol{u}_s = 0$. Then for all $\varepsilon > 0$ there exist real numbers $\lambda_1, \ldots, \lambda_s \geq 0$ and a vector $\boldsymbol{v} \in \mathbb{Z}^n$ such that $\|\boldsymbol{u} + \sum_{i=1}^s \lambda_i \boldsymbol{u}_i - \boldsymbol{v}\|_\infty \leq \varepsilon$.*

By definition, there exist vectors $\boldsymbol{r} \in \mathbb{Q}_{>0}^\ell$ and $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_\ell \in \mathbb{Q}_{>0}^n$ such that $f(\boldsymbol{x}) = \sum_{i=1}^\ell r_i s_{i,1}^{x_1} \ldots s_{i,n}^{x_n}$. Let $\boldsymbol{a} \in \mathbb{R}^\ell$ and $\boldsymbol{b}_i \in \mathbb{R}^n$ be defined by $a_i = \log(r_i)$ and $\boldsymbol{b}_i = (\log(s_{i,1}), \ldots, \log(s_{i,n}))$. We can then rewrite $f(\boldsymbol{x})$ as follows

$$f(\boldsymbol{x}) = \exp(\boldsymbol{b}_1^\top \boldsymbol{x} + a_1) + \cdots + \exp(\boldsymbol{b}_\ell^\top \boldsymbol{x} + a_\ell).$$

Let us now consider the cone

$$C = \left\{ \boldsymbol{x} \in \mathbb{R}^n \;\middle|\; \boldsymbol{b}_1^\top \boldsymbol{x} \leq 0 \wedge \cdots \wedge \boldsymbol{b}_\ell^\top \boldsymbol{x} \leq 0 \wedge M\boldsymbol{x} \leq 0 \right\}. \tag{3}$$

It is easy to see that $X + C \subseteq X$.

Lemma 24. *Suppose that X is non-empty and that no non-zero integer vector in $\mathbb{Z}^n$ is orthogonal to C. Then $X \cap \mathbb{Z}^n$ is non-empty.*

Proof. Let $\boldsymbol{u} \in X$. Since X is open, there exists $\varepsilon > 0$ such that the open ball $B_\varepsilon(\boldsymbol{u})$ is contained in X. We therefore have that $B_\varepsilon(\boldsymbol{u}) + C \subseteq X$.

We will apply Proposition 23 to show that $B_\varepsilon(\boldsymbol{u}) + C$ contains an integer point and hence that X contains an integer point. To this end, let vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s \in C$ be such that $\operatorname{span}\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s\} = \operatorname{span}(C)$. Then no non-zero vector in $\mathbb{Z}^n$ is orthogonal to $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s$. By Proposition 23, there exist real numbers $\lambda_1, \ldots, \lambda_s \geq 0$ and an integer vector $\boldsymbol{v} \in \mathbb{Z}^n$ such that $\|\boldsymbol{u} + \sum_{i=1}^s \lambda_i \boldsymbol{u}_i - \boldsymbol{v}\|_\infty \leq \varepsilon$. Thus, $\boldsymbol{v} \in B_\varepsilon(\boldsymbol{u}) + C \subseteq X$. □

The contrapositive of the above result states that if X contains no integer point, then there must exist an integer vector that is orthogonal to C. For the desired result, it remains for us to prove the boundedness claim.

Lemma 25. *Suppose that $\boldsymbol{d} \in \mathbb{Z}^n$ is orthogonal to the cone C. Then $\{\boldsymbol{d}^\top \boldsymbol{u} \mid \boldsymbol{u} \in X\}$ is bounded.*

Proof. Define the "enveloping polygon" of X to be

$$\widehat{X} = \left\{ \boldsymbol{x} \in \mathbb{R}^n \;\middle|\; \boldsymbol{b}_1^\top \boldsymbol{x} + \boldsymbol{a}_1 \leq 0 \wedge \cdots \wedge \boldsymbol{b}_\ell^\top \boldsymbol{x} + \boldsymbol{a}_\ell \leq 0 \wedge M\boldsymbol{x} \leq \boldsymbol{c} \right\}.$$

Clearly it holds that $X \subseteq \widehat{X}$. Moreover, by the Minkowski-Weyl decomposition theorem we can write $\widehat{X}$ as a sum $\widehat{X} = B + C$ for B a bounded polygon and C the cone defined in (3). Since $\boldsymbol{d}$ is orthogonal to C by assumption, it follows that $\{\boldsymbol{d}^\top \boldsymbol{u} \mid \boldsymbol{u} \in \widehat{X}\} = \{\boldsymbol{d}^\top \boldsymbol{u} \mid \boldsymbol{u} \in B\}$ is bounded and hence $\{\boldsymbol{d}^\top \boldsymbol{u} \mid \boldsymbol{u} \in X\}$ is bounded. The result immediately follows. □

We can now complete the proof of Lemma 21.

Proof of Lemma 21. By Lemma 24 there exists a non-zero integer vector $\boldsymbol{d} \in \mathbb{Z}^n$ such that $\boldsymbol{d}$ is orthogonal to the cone C defined in (3). Then by Lemma 25 we obtain that $\{\boldsymbol{d}^\top \boldsymbol{u} \mid \boldsymbol{u} \in X\}$ is contained in a bounded interval. □

8. Undecidability for linearly ambiguous automata

In this section we prove Theorem 2. That is, we argue that the emptiness and containment problems are undecidable for the class of linearly ambiguous PA. In fact, We will prove a more general result in Proposition 27.

The proof is done by a reduction from the halting problem for two-counter machines. The reduction resembles the one used to prove undecidability of the comparison problem for another quantitative extension of Boolean automata: max-plus automata [33, 34].

8.1. Two-counter machines

Two-counter machines (or Minsky machines) can be defined in several ways, all equivalent in terms of expressiveness. We use here the following description: A two-counter machine is a deterministic finite-state machine with two counters that can be incremented, decremented, or tested for 0. Formally, it is given by a tuple $(Q, T_1^+, T_2^+, T_1^-, T_2^-, q_{init}, q_{halt})$ where:

- Q is a finite set of states.
- T_1^+ (resp. T_2^+) is a subset of Q^2. If $(p,q) \in T_1^+$ (resp. T_2^+) then there is a transition from the state p to the state q which increments the first counter (resp. second counter).
- T_1^- (resp. T_2^-) is a subset of Q^3. If $(p,q,r) \in T_1^-$ (resp. T_2^-) then there is a transition from the state p which goes to the state q if the current value of the first (resp. second) counter is 0 (it does not change the counters), and which goes to the state r otherwise and decrements the first (resp. second) counter.
- $q_{init} \in Q$ is the initial state and $q_{halt} \in Q$ is the final state such that there is no outgoing transition from q_{halt} (for all transitions $(p,q) \in T_1^+ \cup T_2^+$ or $(p,q,r) \in T_1^- \cup T_2^-$, $q \neq q_{halt}$). We also assume that $q_{init} \neq q_{halt}$.

Moreover the machine is deterministic: for every state there is at most one action that can be performed, i.e. for all $p \in Q$, there is at most one transition of the form (p,q) or (p,q,r) in $T_1^+ \cup T_2^+ \cup T_1^- \cup T_2^-$ and $T_1^+ \cap T_2^+ = \emptyset$ and $T_1^- \cap T_2^- = \emptyset$.

The semantics of a two-counter machine are given by means of the valuations of the counters that are pairs of non-negative integers. An execution with counters initialised to (n_1^0, n_2^0) is a sequence of compatible transitions and valuations denoted by

$$(n_1^0, n_2^0) \xrightarrow{t_1} (n_1^1, n_2^1) \xrightarrow{t_2} (n_1^2, n_2^2) \xrightarrow{t_3} \cdots \xrightarrow{t_k} (n_1^k, n_2^k)$$

such that:

- for all $i \in \{1, \ldots, k\}$, if $t_i \in T_1^+$ (resp. T_2^+), then $n_1^i = n_1^{i-1} + 1$ and $n_2^i = n_2^{i-1}$ (resp. $n_1^i = n_1^{i-1}$ and $n_2^i = n_2^{i-1} + 1$);
- for all $i \in \{1, \ldots, k\}$, if $t_i \in T_1^-$ (resp. T_2^-), then $n_1^i = n_1^{i-1} = 0$ or $n_1^i = n_1^{i-1} - 1$ and $n_2^i = n_2^{i-1}$ (resp. $n_2^i = n_2^{i-1} = 0$ or $n_2^i = n_2^{i-1} - 1$ and $n_1^i = n_1^{i-1}$);
- for all $i \in \{1, \ldots, k-1\}$, if $t_i = (p_i, q_i) \in T_1^+ \cup T_2^+$ then $t_{i+1} \in \{q_i\} \times (Q \cup Q^2)$;
- for all $i \in \{1, \ldots, k-1\}$, if $t_i = (p_i, q_i, r_i) \in T_1^- \cup T_2^-$ and $n_{i-1} = 0$ then $t_{i+1} \in \{q_i\} \times (Q \cup Q^2)$, otherwise if $n_{i-1} \neq 0$ then $t_{i+1} \in \{r_i\} \times (Q \cup Q^2)$.

We say that the machine *halts* if there is a (unique) execution with counters initialised to $(0,0)$ starting in q_{init} reaching the state q_{halt}.

Proposition 26 ([35])**.** *The halting problem for two-counter machines is undecidable.*

8.2. Reduction from the halting problem for two-counter machines

Proposition 27. *Given a two counter machine, one can construct a linearly ambiguous probabilistic automaton $\mathcal{A}$ such that the machine halts if and only if there exists a word w such that $[\![\mathcal{A}]\!](w) \geq \frac{1}{2}$ (resp. $>$, $\leq$, $<$).*

Theorem 2 is an immediate corollary of Proposition 27 since it is trivial to construct a PA that outputs probability $\frac{1}{2}$ for all words.

We follow these steps:

1. We construct two linearly ambiguous PA $\mathcal{A}$ and $\mathcal{B}$ such that the machine halts if and only if there is a word w such that $[\![\mathcal{A}]\!](w) \leq [\![\mathcal{B}]\!](w)$.

2. From $\mathcal{A}$ and $\mathcal{B}$, we construct $\mathcal{A}'$ and $\mathcal{B}'$, also linearly ambiguous, such that the machine halts if and only if there is a word w such that $[\![\mathcal{A}']\!](w) < [\![\mathcal{B}']\!](w)$.

3. We show that the functions $1 - [\![\mathcal{A}]\!]$, $1 - [\![\mathcal{B}]\!]$, $1 - [\![\mathcal{A}']\!]$, $1 - [\![\mathcal{B}']\!]$ are also computed by linearly ambiguous PA.

We show that Proposition 27 follows from steps 1, 2, and 3. We have that for all words w

$$\begin{aligned} [\![\mathcal{A}]\!](w) \leq [\![\mathcal{B}]\!](w) \quad &\iff \quad \frac{1}{2}[\![\mathcal{A}]\!](w) + \frac{1}{2}(1 - [\![\mathcal{B}]\!](w)) \leq \frac{1}{2} \\ &\iff \quad \frac{1}{2}[\![\mathcal{B}]\!](w) + \frac{1}{2}(1 - [\![\mathcal{A}]\!](w)) \geq \frac{1}{2}. \end{aligned}$$

Since $1 - [\![\mathcal{A}]\!]$ and $1 - [\![\mathcal{B}]\!]$ are linearly ambiguous by 3, then, $\frac{1}{2}[\![\mathcal{A}]\!] + \frac{1}{2}(1 - [\![\mathcal{B}]\!])$ and $\frac{1}{2}[\![\mathcal{B}]\!] + \frac{1}{2}(1 - [\![\mathcal{A}]\!])$ are also linearly ambiguous. From 1, we get that the two variants with non-strict inequalities of the problem are undecidable. We proceed similarly using $\mathcal{A}'$ and $\mathcal{B}'$ to prove undecidability of the two other variants with strict inequalities.

Step 1

Let $T = T_1^+ \cup T_2^+ \cup T_1^- \cup T_2^-$ and $\Sigma = \{a, b\} \cup T$. Some words over Σ will be considered as encoding execution of the two counter machine as follows: A block a^m (resp. b^m) encodes the fact that the value of the first (resp. second) counter is m. For example, given $(p, q) \in T_1^+$ and $(q, r, s) \in T_2^-$, a word $a^n b^m (p, q) a^{n+1} b^m (q, r, s) a^{n+1} b^{m'}$, encodes an execution starting with value n in the first counter and m in the second counter. Transition (p, q) then increases the value of the first counter to $n + 1$ without changing the value of the second one. The configuration is thus encoded by the infix $a^{n+1} b^m$. Next, transition (q, r, s) is taken, and either $m' = m = 0$ or $m' = m - 1$.

The PA $\mathcal{A}$ and $\mathcal{B}$ are constructed in such a way that for all words w it holds that

$$\begin{cases} [\![\mathcal{A}]\!](w) = [\![\mathcal{B}]\!](w) & \text{if } w \text{ represents a halting execution of the machine} \\ [\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w) & \text{otherwise.} \end{cases}$$

The automata $\mathcal{A}$ and $\mathcal{B}$ are constructed as a weighted sum of seven PA, each of them checking some criteria that a word w should fulfill in order to represent the halting execution.

Automaton $\mathcal{A}_0$. Conditions such as asking that the word is of the good shape, i.e. contains alternating blocks of a's, blocks of b's and letters from T, with letters from T being state-compatible transitions, that it starts in q_{init} and ends in q_{halt} are all regular conditions that can thus be checked by a (deterministic Boolean) automaton. The exhaustive list of such conditions and their explanations are given below. We then define $\mathcal{A}_0$ to be a deterministic PA such that $[\![\mathcal{A}_0]\!](w) = 0$ if and only if w satisfies all these regular conditions and $[\![\mathcal{A}_0]\!](w) = 1$ otherwise.

We give here a precise description of the conditions checked by $\mathcal{A}_0$.

1. The word w belongs to $T_{init}((a^*b^*)T)^*$ where T_{init} is the subset of T of the transitions started in q_{init}.

2. The word w represents an execution ending in q_{halt}, i.e. it either ends

   - with a letter (q, q_{halt}),
   - with a word of the form $ta^n(q, q_{halt}, r)$ where $t \in T$, $(q, q_{halt}, r) \in T_2^-$ and n is a non-negative integer (resp. $tb^n(q, q_{halt}, r)$ where $t \in T$, $(q, q_{halt}, r) \in T_1^-$ and n a non-negative integer),
   - or with a word of the form $ta^nb^m(q, r, q_{halt})$ where $t \in T$, $(q, r, q_{halt}) \in T_2^-$ and m a positive integer (resp. $ta^nb^m(q, r, q_{halt})$ where $t \in T$, $(q, r, q_{halt}) \in T_1^-$ and n a positive integer).

3. The transitions are state-compatible, i.e. if w contains a factor $(p, q)a^nb^mt$ with $t \in T$ then t starts in q and if w contains a factor $t'a^nb^m(p, q, r)a^{n'}b^{m'}t$ with $t, t' \in T$ and $(p, q, r) \in T_1^-$ (resp. T_2^-) then t starts in q if $n = 0$ and in r if $n > 0$ (resp. t starts in q if $m = 0$ and in r if $m > 0$).

4. We also check that if in the execution represented by the word, at some point the value in the first (resp. second) counter is 0 and a transition from T_1^- (resp. T_2^-) is taken then the value in the counter is still 0 after the transition. In terms of words, this means that if $tb^mt'a^nb^{m'}t''$ is a factor of the word with $t, t'' \in T$ and $t' \in T_1^-$ then $n = 0$ (and similarly for the second counter).

We define a *proper* word w as one of the good shape as given above. The automaton $\mathcal{A}_0$ will make sure that $[\![\mathcal{A}]\!](w) = 0$ only if w is proper. By combining $\mathcal{A}_0$ with the automata constructed below in a clever way (explained a bit later), we are able to focus only on proper words from now - the semantics of the automata constructed below on non-proper words have no importance. For a proper word to encode the unique halting execution - if it exists - we are now left to check that the counters are properly incremented and decremented

Automata $\mathcal{A}_1$ and $\mathcal{B}_1$. The automata $\mathcal{A}_1$ and $\mathcal{B}_1$ check that in a proper word, the first counter is always correctly incremented after reading transitions from T_1^+. The automata $\mathcal{A}_1$ and $\mathcal{B}_1$ will be build on top of the automaton $\mathcal{C}(x, y, z)$ in Figure 4. The automaton $\mathcal{C}(x, y, z)$ is parameterised by three probability variables $x, y, z > 0$. The parameter x is the probability used by the initial distribution, and parameters y and z are used by some transitions.

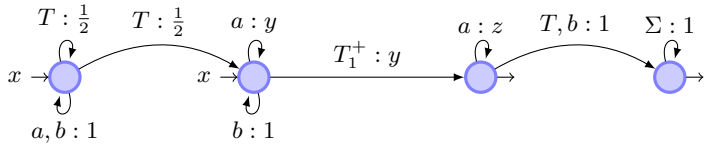

Figure 4: Gadget automaton $\mathcal{C}(x, y, z)$ used to check if the first counter is incremented properly.

As explained above, we will carry out our analysis only on proper words as given by the automaton $\mathcal{A}_0$. The semantics of $\mathcal{C}(x, y, z)$ on non-proper words

will have no impact in our construction. Notice that the only non-deterministic transitions in $\mathcal{C}(x,y,z)$ are the ones going out from the leftmost state upon reading letters from T. It follows that $\mathcal{C}(x,y,z)$ is linearly ambiguous. In fact, for every position in a proper word w labelled by an element t from T_1^+ there is a unique accepting run that first reaches a final state upon reading t. By construction, if $w = a^{n_1}b^{m_1}t_1a^{n_2}b^{m_2}t_2\dots a^{n_k}b^{m_k}t_k$, where $t_i \in T$, and n_i and m_i are non-negative integers, for $i = 1,\dots,k$, then we have

$$[\![\mathcal{C}(x,y,z)]\!](w) \quad = \quad \sum_{t_i \in T_1^+} x\left(\frac{1}{2}\right)^{i-1} y^{n_i+1}z^{n_{i+1}}. \tag{4}$$

Let $x = \frac{1}{2}$, $y = 1$ and $z = \frac{1}{4}$. We define $\mathcal{B}_1$ as $\mathcal{C}(x,x,x)$ and $\mathcal{A}_1$ as a weighted sum of $\mathcal{C}(x,y,z)$ and $\mathcal{C}(x,z,y)$ with weights $(\frac{1}{2},\frac{1}{2})$. Since $\mathcal{C}(\cdot,\cdot,\cdot)$ is linearly ambiguous the obtained automata are also linearly ambiguous. We prove that $[\![\mathcal{A}_1]\!](w) = [\![\mathcal{B}_1]\!](w)$ only if $n_i + 1 = n_{i+1}$ for all i such that $t_i \in T_1^+$ and $[\![\mathcal{A}_1]\!](w) > [\![\mathcal{B}_1]\!](w)$ otherwise.

By (4) it suffices to show that for every i it holds that

$$\left(\frac{1}{2}\right)^{n_i+1+n_{i+1}} \quad \leq \quad \frac{1}{2}\left(\left(\frac{1}{4}\right)^{n_{i+1}} + \left(\frac{1}{4}\right)^{n_i+1}\right)$$

and that the equality holds only if $n_i+1 = n_{i+1}$. Let $p = \left(\frac{1}{2}\right)^{n_i+1}$ and $q = \left(\frac{1}{2}\right)^{n_{i+1}}$ then this reduces to

$$pq \leq \frac{1}{2}\left(p^2+q^2\right).$$

This is true for every p, q and moreover the equality holds if and only if $p = q$, which is equivalent to $n_i + 1 = n_{i+1}$. We conclude with the following remark that will be useful for Step 2.

Remark 2. If $[\![\mathcal{A}_1]\!](w) > [\![\mathcal{B}_1]\!](w)$ then $[\![\mathcal{A}_1]\!](w) \geq [\![\mathcal{B}_1]\!](w) + \left(\frac{1}{2}\right)^{2(|w|+1)}$. To obtain this bound notice that the probabilities and initial distribution of the automata $\mathcal{A}_1$ and $\mathcal{B}_1$ are 0, 1, $\frac{1}{2}$ and $\frac{1}{4}$. Hence for every word w the probability assigned to it is either 0 or a multiple of $\left(\frac{1}{4}\right)^{|w|+1}$. It follows that if $\mathcal{A}_1$ and $\mathcal{B}_1$ assign different probabilities to w then the difference is at least $\left(\frac{1}{4}\right)^{|w|+1}$, which proves the remark.

Automata $\mathcal{A}_2$, $\mathcal{B}_2$, ... Similarly, we construct automata $\mathcal{A}_2$, $\mathcal{B}_2$, $\mathcal{A}_3$, $\mathcal{B}_3$, $\dots\mathcal{A}_6$, $\mathcal{B}_6$ to check the other criteria:

- the value of the second counter is correctly incremented when taking a transition from T_2^+,
- the value of the first (resp. second) counter remains the same when using a transition from $T_2^+ \cup T_2^-$ (resp. $T_1^+ \cup T_1^-$),
- the first (resp. second) counter is correctly decremented when using a transition from T_1^- (resp. T_2^-) and the current value is not 0.

For example we define $\mathcal{A}_5$ and $\mathcal{B}_5$ to check if the decrements of the first counter are correct using the gadget automaton $\mathcal{D}(x,y,z)$ in Figure 5. Let $x = \frac{1}{2}$,

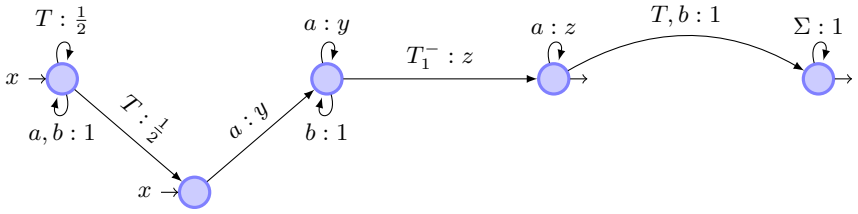

Figure 5: Gadget automaton $\mathcal{D}(x,y,z)$ used to check if the first counter is decremented properly.

$y = 1$ and $z = \frac{1}{4}$. We define $\mathcal{B}_5$ as $\mathcal{C}(x,x,x)$ and $\mathcal{A}_5$ as a weighted sum of $\mathcal{D}(x,y,z)$ and $\mathcal{D}(x,z,y)$ with weights $(\frac{1}{2},\frac{1}{2})$.

For all these automata $\mathcal{A}_1$, $\mathcal{B}_1$, $\ldots\mathcal{A}_6$, $\mathcal{B}_6$, we have that for every proper word w, $[\![\mathcal{A}_i]\!](w) = [\![\mathcal{B}_i]\!](w)$ if the corresponding increments or decrements are properly performed, and $[\![\mathcal{A}_i]\!](w) > [\![\mathcal{B}_i]\!](w)$ otherwise. We remark again that, in that case, for all i, we have $[\![\mathcal{A}_i]\!](w) \geq [\![\mathcal{B}_i]\!](w) + \left(\frac{1}{2}\right)^{2(|w|+1)}$. Note also that all the automata constructed above are linearly ambiguous (the only non-deterministic choices are in the first states when reading T).

Let us define $\mathcal{A}$ (resp. $\mathcal{B}$) as the weighted sum of the above automata computing the function $\frac{7}{13}[\![\mathcal{A}_0]\!] + \frac{1}{13}[\![\mathcal{A}_1]\!] + \cdots + \frac{1}{13}[\![\mathcal{A}_6]\!]$ (resp. $\frac{1}{13}[\![\mathcal{B}_1]\!] + \cdots + \frac{1}{13}[\![\mathcal{B}_6]\!]$). The PA $\mathcal{A}$ and $\mathcal{B}$ are linearly ambiguous.

Fact 28. *For every word w that does not represent the halting execution exactly one of the two cases below applies:*

- *either w is not proper, $[\![\mathcal{A}_0]\!](w) = 1$, and $[\![\mathcal{A}]\!](w) \geq \frac{7}{13} > \frac{6}{13} \geq [\![\mathcal{B}]\!](w)$;*
- *or w is proper but the counters are not properly valued in at least one place, and in that case, there exists an $i \geq 1$ such that $\mathcal{A}_i(w) > \mathcal{B}_i(w)$ (and $\geq$ for the other i). We thus obtain that $[\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w)$.*

The inequalities in the first bullet follow from defining $[\![\mathcal{A}]\!]$ and $[\![\mathcal{B}]\!]$ as weighted sums of probabilistic automata. Remark that in the case when $[\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w)$ we have $[\![\mathcal{A}]\!](w) \geq [\![\mathcal{B}]\!](w) + \frac{1}{13}\left(\frac{1}{2}\right)^{2(|w|+1)}$.

Fact 29. *If the two-counter machine halts then for the word representing the halting execution we have that $[\![\mathcal{A}]\!](w) = [\![\mathcal{B}]\!](w)$.*

Step 2

As noticed previously, by construction of $\mathcal{A}$ and $\mathcal{B}$, we have that for all words w the following hold.

$$\begin{aligned} & [\![\mathcal{A}]\!](w) > [\![\mathcal{B}]\!](w) \\ \iff & [\![\mathcal{A}]\!](w) \geq [\![\mathcal{B}]\!](w) + \frac{1}{13}\left(\frac{1}{2}\right)^{2(|w|+1)} \\ \iff & \frac{1}{2}[\![\mathcal{A}]\!](w) \geq \frac{1}{2}[\![\mathcal{B}]\!](w) + \frac{1}{26}\left(\frac{1}{2}\right)^{2(|w|+1)} \end{aligned} \tag{5}$$

Both functions in the last inequality are computed by linearly ambiguous PA. To obtain the weighted sum of the right hand side we need to construct an

automaton that outputs, for every word w, the probability $\frac{1}{13}\left(\frac{1}{2}\right)^{2(|w|+1)}$. This is easy to obtain by slightly modifying the example in Figure 1. Notice that this is an unambiguous automaton and its complement is linearly ambiguous.

For the rest of this section, we denote by $\mathcal{A}'$ and $\mathcal{B}'$ the PA for the right and left sides of the inequality (5), respectively.

Step 3

We will now argue that $1 - [\![\mathcal{A}]\!]$, $1 - [\![\mathcal{B}]\!]$, $1 - [\![\mathcal{A}']\!]$, $1 - [\![\mathcal{B}']\!]$ are also computed by linearly ambiguous PA. In Figure 6, we show the complement automaton of the gadget $\mathcal{C}(x, y, z)$ from Figure 4. We show that the so-obtained automaton is linearly ambiguous. In the end we will conclude the argument using the property that all components $\mathcal{A}_i$ and $\mathcal{B}_i$ for $i > 0$ are weighted sums of such gadgets.

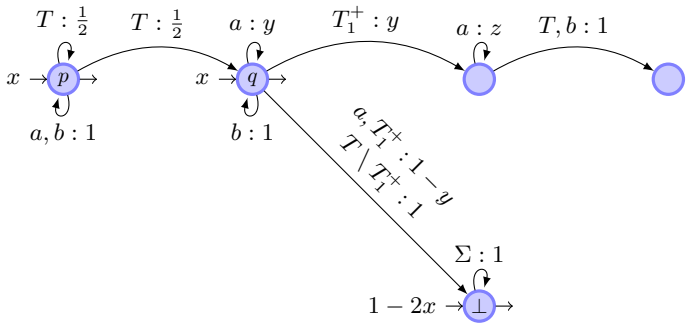

Figure 6: Complement automaton of $\mathcal{C}(x, y, z)$ after trimming.

First, notice that we can trim the automaton to have only three states p, q and $\bot$ because the remaining states are not accepting and it is not possible to reach an accepting state from them. Recall that the ambiguity of an automaton relies only on its underlying structure (Section 2.1). Hence we can focus on the Boolean automaton (i.e. without probabilities) from Figure 7 and analyze its ambiguity. In the trimmed automaton there are only two places with non-deterministic choices: when reading a letter in T from p the automaton can either remain in p or move to q; when reading a from q the automaton can either remain in q or move to $\bot$.

Notice that all three states are both initial and accepting. Let us decompose the set of accepting runs of the automaton on a word w depending on where the run starts and where the run ends, which is 9 cases in total. We focus only on the case for run starting in p and ending in $\bot$; the remaining cases obviously provide at most a linear number of runs. The automaton can move from p to q only when reading an element of T. Fix a word w and consider positions i and i' such that $t_i \in T$ and i' is maximal such that positions between i and i' have labels from $\Sigma \setminus T$ (i.e. position i' is labelled with a letter from T or is equal to $|w| + 1$). We show that there are at most $f(i) \stackrel{\text{def}}{=} i' - i + 1$ accepting runs starting from p and ending in q after reading t_i. This is because in state q, when reading an element from T, the automaton has to move to $\bot$. Hence the number of all accepting runs from p to $\bot$ is bounded by the sum of all $f(i)$ through all positions $1 \leq i \leq |w|$ such that $t_i \in T$. We conclude that the automaton is linearly ambiguous.

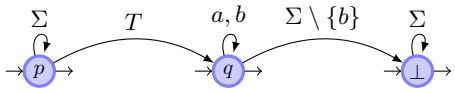

Figure 7: Trimmed version of the automaton from Figure 6; the transition probabilities have been omitted.

To conclude our argument about the ambiguity of all constructed automata, recall that automata $\mathcal{A}$, $\mathcal{B}$, $\mathcal{A}'$, and $\mathcal{B}'$ are constructed as weighted sums of PA obtained from $\mathcal{A}_i$ and $\mathcal{B}_j$, which are weighted sums of gadgets like $\mathcal{C}(x, y, z)$. For example the function $1 - [\![\mathcal{A}]\!]$ is

$$\begin{aligned} 1 - [\![\mathcal{A}]\!] &= 1 - \left(\frac{7}{13}[\![\mathcal{A}_0]\!] + \frac{1}{13}[\![\mathcal{A}_1]\!] + \cdots + \frac{1}{13}[\![\mathcal{A}_6]\!] \right) \\ &= \frac{7}{13}(1 - [\![\mathcal{A}_0]\!]) + \frac{1}{13}(1 - [\![\mathcal{A}_1]\!]) + \cdots + \frac{1}{13}(1 - [\![\mathcal{A}_6]\!]) \end{aligned}$$

It is thus sufficient to complement each member of the sum. Showing that all these complements are still linearly ambiguous follows the ideas given above for the automaton in Figure 6.

9. Conclusion

In this work we have given a conditional proof of the containment problem for PA being decidable if one of the automata is finitely ambiguous and the other one is unambiguous. Interestingly, for one of the two cases, our proposed algorithm uses a satisfiability oracle for a theory whose decidability is equivalent to a weak form of Schanuel's conjecture. We have complemented our decidability results with a proof of undecidability for the case when the given automata are linearly ambiguous.

Decidability of the containment problem when both automata are allowed to be finitely ambiguous remains open. One way to tackle it is to study generalizations of the IP+EXP problem introduced in Section 7. This problem asks whether there exists $\boldsymbol{x} \in \mathbb{N}^n$ such that $f(\boldsymbol{x}) < 1$ and $M\boldsymbol{x} < \boldsymbol{c}$ for a given function f defined using exponentiations, a given matrix M, and vector $\boldsymbol{c}$. A natural way to extend the latter would be to ask that $f(\boldsymbol{x}) < g(\boldsymbol{x})$, where g is obtained in a similar way as f. The main obstacle, when trying to generalize our decidability proof for that problem, is that we lack a replacement for the cone C needed in order to obtain a result similar to Lemma 25 using the Minkowski-Weyl decomposition.

Acknowledgements

This work was supported by the EPSRC grant EP/P020992/1 and the EPSRC fellowship EP/N008197/1. R. Lazić was also supported by a Leverhulme Trust Research Fellowship RF-2017-579; F. Mazowiecki, by the French National Research Agency (ANR) in the frame of the "Investments for the future" Programme IdEx Bordeaux (ANR-10-IDEX-03-02); G. A. Pérez, by an F.R.S.-FNRS

Aspirant fellowship and an FWA postdoc fellowship. L.Daviaud was affiliated to the University of Warwick when this work started and is now supported by the EPSRC grant EP/T018313/1.

We thank Shaull Almagor and Ismaël Jecker for some helpful remarks.

References

[1] M. O. Rabin, Probabilistic automata, Information and Control 6 (3) (1963) 230–245. doi:10.1016/S0019-9958(63)90290-0.

[2] M. L. Puterman, Markov Decision Processes, Wiley-Interscience, 2005.

[3] L. E. Baum, T. Petrie, Statistical inference for probabilistic functions of finite state Markov chains, Ann. Math. Statist. 37 (6) (1966) 1554–1563. doi:10.1214/aoms/1177699147.

[4] S. J. Russell, P. Norvig, Artificial Intelligence - A Modern Approach (3. internat. ed.), Pearson Education, 2010.

[5] L. P. Kaelbling, M. L. Littman, A. W. Moore, Reinforcement learning: A survey, Journal of Artificial Intelligence Research 4 (1996) 237–285. doi:10.1613/jair.301.

[6] M. Y. Vardi, Automatic verification of probabilistic concurrent finite-state programs, in: 26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985, IEEE Computer Society, 1985, pp. 327–338. doi:10.1109/SFCS.1985.12.

[7] M. Z. Kwiatkowska, G. Norman, D. Parker, H. Qu, Assume-guarantee verification for probabilistic systems, in: J. Esparza, R. Majumdar (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings, Vol. 6015 of Lecture Notes in Computer Science, Springer, 2010, pp. 23–37. doi:10.1007/978-3-642-12002-2_3.

[8] L. Feng, T. Han, M. Z. Kwiatkowska, D. Parker, Learning-based compositional verification for synchronous probabilistic systems, in: T. Bultan, P. Hsiung (Eds.), Automated Technology for Verification and Analysis, 9th International Symposium, ATVA 2011, Taipei, Taiwan, October 11-14, 2011, Vol. 6996 of Lecture Notes in Computer Science, Springer, 2011, pp. 511–521. doi:10.1007/978-3-642-24372-1_40.

[9] K. V. Palem, L. Avinash, Ten years of building broken chips: The physics and engineering of inexact computing, ACM Transactions on Embedded Computing Systems 12 (2s) (2013) 87:1–87:23. doi:10.1145/2465787.2465789.

[10] A. Yakaryilmaz, A. C. C. Say, Unbounded-error quantum computation with small space bounds, Information and Computation 209 (6) (2011) 873–892. doi:10.1016/j.ic.2011.01.008.

[11] H. Giese, N. Bencomo, L. Pasquale, A. J. Ramirez, P. Inverardi, S. Wätzoldt, S. Clarke, Living with uncertainty in the age of runtime models, in: N. Bencomo, R. B. France, B. H. C. Cheng, U. Assmann (Eds.), Models@run.time - Foundations, Applications, and Roadmaps [Dagstuhl Seminar 11481, November 27 - December 2, 2011], Vol. 8378 of Lecture Notes in Computer Science, Springer, 2014, pp. 47–100. doi:10.1007/978-3-319-08915-7_3.

[12] M. Mohri, F. Pereira, M. Riley, Weighted finite-state transducers in speech recognition, Computer Speech & Language 16 (1) (2002) 69–88. doi:10.1006/csla.2001.0184.

[13] M. P. Schützenberger, On the definition of a family of automata, Information and Control 4 (2-3) (1961) 245–270.

[14] W. Tzeng, A polynomial-time algorithm for the equivalence of probabilistic automata, SIAM J. Comput. 21 (2) (1992) 216–227.

[15] N. Fijalkow, H. Gimbert, E. Kelmendi, Y. Oualhadj, Deciding the value 1 problem for probabilistic leaktight automata, Logical Methods in Computer Science 11 (2) (2015). doi:10.2168/LMCS-11(2:12)2015.

[16] R. Chadha, A. P. Sistla, M. Viswanathan, Y. Ben, Decidable and expressive classes of probabilistic automata, in: A. M. Pitts (Ed.), Foundations of Software Science and Computation Structures - 18th International Conference, FoSSaCS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings, Vol. 9034 of Lecture Notes in Computer Science, Springer, 2015, pp. 200–214. doi:10.1007/978-3-662-46678-0_13.

[17] N. Fijalkow, C. Riveros, J. Worrell, Probabilistic automata of bounded ambiguity, in: R. Meyer, U. Nestmann (Eds.), 28th International Conference on Concurrency Theory, CONCUR 2017, September 5-8, 2017, Berlin, Germany, Vol. 85 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017, pp. 19:1–19:14. doi:10.4230/LIPIcs.CONCUR.2017.19.

[18] N. Fijalkow, Undecidability results for probabilistic automata, SIGLOG News 4 (4) (2017) 10–17. doi:10.1145/3157831.3157833.

[19] A. Paz, Introduction to probabilistic automata, Academic Press, 1971.

[20] V. D. Blondel, V. Canterini, Undecidable problems for probabilistic automata of fixed dimension, Theory Comput. Syst. 36 (3) (2003) 231–245.

[21] S. Akshay, T. Antonopoulos, J. Ouaknine, J. Worrell, Reachability problems for Markov chains, Inf. Process. Lett. 115 (2) (2015) 155–158.

[22] A. Condon, R. J. Lipton, On the complexity of space bounded interactive proofs (extended abstract), in: 30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989, IEEE Computer Society, 1989, pp. 462–467. doi:10.1109/SFCS.1989.63519.

[23] O. Goldreich, On promise problems (a survey in memory of Shimon Even [1935-2004]), Electronic Colloquium on Computational Complexity (ECCC) (018) (2005).
URL http://eccc.hpi-web.de/eccc-reports/2005/TR05-018/index.html

[24] H. Derksen, E. Jeandel, P. Koiran, Quantum automata and algebraic groups, J. Symb. Comput. 39 (3-4) (2005) 357–371.

[25] D. Kirsten, S. Lombardy, Deciding unambiguity and sequentiality of polynomially ambiguous min-plus automata, in: S. Albers, J.-Y. Marion (Eds.), 26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009, February 26-28, 2009, Freiburg, Germany, Proceedings, Vol. 3 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009, pp. 589–600. doi:10.4230/LIPIcs.STACS.2009.1850.

[26] A. Mottet, K. Quaas, The containment problem for unambiguous register automata, in: R. Niedermeier, C. Paul (Eds.), 36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany, Vol. 126 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, pp. 53:1–53:15. doi:10.4230/LIPIcs.STACS.2019.53.
URL https://doi.org/10.4230/LIPIcs.STACS.2019.53

[27] A. Weber, H. Seidl, On the degree of ambiguity of finite automata, Theor. Comput. Sci. 88 (2) (1991) 325–349.

[28] A. Macintyre, A. J. Wilkie, On the decidability of the real exponential field, in: P. Odifreddi (Ed.), Kreiseliana. About and Around Georg Kreisel, AK Peters, 1996, pp. 441–467.

[29] A. Weber, H. Seidl, On the degree of ambiguity of finite automata, Theoretical Computer Science 88 (2) (1991) 325–349. doi:10.1016/0304-3975(91)90381-B.

[30] C. Rackoff, The covering and boundedness problems for vector addition systems, Theoretical Compututer Science 6 (1978) 223–231. doi:10.1016/0304-3975(78)90036-1.

[31] H. Cohen, A course in computational algebraic number theory, Vol. 138 of Graduate texts in mathematics, Springer, 1993.

[32] L. Khachiyan, L. Porkolab, Computing integral points in convex semi-algebraic sets, in: 38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997, IEEE Computer Society, 1997, pp. 162–171. doi:10.1109/SFCS.1997.646105.

[33] T. Colcombet, On distance automata and regular cost function, presented at the Dagstuhl seminar "Advances and Applications of Automata on Words and Trees" (2010).

[34] S. Almagor, U. Boker, O. Kupferman, What's decidable about weighted automata?, in: T. Bultan, P. Hsiung (Eds.), Automated Technology for Verification and Analysis, 9th International Symposium, ATVA 2011,

Taipei, Taiwan, October 11-14, 2011. Proceedings, Vol. 6996 of Lecture Notes in Computer Science, Springer, 2011, pp. 482–491. doi:10.1007/978-3-642-24372-1_37.

[35] M. L. Minsky, Computation: Finite and Infinite Machines, Prentice-Hall, 1967.