Non-deterministic weighted automata
evaluated over Markov chains

Jakub Michaliszyn
University of Wrocław
jmi@cs.uni.wroc.pl

Jan Otop
University of Wrocław
jotop@cs.uni.wroc.pl

Abstract
We present the first study of non-deterministic weighted automata under probabilistic semantics. In this semantics words are random events, generated by a Markov chain, and functions computed by weighted automata are random variables. We consider the probabilistic questions of computing the expected value and the cumulative distribution for such random variables.

The exact answers to the probabilistic questions for non-deterministic automata can be irrational and are uncomputable in general. To overcome this limitation, we propose approximation algorithms for the probabilistic questions, which work in exponential time in the size of the automaton and polynomial time in the size of the Markov chain and the given precision. We apply this result to show that non-deterministic automata can be effectively determinised with respect to the standard deviation metric.

2012 ACM Subject Classification Theory of computation → Automata over infinite objects; Theory of computation → Quantitative automata

Keywords and phrases quantitative verification, weighted automata, expected value

Funding The National Science Centre (NCN), Poland under grant 2017/27/B/ST6/00299.

1 Introduction

Weighted automata are (non-deterministic) finite automata in which transitions carry weights [16]. We study here weighted automata (on finite and infinite words) whose semantics is given by value functions (such as the sum or the average) [11]. In such weighted automata transitions are labeled with rational numbers and hence every run yields a sequence of rationals, which the value function aggregates into a single (real) number. This number is the value of the run, and the value of a word is the infimum over the values of all accepting runs on that word.

The value function approach has been introduced to express quantitative system properties (performance, energy consumption, etc.) and it serves as a foundation for quantitative verification [11, 21]. Basic decision questions for weighted automata are quantitative counterparts of the emptiness and universality questions obtained by imposing a threshold on the values of words.

Probabilistic semantics. The emptiness and the universality problems correspond to the best-case and the worst-case analysis. For the average-case analysis, weighted automata are considered under probabilistic semantics, in which words are random events generated by a Markov chain [10] [12]. In such a setting, functions from words to reals computed by deterministic weighted automata are measurable and hence can be considered as random
variables. The fundamental probabilistic questions are to compute the expected value and the cumulative distribution for a given automaton and a Markov chain.

The deterministic case. Weighted automata under probabilistic semantics have been studied only in the deterministic case. A close relationship has been established between weighted automata under probabilistic semantics and weighted Markov chains [10]. For a weighted automaton $A$ and a Markov chain $M$ representing the distribution over words, the probabilistic problems for $A$ and $M$ coincide with the probabilistic problem of the weighted Markov chain $A \times M$. Weighted Markov chains have been intensively studied with single and multiple quantitative objectives [4, 13, 18, 29]. The above reduction does not extend to non-deterministic weighted automata [12, Example 30].

Significance of nondeterminism. Non-deterministic weighted automata are provably more expressive than their deterministic counterpart [11]. Many important system properties can be expressed with weighted automata only in the nondeterministic setting. This includes minimal response time, minimal number of errors and the edit distance problem [21], which serves as the foundation for the specification repair framework from [6].

Non-determinism can also arise as a result of abstraction. The exact systems are often too large and complex to operate on and hence they are approximated with smaller non-deterministic models [14]. The abstraction is especially important for multi-threaded programs, where the explicit model grows exponentially with the number of threads [20].

Our contributions

We study non-deterministic weighted automata under probabilistic semantics. We work with weighted automata as defined in [11], where a value function $f$ is used to aggregate weights along a run, and the value of the word is the infimum over values of all runs. (The infimum can be changed to supremum as both definitions are dual). We primarily focus on the two most interesting value functions: the sum of weights over finite runs, and the limit average over infinite runs. The main results presented in this paper are as follows.

- We show that the answers to the probabilistic questions for weighted automata with the sum and limit-average value functions can be irrational and even transcendental (Theorem 5) and cannot be computed by any effective representation (Theorem 7).
- We establish approximation algorithms for the probabilistic questions for weighted automata with the sum and limit-average value functions. The approximation is $\#P$-complete for (total) weighted automata with the sum value function (Theorem 11), and it is $\text{PSPACE}$-hard and solvable in exponential time for weighted automata with the limit-average value function (Theorem 22).
- We show that weighted automata with the limit-average value function can be approximately determinised (Theorem 24). Given an automaton $A$ and $\epsilon > 0$, we show how to compute a deterministic automaton $A_D$ such that the expected difference between the values returned by both automata is at most $\epsilon$.

Applications

We briefly discuss applications of our contributions in quantitative verification.

- The expected-value question corresponds to the average-case analysis in quantitative verification [10, 12]. Using results from this paper, we can perform the average-case analysis with respect to quantitative specifications given by non-deterministic weighted automata.
Some quantitative-model-checking frameworks [11] are based on the universality problem for non-deterministic automata, which asks whether all words have the value below a given threshold. Unfortunately, the universality problem is undecidable for weighted automata with the sum or the limit average values functions. The distribution question can be considered as a computationally-attractive variant of universality, i.e., we ask whether almost all words have value below some given threshold. We show that if the threshold can be approximated, the distribution question can be computed effectively.

Weighted automata have been used to formally study online algorithms [2]. Online algorithms have been modeled by deterministic weighted automata, which make choices based solely on the past, while offline algorithms have been modeled by non-deterministic weighted automata. Relating deterministic and non-deterministic models allowed for formal verification of the worst-case competitiveness ratio of online algorithms. Using the result from our paper, we can extend the analysis from [2] to the average-case competitiveness.

Related work

The problem considered in this paper is related to the following areas from the literature.

Probabilistic verification of qualitative properties. Probabilistic verification asks for the probability of the set of traces satisfying a given property. For non-weighted automata, it has been extensively studied [33, 15, 4] and implemented [26, 22]. The prevalent approach in this area is to work with deterministic automata, and apply determinisation as needed. To obtain better complexity bounds, the probabilistic verification problem has been directly studied for unambiguous Büchi automata in [5]; the authors explain there the potential pitfalls in the probabilistic analysis of non-deterministic automata.

Weighted automata under probabilistic semantics. Probabilistic verification of weighted automata and their extensions has been studied in [12]. All automata considered there are deterministic.

Markov Decision Processes (MDPs). MDPs are a classical extension of Markov chains, which models control in a stochastic environment [4, 18]. In MDPs, probabilistic and non-deterministic transitions are interleaved; this can be explained as a game between two players: Controller and Environment. Given a game objective (e.g. state reachability), the goal of Controller is to maximize the probability of the objective by selecting non-deterministic transitions. Environment is selecting probabilistic transitions at random w.r.t. a probability distribution described in the current state of the MDP. Intuitively, the non-determinism in MDPs is resolved based on the past, i.e., each time Controller selects a non-deterministic transition, its choice is based on previously picked transitions. Our setting can be also explained in such a game-theoretic framework: first, Environment generates a complete word, and only then non-deterministic choices are resolved by Controller, who generates a run of a given weighted automaton. That non-determinism in a run at some position may depend on letters in the input word on positions past i (i.e., future events).

Partially Observable Markov Decision Process (POMDPs) [3] are an extension of MDPs, which models weaker non-determinism. In this setting, the state space is partitioned into observations and the non-deterministic choices have to be the same for sequences consisting of the same observations (but possibly different states). Intuitively, Controller can make choices based only on the sequence of observations it has seen so far. While in POMDPs Controller is restricted, in our setting Controller is stronger than in the MDPs case.

Non-deterministic probabilistic automata. The combination of nondeterminism with stochas-
Non-deterministic weighted automata evaluated over Markov chains

ticity has been recently studied in the framework of probabilistic automata \cite{32}. There have been defined non-deterministic probabilistic automata (NPA) and there has been proposed two possible semantics for NPA. It has been shown that the equivalence problem for NPA is undecidable (under either of the considered two semantics). Related problems, such as the threshold problem, are undecidable already for (deterministic) probabilistic automata \cite{7}. While NPA work only over finite words, the interaction between probabilistic and non-deterministic transitions is more general than in our framework. In particular, non-determinism in NPA can influence the probability distribution, which is not possible in our framework.

Approximate determinisation. As weighted automata are not determinisable, Boker and Henzinger \cite{9} studied approximate determinisation defined as follows. The distance $d_{\text{sup}}$ between weighted automata $A_1, A_2$ is defined as $d_{\text{sup}}(A_1, A_2) = \sup_w |A_1(w) - A_2(w)|$. A nondeterministic weighted automaton $A$ can be approximately determinised if for every $\epsilon > 0$ there exists a deterministic automaton $A_D$ such that $d_{\text{sup}}(A, A_D) \leq \epsilon$. Unfortunately, weighted automata with the limit average value function cannot be approximately determinised \cite{9}. In this work we show that the approximate determinisation is possible for the standard deviation metric $d_{\text{std}}$ defined as $d_{\text{std}}(A_1, A_2) = E(|A_1(w) - A_2(w)|)$.

This paper is an extended and corrected version of \cite{27}. It contains full proofs, an extended discussion and a stronger version of Theorem \ref{main_thm}. We have showed in \cite{27} that the expected values and the distribution values may be irrational. In this paper we show that these values can be even transcendental (Theorem \ref{thm:transcendental_values}).

We have corrected two claims from \cite{27}. First, we have corrected statements of Theorems \ref{thm:LimAvg} and \ref{thm:LimAvg2}. For LimAvg-automata and the distribution question $\lim_{\text{Av}}(\lambda)$, the values, which can be irrational and uncomputable are not the values of the probability $\lim_{\text{Av}}(\lambda) = P_M(\{w \mid L_A = \lambda\})$, but the values of the thresholds $\lambda$ that correspond to mass points, i.e., values $\lambda$ such that $P_M(\{w \mid L_A = \lambda\}) > 0$. We have also removed from Theorem \ref{thm:LimAvg} PSPACE-hardness claim for the distribution question for (non-total) Sum-automata. We show that the (exact) distribution question for all Sum-automata is $\#P$-complete.

2 Preliminaries

Given a finite alphabet $\Sigma$ of letters, a word $w$ is a finite or infinite sequence of letters. We denote the set of all finite words over $\Sigma$ by $\Sigma^*$. For a word $w$, we define $w[i]$ as the $i$-th letter of $w$, and we define $w[i,j]$ as the subword $w[i]w[i+1]\ldots w[j]$ of $w$. We use the same notation for other sequences defined later on. By $|w|$ we denote the length of $w$.

A (non-deterministic) finite automaton (NFA) is a tuple $(\Sigma, Q, Q_0, F, \delta)$ consisting of an input alphabet $\Sigma$, a finite set of states $Q$, a set of initial states $Q_0 \subseteq Q$, a set of final states $F$, and a finite transition relation $\delta \subseteq Q \times \Sigma \times Q$.

We define $\delta(q, a) = \{q' \in Q \mid \delta(q, a, q')\}$ and $\delta(S, a) = \bigcup_{q \in S} \delta(q, a)$. We extend this to words $\delta : 2^Q \times \Sigma^* \rightarrow 2^Q$ in the following way: $\delta(S, \epsilon) = S$ (where $\epsilon$ is the empty word) and $\delta(S, aw) = \delta(\delta(S, a), w)$, i.e., $\delta(S, w)$ is the set of states reachable from $S$ via $\delta$ over the word $w$.

Weighted automata. A weighted automaton is a finite automaton whose transitions are labeled by rational numbers called weights. Formally, a weighted automaton is a tuple $(\Sigma, Q, Q_0, F, \delta, C)$, where the first five elements are as in the finite automata, and $C : \delta \rightarrow Q$ is a function that defines weights of transitions. An example of a weighted automaton is depicted in Figure \ref{fig:weighted_automaton}.
The size of a weighted automaton $A$, denoted by $|A|$, is $|Q| + |\delta| + \sum_{q,q',a} \text{len}(C(q,a,q'))$, where len is the sum of the lengths of the binary representations of the numerator and the denominator of a given rational number.

A run $\pi$ of an automaton $A$ on a word $w$ is a sequence of states $\pi[0]\pi[1] \ldots$ such that $\pi[0]$ is an initial state and for each $i$ we have $(\pi[i-1], w[i], \pi[i]) \in \delta$. A finite run $\pi$ of length $k$ is accepting if and only if the last state $\pi[k]$ belongs to the set of accepting states $F$. As in [11], we do not consider $\omega$-accepting conditions and assume that all infinite runs are accepting. Every run $\pi$ of an automaton $A$ on a (finite or infinite) word $w$ defines a sequence of weights of successive transitions of $A$ as follows. Let $(C(\pi))[i]$ be the weight of the $i$-th transition, i.e., $C(\pi[i-1], w[i], \pi[i])$. Then, $C(\pi) = (C(\pi)[i])_{1 \leq i \leq |w|}$. A value functions $f$ is a function that assigns real numbers to sequences of rational numbers. The value $f(\pi)$ of the run $\pi$ is defined as $f(C(\pi))$.

The value of a (non-empty) word $w$ assigned by the automaton $A$, denoted by $L_A(w)$, is the infimum of the set of values of all accepting runs on $w$. The value of a word that has no (accepting) runs is infinite. To indicate a particular value function $f$ that defines the semantics, we will call a weighted automaton $A$ an $f$-automaton.

Value functions. We consider the following value functions. For finite runs, functions $\text{MIN}$ and $\text{MAX}$ are defined in the usual manner, and the function $\text{SUM}$ is defined as

$$\text{SUM}(\pi) = \sum_{i=1}^{\text{len}(C(\pi))} (C(\pi))[i].$$

For infinite runs we consider the supremum $\text{SUP}$ and infimum $\text{INF}$ functions (defined like $\text{MAX}$ and $\text{MIN}$ but on infinite runs) and the limit average function $\text{LimAvg}$ defined as

$$\text{LimAvg}(\pi) = \limsup_{k \to \infty} \text{Avg}(\pi[0,k])$$

where for finite runs $\pi$ we have $\text{Avg}(\pi) = \frac{\text{SUM}(\pi)}{|C(\pi)|}$.  

2.1 Probabilistic semantics

A (finite-state discrete-time) Markov chain is a tuple $\langle \Sigma, S, s_0, E \rangle$, where $\Sigma$ is the alphabet of letters, $S$ is a finite set of states, $s_0$ is an initial state, $E: S \times \Sigma \times S \to [0,1]$ is an edge probability function, which for every $s \in S$ satisfies that $\sum_{a \in \Sigma} E(s,a,s') = 1$. An example of a single-state Markov chain is depicted in Figure 1.

In this paper, Markov chains serve as a mathematical model as well as the input to algorithms. Whenever a Markov chain is the input to a problem or an algorithm, we assume that all edge probabilities are rational and the size of a Markov chain $M$ is defined as $|M| = |S| + |E| + \sum_{q,q',a} \text{len}(E(q,a,q'))$.

The probability of a finite word $w$ w.r.t. a Markov chain $M$, denoted by $P_M(u)$, is the sum of the probabilities of paths from $s_0$ labeled by $u$, where the probability of a path is the product of probabilities of its edges. For sets $u, \Sigma^\omega = \{uw \mid w \in \Sigma^\omega\}$, called cylinders, we have $P_M(u \cdot \Sigma^\omega) = P_M(u)$, and then the probability measure over infinite words defined by $M$ is the unique extension of the above measure to the $\sigma$-algebra generated by cylinders (by Carathéodory’s extension theorem [17]). We will denote the unique probability measure defined by $M$ as $P_M$. For example, for the Markov chain $M$ presented in Figure 1 we have that $P_M(ab) = \frac{1}{4}$, and so $P_M(\{w \in \{a,b\}^\omega \mid w[0,1] = ab\}) = \frac{1}{4}$, whereas $P_M(X) = 0$ for any countable set of infinite words $X$.

A function $f: \Sigma^\omega \to \mathbb{R}$ is measureable w.r.t. $P_M$ is called a random variable (w.r.t. $P_M$). A random variable $g$ is discrete, if there exists a countable set $Y \subset \mathbb{R}$ such that $g$
returns a value for $Y$ with probability 1 ($\mathbb{P}(\{w \mid g(w) \in Y\}) = 1$). For the discrete random variable $g$, we define the expected value $\mathbb{E}(g)$ (w.r.t. the measure $\mathbb{P}$) as

$$\mathbb{E}(g) = \sum_{y \in Y} y \cdot \mathbb{P}(\{w \mid g(w) = y\}).$$

Every non-negative random variable $h: \Sigma^* \to \mathbb{R}^+$ is a point-wise limit of some sequence of monotonically increasing discrete random variables $g_1, g_2, \ldots$ and the expected value $\mathbb{E}(h)$ is the limit of expected values $\mathbb{E}(g_i)$ [17]. Finally, every random variable $f$ can be presented as the difference $h_1 - h_2$ of non-negative random variables $h_1, h_2$ and we have $\mathbb{E}(f) = \mathbb{E}(h_1) - \mathbb{E}(h_2)$ [17].

A terminating Markov chain $\mathcal{M}$ is a tuple $\langle \Sigma, S, S_0, E, T \rangle$, where $\Sigma$, $S$, and $s_0$ are as usual, $E: S \times (\Sigma \cup \{\epsilon\}) \times S \rightarrow [0, 1]$ is the edge probability function, such that if $E(s, a, t)$, then $a = \epsilon$ if and only if $t \in T$, and for every $s \in S$ we have $\sum_{\sigma \in \Sigma \cup \{\epsilon\}, s' \in S} E(s, a, s') = 1$, and $T$ is a set of terminating states such that the probability of reaching a terminating state from any state $s$ is positive. Notice that the only $\epsilon$-transitions in a terminating Markov chain are those that lead to a terminating state.

The probability of a finite word $w$ w.r.t. $\mathcal{M}$, denoted $\mathbb{P}_\mathcal{M}(w)$, is the sum of probabilities of paths from $s_0$ labeled by $w$ such that the only terminating state on this path is the last one. Notice that $\mathbb{P}_\mathcal{M}$ is a probability distribution on finite words whereas $\mathbb{P}$ is not (because the sum of probabilities may exceed 1).

A function $f: \Sigma^* \to \mathbb{R}$ is called a random variable (w.r.t. $\mathbb{P}$). Since words generated by $\mathcal{M}$ are finite, the co-domain of $f$ is countable and hence $f$ is discrete. The expected value of $f$ w.r.t. $\mathcal{M}$ is defined in the same way as for non-terminating Markov chains.

**Automata as random variables.** An infinite-word-weighted automaton $A$ defines the function $\mathcal{L}_A$ that assigns each word from $\Sigma^*$ its value $\mathcal{L}_A(w)$. This function is measurable for all the automata types we consider in this paper (see Remark 2 below). Thus, this function can be interpreted as a random variable with respect to the probabilistic space we consider. Hence, for a given automaton $A$ (over infinite words) and a Markov chain $\mathcal{M}$, we consider the following quantities:

| $\mathbb{E}_\mathcal{M}(A)$ — the expected value of the random variable $\mathcal{L}_A$ w.r.t. the measure $\mathbb{P}_\mathcal{M}$. | $\mathbb{D}_{\mathcal{M},A}(\lambda) = \mathbb{P}_\mathcal{M}(\{w \mid \mathcal{L}_A(w) \leq \lambda\})$ — the (cumulative) distribution function of $\mathcal{L}_A$ w.r.t. the measure $\mathbb{P}_\mathcal{M}$. |
| --- | --- |

In the finite words case, the expected value $\mathbb{E}_{\mathcal{M}}$ and the distribution $\mathbb{D}_{\mathcal{M},A}$ are defined in the same manner.

**Remark 1** (Bounds on the expected value and the distribution). Both quantities can be easily bounded: the value of the distribution function $\mathbb{D}_{\mathcal{M},A}$ is always between 0 and 1.

For a LimAvg-automaton $A$, we have $\mathbb{E}_\mathcal{M}(A) \in [\min_A, \max_A] \cup \{\infty\}$, where $\min_A$ and $\max_A$ denote the minimal and the maximal weight of $A$ and $\mathbb{E}_\mathcal{M}(A) = \infty$ if and only if the probability of the set of words with no accepting runs in $A$ is positive. Note that we consider no $\omega$-accepting conditions, and hence all infinite runs of LimAvg-automata are accepting, but there can be infinite words, on which a given LimAvg-automaton has no infinite runs.

For a Sum-automaton $A$, we have $\mathbb{E}_{\mathcal{M}}(A) \in [L_{\mathcal{M}} \cdot \min_A, L_{\mathcal{M}} \cdot \max_A] \cup \{\infty\}$, where $L_{\mathcal{M}}$ is the expected length of a word generated by $\mathcal{M}$ (it can be computed in a standard way [19 Section 11.2]) and, as above, $\mathbb{E}_{\mathcal{M}}(A) = \infty$ if and only if there is a finite word $w$ generated by $\mathcal{M}$ with non-zero probability such that $A$ has no accepting runs on $w$.

We show in Section 3.2 that the distribution and expected value may be irrational, even for integer weights and uniform distributions.
Remark 2 (Measurability of functions represented by automata). For automata on finite words, \( \inf \)-automata and \( \sup \)-automata, measurability of \( \mathcal{L}_A \) is straightforward. To show that \( \mathcal{L}_A(w) : \Sigma^\omega \mapsto \mathbb{R} \) is measurable for any non-deterministic \( \text{LimAvg} \)-automaton \( A \), it suffices to show that for every \( x \in \mathbb{R} \), the preimage \( \mathcal{L}_A^{-1}(\{-\infty, x]\) is measurable. Let \( Q \) be the set of states of \( A \). We can define a subset \( A_x \subseteq \Sigma^\omega \times Q^\omega \) of the pairs, the word and the run on it, where the value of the run is less than or equal to \( x \). We show that \( A_x \) is Borel. For \( p \in \mathbb{N} \), let \( B_p^x \) be the subset of \( \Sigma^\omega \times Q^\omega \) of pairs \( (w, \pi) \) such that up to position \( p \) the sequence \( \pi \) is a run on \( w \) and the average of weights up to \( p \) is at most \( x \). Observe that \( B_p^x \) is an open set and \( A_x \) is equal to \( \bigcap_{x \in \mathbb{R}^+} \bigcup_{p \in \mathbb{N}} \bigcap_{p \geq p_0} B_{p+p_0}^{p_0} \), i.e., \( A_x \) consists of pairs \( (w, \pi) \) satisfying that for every \( \epsilon \in \mathbb{Q}^+ \) there exists \( p_0 \) such that for all \( p \geq p_0 \) the average weight of \( \pi \) at \( p \) does not exceed \( x + \epsilon \) and \( \pi \) is a run on \( w \) (each finite prefix is a run). Finally, \( \mathcal{L}_A^{-1}(\{-\infty, x]\) is the projection of \( A_x \) on the first component \( \Sigma^\omega \). The projection of a Borel set is an analytic set, which is measurable \cite{24}. Thus, \( \mathcal{L}_A \) defined by a non-deterministic \( \text{LimAvg} \)-automaton is measurable.

The above proof of measurability requires some knowledge of descriptive set theory. We will give a direct proof of measurability of \( \mathcal{L}_A \) in the paper (Theorem \cite{22}).

### 2.2 Computational questions

We consider the following basic computational questions:

**The expected value question:** Given an \( f \)-automaton \( A \) and a (terminating) Markov chain \( M \), compute \( E_M(A) \).

**The distribution question:** Given an \( f \)-automaton \( A \), a (terminating) Markov chain \( M \) and a threshold \( \lambda \in \mathbb{Q} \), compute \( D_{M,A}(\lambda) \).

Each of the above questions has its decision variant (useful for lower bounds), where instead of computing the value we ask whether the value is less than a given threshold \( t \).

The above questions have their approximate variants:

**The approximate expected value question:** Given an \( f \)-automaton \( A \), a (terminating) Markov chain \( M \), \( \epsilon \in \mathbb{Q}^+ \), compute a number \( y \in \mathbb{Q} \) such that \( |y - E_M(A)| \leq \epsilon \).

**The approximate distribution question:** Given an \( f \)-automaton \( A \), a (terminating) Markov chain \( M \), a threshold \( \lambda \in \mathbb{Q} \) and \( \epsilon \in \mathbb{Q}^+ \) compute a number \( y \in \mathbb{Q} \) which belongs to \( [D_{M,A}(\lambda - \epsilon), D_{M,A}(\lambda + \epsilon)] \).

Remark 3. The notion of approximation for the distribution question is based on the Skorokhod metric \cite{3}. Let us compare here this notion with two possible alternatives: the inside approximation, where \( y \) belongs to \( [D_{M,A}(\lambda - \epsilon), D_{M,A}(\lambda + \epsilon)] \), and the outside approximation, where \( y \) belongs to \( [D_{M,A}(\lambda - \epsilon), D_{M,A}(\lambda + \epsilon)] \).

The outside approximation is reasonable for \( \text{SUM} \)-automata, where the exact value of the probability is hard to compute, but for the \( \text{LimAvg} \)-automata its complexity is the same as computing the exact value (because the latter is difficult already for automata which return the same value for almost all words, as shown in Remark \cite{17}). For the inside approximation, it is the other way round: for \( \text{SUM} \)-automata it makes little sense as the problem is undecidable even for automata returning integer values, but for \( \text{LimAvg} \)-automata it is a reasonable definition as the returned values can be irrational.

We chose a definition that works for both types of automata. However, the results we present can be easily adjusted to work in the case of the outside approximation for \( \text{SUM} \)-automata and in the case of the inside approximation for the \( \text{LimAvg} \)-automata.
3 Basic properties

Consider an $f$-automaton $A$, a Markov chain $M$ and a set of words $X$. We denote by $E_M(A \mid X)$ the expected value of $A$ w.r.t. $M$ restricted only to words in the set $X$ (see [17]). The following says that we can disregard a set of words with probability 0 (e.g. containing only some of the letters under uniform distribution) while computing the expected value.

▷ Fact 4. If $\mathbb{P}(X) = 1$ then $E_M(A) = E_M(A \mid X)$.

The proof is rather straightforward; the only interesting case is when there are some words not in $X$ with infinite values. But for all the functions we consider, one can show that in this case there is a set of words with infinite value that has a non-zero probability, and therefore $E_M(A) = E_M(A \mid X) = \infty$.

One corollary of Fact 4 is that if $M$ is, for example, uniform, then because the set $Y$ of ultimately-periodic words (i.e., words of the form $vw^\omega$) is countable and hence has probability 0, we have $E_M(A) = E_M(A \mid \Sigma^\omega \setminus Y)$. This suggests that the values of ultimately-periodic words might not be representative for an automaton. We exemplify this in Remark [17] where we show an automaton whose value is irrational for almost all words, yet rational for all ultimately-periodic words.

3.1 Example of computing expected value by hand

Consider a LIMAVG-automaton $A$ and a Markov chain $M$ depicted in Figure 1. We encourage the reader to take a moment to study this automaton and try to figure out its expected value.

The idea behind $A$ is as follows. Assume that $A$ is in a state $q_i$ for some $i \in \{a, b\}$. Then, it reads a word up to the first occurrence of a subword $ba$, where it has a possibility to go to $q_x$ and then to non-deterministically choose $q_a$ or $q_b$ as the next state. Since going to $q_x$ and back to $q_i$ costs the same as staying in $q_i$, we will assume that the automaton always goes to $q_x$ in such a case. When an automaton is in the state $q_x$ and has to read a word $w = a^j b^k$, then the average cost of a run on $w$ is $\frac{1}{2}$ if the run goes to $q_a$ and $\frac{1}{2}$ otherwise. So the run with the lowest value is the one that goes to $q_a$ if $j > k$ and $q_b$ otherwise.

To compute the expected value of the automaton, we focus on the set $X$ of words $w$ such that for each positive $n \in \mathbb{N}$ there are only finitely many prefixes of $w$ of the form $w^i a^j b^k$ such that $\frac{i+j+k}{w^i|a^j|b^k} \geq \frac{1}{n}$. Notice that this means that $w$ contains infinitely many $a$ and infinitely many $b$. It can be proved in a standard manner that $\mathbb{P}_M(X) = 1$.

Let $w \in X$ be a random event, which is a word generated by $M$. Since $w$ contains infinitely many letters $a$ and $b$, it can be partitioned in the following way. Let $w = w_1 w_2 w_3 \ldots$ be a partition of $w$ such that each $w_i$ for $i > 0$ is of the form $a^j b^k$ for $j \geq 0, k > 0$, and for
We now define a run $\pi_w$ on $w$ as follows:

$$q_1^w \ldots q_i^w q_j^w \ldots q_k^w$$

where the length of each block of $q_i$ is $|w_i| - 1$, $q_0^w = q_a$ and $q_j^w = q_a$ if $w_i = a^j b^k$ for some $j > k$ and $q_i^w = q_b$ otherwise. It can be shown by a careful consideration of all possible runs that this run’s value is the infimum of values of all the runs on this word.

\[\text{Lemma 5.} \quad \text{For every } w \in X \text{ we have } L_A(w) = \text{LimAvg}(\pi_w).\]

**Proof.** We show that for every accepting run $\pi$ on $w \in X$ we have $\text{LimAvg}(\pi_w) \leq \text{LimAvg}(\pi)$. It follows that $L_A(w) = \text{LimAvg}(\pi_w)$.

Consider a run $\pi$ of $A$ on $w$. The cost of a run over $w_i = a^j b^k$ is at least $\min(j,k) - 1$, which is reached by $\pi_w$, therefore for every $i \in \mathbb{N}$ we have

$$\text{Avg}(\pi_w[i_0,i_i]) \leq \text{Avg}(\pi[0,i_i]).$$

(1)

It may happen, however, that for some $p$, the value of $\text{Avg}(\pi[0,p])$ is less than $\text{Avg}(\pi_w[0,p])$; for example, for a word starting with $baaabb$, we have $\pi_w[0,4] = q_a q_a q_b q_a$ and $\text{Avg}(\pi[0,4])$ is $\frac{1}{2}$, but for a run $\pi' = q_a q_a q_b q_a \ldots$ we have $\text{Avg}(\pi'[0,4]) = 0$. For arbitrary words, a run that never visits $q_b$ may have a better value. We show, however, that for words from $X$ this is not the case.

We show that for any position $p$ such that $s_i < p < s_{i+1}$,

$$\text{Avg}(\pi_w[0,p]) \leq \text{Avg}(\pi[0,s_i]) + \frac{p-s_i}{p}$$

(2)

By (1) and the fact that the weights of the automaton do not exceed 1, we obtain

$$\text{Avg}(\pi_w[0,s_i]) + \frac{\text{SUM}(\pi_w[0,s_i])}{p} \leq \text{Avg}(\pi[0,s_i]) + \frac{p-s_i}{p},$$

thus (2).

Assume $n \in \mathbb{N}$. By the definition of $X$, there can be only finitely many prefixes of $w$ of the form $w' a^j b^k$ where $\frac{j+k}{|w'|} \geq \frac{1}{n}$, so $\text{Avg}(\pi_w[0,p]) \geq \text{Avg}(\pi[0,s_i]) + \frac{1}{n}$ may hold only for finitely many $p$. Therefore, $\text{LimAvg}(\pi_w) \leq \text{LimAvg}(\pi) + \frac{1}{n}$ for every $n$, so $\text{LimAvg}(\pi_w) \leq \text{LimAvg}(\pi)$.

By Fact 2 and Lemma 5 it remains to compute the expected value of $\text{LimAvg}(\{\pi_w \mid w \in X\})$. As the expected value of the sum is the sum of expected values, we can state that

$$\mathbb{E}_M(\text{LimAvg}(\{\pi_w \mid w \in X\})) = \limsup_{n \to \infty} \frac{1}{s} \sum_{i=1}^{s} \mathbb{E}_M(\{(C(\pi_w)) | i \mid w \in X\})$$

It remains to compute $\mathbb{E}_M(\{C(\pi_w) | i\})$. If $i$ is large enough (and since the expected value does not depend on a finite number of values, we assume that it is), the letter $\pi_w[i]$ is in
some block $w_s = a^ib^k$. There are $j + k$ possible letters in this block, and the probability that the letter $\pi_s[i]$ is an $i$th letter in such a block is $2^{-(j+k+2)}$ ($\sim 2^n$, because the block has to be maximal, so we need to include the letters before the block and after the block). So the probability that a letter is in a block $a^ib^k$ is $\frac{j+k}{2^{j+k+2}}$. The average cost of a such a letter is $\frac{\min(j,k)}{j+k}$, as there are $j+k$ letters in this block and the block contributes $\min(j,k)$ to the sum.

It can be analytically checked that

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{j+k}{2^{j+k+2}} \cdot \frac{\min(j,k)}{j+k} = \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{\min(j,k)}{2^{j+k+2}} = \frac{1}{3}$$

We can conclude that $E_A(\text{LimAvg}(\pi_w)) = \frac{1}{3}$ and, by Lemma 5, $E_M(A) = \frac{1}{3}$.

The bottom line is that even for such a simple automaton with only one strongly connected component consisting of three states (and two of them being symmetrical), the analysis is complicated. On the other hand, we conducted a simple Monte Carlo experiment in which we computed the value of this automaton on 10000 random words of length 212 generated by $M$, and observed that the obtained values are in the interval $[0.3283, 0.3382]$, with the average of 0.33336, which is a good approximation of the expected value 0.3. This foreshadows our results for $\text{LimAvg}$-automata: we show that computing the expected value is, in general, impossible, but it is possible to approximate it with arbitrary precision. Furthermore, the small variation of the results is not accidental – we show that for strongly-connected $\text{LimAvg}$-automata, almost all words have the same value (which is equal to the expected value).

### 3.2 Irrationality of the distribution and the expected value

We show that the exact values in the probabilistic questions for $\text{SUM}$-automata and $\text{LimAvg}$-automata may be (strongly) irrational. More precisely, we show that for the $\text{SUM}$-automaton depicted in Figure 2, the distribution $D_A(-1)$ is transcendental, i.e., it is irrational and, unlike for instance $\sqrt{2}$, there is no polynomial with integer coefficients whose one of the roots is $D_A(-1)$. For the expected value, we construct an automaton $A'$ such that $E(A) - E(A') = 1 - D_A(-1)$ is transcendental. Therefore, one of $E(A)$, $E(A')$ is transcendental. Furthermore, we modify $A$ and $A'$ to show that there exists $\text{LimAvg}$-automaton $A_\infty$ whose expected value is transcendental and the value $\lambda$ such that $P\{(w \mid \mathcal{L}_{A_\infty}(w) = \lambda) = 1$ is transcendental. It follows that the minimal $\lambda$ such that $D_{A_\infty}(\lambda) = 1$ is transcendental.

**Theorem 6 (Irrational values).** The following conditions hold:

1. There exists a $\text{SUM}$-automaton whose distribution and expected value w.r.t. the uniform distribution are transcendental.
2. There exists a $\text{LimAvg}$-automaton such that the expected value and the value of almost all words w.r.t. the uniform distribution are transcendental.

**Proof.** We assume that the distribution of words is uniform. In the infinite case, this means that the Markov chain contains a single state where it loops over any letter with probability $\frac{1}{|\Sigma|}$, where $\Sigma$ is the alphabet. In the finite case, this amounts to a terminating Markov chain with one regular state and one terminating state; it loops over any letter in the non-terminating state with probability $\frac{1}{|\Sigma|+1}$ or go to the terminating state over $\epsilon$ with probability $\frac{1}{|\Sigma|+1}$. Below we omit the Markov chain as it is fixed (for a given alphabet).

We define a $\text{SUM}$-automaton $A$ (Figure 2) over the alphabet $\Sigma = \{a, \#\}$ such that $A(w) = 0$ if $w = a#a^4# \ldots #a^w$ and $A(w) \leq -1$ otherwise. Such an automaton basically
picks a block with an inconsistency and verifies it. For example, if \( w \) contains a block \(#a^n#\#a^n\#, the automaton \( A \) first assigns \(-4\) to each letter \( a \) and upon \# it switches to the mode in which it assigns \( 1 \) to each letter \( a \). Then, \( A \) returns the value \( -4 \cdot i \). Similarly, we can encode the run that returns the value \( 4 \cdot i - j \). Therefore, all the runs return \( 0 \) if and only if each block of \( a \)'s is four times as long as the previous block. Finally, \( A \) checks whether the first block of \( a \)'s has length \( 1 \) and returns \(-1 \) otherwise.

Let \( \gamma \) be the probability that a word is of the form \( a#a^4#\ldots#a^n \). Such a word has length \( l_n = 4^{n+1} + 1 + n \) and its probability is \( 3^{-(l_n+1)} \) (as the probability of any given word with \( m \) letters over a two-letters alphabet is \( 3^{-(m+1)} \)). Therefore \( \gamma \) is equal to \( \sum_{n=0}^{\infty} 3^{-(l_n+1)} \). Observe that \( \gamma \) written in base \( 3 \) has arbitrary long sequences of \( 0 \)'s and hence its representation is acyclic. Thus, \( \gamma \) is irrational.

Due to Roth’s Theorem [30], if \( \alpha \in \mathbb{R} \) is algebraic but irrational, then there are only finitely many pairs \( (p, q) \) such that \( |\alpha - \frac{p}{q}| \leq \frac{1}{q^2} \). We show that there are infinitely many such pairs for \( \gamma \) and hence it is transcendental. Consider \( i \in \mathbb{N} \) and let \( p_i, q_i \in \mathbb{N} \) be such that \( q_i = 3^{-(l_i+1)} \) and \( \frac{p_i}{q_i} = \sum_{i=0}^{i} 3^{-(l_n+1)} \). Then,
\[
0 < \gamma - \frac{p_i}{q_i} < 2 \cdot 3^{-(l_{i+1}+1)}
\]
Observe that for \( i > 1 \) we have \( l_{i+1} > 3(l_i + 1) \) and hence
\[
\gamma - \frac{p_i}{q_i} < 2 \cdot 3^{-(l_{i+1}+1)} < \frac{2}{3} 3^{-3(l_i+1)} < \frac{1}{q_i^2}.
\]
Therefore, \( \gamma \) is transcendental.

Observe that \( \gamma = 1 - D_\mathcal{A}(-1) \). Therefore, \( D_\mathcal{A}(-1) \) is transcendental. For the expected value, we construct \( \mathcal{A}' \) such that for every word \( w \) we have \( L_{\mathcal{A}'}(w) = \min(L_\mathcal{A}(w), -1) \). This can be done by adding to \( \mathcal{A} \) an additional initial state \( q_0 \), which starts an automaton that assigns to all words value \(-1\). Observe that \( \mathcal{A} \) and \( \mathcal{A}' \) differ only on words \( w \) of the form \( a#a^4#\ldots#a^n \), where \( \mathcal{A}(w) = 0 \) and \( \mathcal{A}'(w) = -1 \). On all other words, both automata return the same values. Therefore, \( E(\mathcal{A}) - E(\mathcal{A}') = \gamma \). It follows that at least one of the values \( E(\mathcal{A}), E(\mathcal{A}') \) is transcendental.

The same construction works for \( \text{LIMAVG-automata} \). We take \( \mathcal{A} \) defined as above and convert it to a \( \text{LIMAVG-automaton} \) \( \mathcal{A}_\infty \) over \( \Sigma' = \Sigma \cup \{\$\} \), where the fresh letter \$ resets the automaton, i.e., \( \mathcal{A}_\infty \) has transitions labeled by \$ from any final state of \( \mathcal{A} \) to any of its initial states. We apply the same construction to \( \mathcal{A}' \) defined as above and denote the resulting automaton by \( \mathcal{A}'_\infty \). Observe that \( E(\mathcal{A}_\infty) = E(\mathcal{A}) \) (resp., \( E(\mathcal{A}'_\infty) = E(\mathcal{A}') \)). To see that, consider random variables \( X_1, X_2, \ldots \) defined on \( \Sigma^\omega \), where \( X_i(w) \) is the average value \( \frac{1}{|w|} L_{\mathcal{A}}(u) \) of the \( i \)-th block \( w_0 w_1 \ldots \) in \( w \), i.e., \( w = w_1 w_2 \ldots w_j \ldots \), all \( w_j \) are from \( \Sigma^* \) and \( u = u_i \). Observe that \( X_1, X_2, \ldots \) are independent and identically distributed random variables and hence with probability 1 we have
\[
\lim_{s \to \infty} \frac{1}{s} (X_1 + \ldots + X_s) = \limsup_{s \to \infty} \frac{1}{s} (X_1 + \ldots + X_s) = E(X_i) = E(\mathcal{A})
\]
Therefore, with probability 1 over words \( w \) we have \( L_{\mathcal{A}_\infty}(w) = E(\mathcal{A}) \). It follows that \( E(\mathcal{A}_\infty) = E(\mathcal{A}) \) and the minimal \( \lambda \) such that \( D_{\mathcal{A}_\infty}(\lambda) = 1 \) equals \( E(\mathcal{A}) \). Similarly, \( E(\mathcal{A}'_\infty) = E(\mathcal{A}') \) and the minimal \( \lambda \) such that \( D_{\mathcal{A}'_\infty}(\lambda) = 1 \) equals \( E(\mathcal{A}') \). Therefore, for one of automata \( \mathcal{A}_\infty, \mathcal{A}'_\infty \), the value of almost all words and the expected value are transcendental. □
4 The exact value problems

In this section we consider the probabilistic questions for non-deterministic SUM-automata and LIMAVG-automata, i.e., the problems of computing the exact values of the expected value $E_M(A)$ and the distribution $D_{M,A}(\lambda)$ w.r.t. a Markov chain $M$ and an $f$-automaton $A$. The answers to these problems are related values may be irrational (Theorem 6), but one can perhaps argue that there might be some representation of irrational numbers that can be employed to avoid this problem. We prove that this is not the case by showing that computing the exact value to any representation with decidable equality of two numbers is impossible.

Theorem 7. The following conditions hold:
1. The expected value and the distribution of (non-deterministic) SUM-automata are uncomputable even for the uniform probability measure.
2. The expected value and the value of almost all words (if it exists) of (non-deterministic) LIMAVG-automata are uncomputable even for the uniform probability measure.

Proof. The proof is by a (Turing) reduction from the quantitative universality problem for SUM-automata, which is undecidable [23]:

*The quantitative universality problem for SUM-automata*: Given a SUM-automaton with weights $-1, 0$ and $1$, decide whether for all words $w$ we have $L_A(w) \leq 0$.

We first discuss reductions to the probabilistic problems for SUM-automata. Consider an instance of the quantitative universality problem, which is a SUM-automaton $A$. If there is a word $w$ with the value greater than $0$, then due to uniformity of the probability measure we have $P(w) > 0$, and thus $D_A(0) < 1$. Otherwise, clearly $D_A(0) = 1$. Therefore, solving the universality problem amounts to computing whether the $D_A(0) = 1$, and thus the latter problem is undecidable. For the expected value, we construct a SUM-automaton $A'$ such that for every word $w$ we have $L_A'(w) = \min(L_A(w), 0)$. Observe that $E(A) = E(A')$ if and only if for every word $w$ we have $L_A(w) \leq 0$, i.e., the answer to the universality problem
is YES. Therefore, there is no Turing machine, which given a SUM-automaton \( \mathcal{A} \) computes \( E(\mathcal{A}) \) (in any representation allowing for effective equality testing).

For the LIMAVG case, we construct a LIMAVG-automaton \( \mathcal{A}_\infty \) from the SUM-automaton \( \mathcal{A} \), by connecting all accepting states (of \( \mathcal{A} \)) with all initial states by transitions of weight 0 labeled by an auxiliary letter \( \# \). We construct \( \mathcal{A}_\infty' \) from \( \mathcal{A}' \) in the same way. The automata \( \mathcal{A}_\infty, \mathcal{A}_\infty' \) have been constructed from \( \mathcal{A} \) and respectively \( \mathcal{A}' \) as in the proof of Theorem 6, and the virtually the same argument shows that for almost all words \( w \) (i.e., with the probability 1) we have \( \mathcal{L}_{\mathcal{A}_\infty}(w) = \mathbb{E}(\mathcal{A}) \) (resp., \( \mathcal{L}_{\mathcal{A}_\infty'}(w) = \mathbb{E}(\mathcal{A}') \)). Therefore, \( \mathbb{E}(\mathcal{A}_\infty) = \mathbb{E}(\mathcal{A}_\infty') \) if and only if for every finite word \( u \) we have \( \mathcal{L}_\mathcal{A}(u) \leq 0 \). In consequence, there is no Turing machine computing the expected value of a given LIMAVG-automaton. Furthermore, since \( \mathcal{A}_\infty \) (resp., \( \mathcal{A}_\infty' \)) returns \( \mathbb{E}(\mathcal{A}_\infty) \) (resp., \( \mathbb{E}(\mathcal{A}_\infty') \)) on almost all words, there is no Turing machine computing the value of almost all words of a given (non-deterministic) LIMAVG-automaton.

\[ \mathcal{A}_\infty \]

4.1 Extrema automata

We discuss the distribution problem for Min-, Max-, Inf- and Sup-automata, where Min and Max return the minimal and respectively the maximal element of a finite sequence, and Inf and Sup return the minimal and respectively the maximal element of an infinite sequence. The expected value of an automaton can be easily computed based on the distribution as there are only finitely many possible values of a run (each possible value is a label of some transition).

\[ \text{Theorem 8. For Min-, Max-, Inf- and Sup-automata } \mathcal{A} \text{ and a Markov chain } \mathcal{M}, \text{ the expected value and the distribution problems can be solved in exponential time in } |\mathcal{A}| \text{ and polynomial time in } |\mathcal{M}|. \]

\[ \text{Proof.} \]

We discuss the case of \( f = \text{Inf} \) as the other cases are similar. Consider an Inf-automaton \( \mathcal{A} \). Observe that every value returned by \( \mathcal{A} \) is one of its weights. For each weight \( x \) of \( \mathcal{A} \), we construct a (non-deterministic) \( \omega \)-automaton \( \mathcal{A}_x \) that accepts only words of value greater than \( x \), i.e., \( \mathcal{L}_{\mathcal{A}_x} = \{ w \mid \mathcal{L}_\mathcal{A}(w) > x \} \). To construct \( \mathcal{A}_x \), we take \( \mathcal{A} \), remove the transitions of weight less or equal to \( x \), and drop all the weights. Therefore, the set of words with the value greater than \( x \) is regular, and hence it is measurable and we can compute its probability \( p_x \) by computing the probability of \( \mathcal{L}_{\mathcal{A}_x} \). The probability of an \( \omega \)-regular language given by a non-deterministic \( \omega \)-automaton (without acceptance conditions) can be computed in exponential time in the size of the automaton and polynomial time in the Markov chain defining the probability distribution [4 Chapter 10.3]. It follows that \( p_x \) can be computed in exponential time in \( |\mathcal{A}| \) and polynomial time in \( |\mathcal{M}| \).

Observe that \( p_x = 1 - \mathbb{P}_{\mathcal{M},\mathcal{A}}(x) \) and hence we can compute the distribution question \( \mathbb{P}_{\mathcal{M},\mathcal{A}}(\lambda) \) by computing \( 1 - p_x \) for the maximal weight \( x \) that does not exceed \( \lambda \). For the expected value, let \( x_1, \ldots, x_k \) be all weights of \( \mathcal{A} \) listed in the ascending order. Let \( p_0 = 1 \). Observe that for all \( i \in \{1, \ldots, k\} \) we have \( p_{x_{i-1}} - p_{x_i} = \mathbb{P}_{\mathcal{M}}(\{w \mid \mathcal{L}_\mathcal{A}(w) = x_i\}) \) is the probability of the set of words of the value \( x_i \). Therefore, \( \mathbb{E}_{\mathcal{M}}(\mathcal{A}) = \sum_{i=1}^{k}(p_{x_{i-1}} - p_{x_i}) \cdot x_i \) and hence the expected value can be computed in exponential time in \( |\mathcal{A}| \) and polynomial time in \( |\mathcal{M}| \).}

\[ \text{\&} \]

5 The approximation problems

We start the discussion on the approximation problems by showing a hardness result that holds for a wide range of value functions. We say that a function is 0-preserving if its value
is 0 whenever the input consists only of 0s. The functions \textsc{Sum}, \textsc{LIMAVG}, \textsc{Min}, \textsc{Max}, \textsc{Inf}, \textsc{Sup} and virtually all the functions from the literature \cite{11} are 0-preserving. The hardness result follows from the fact that accepted words have finite values, which we can force to be 0, while words without accepting runs have infinite values.

The answers in the approximation problems are numbers and to study the lower bounds, we consider their decision variants, called the separation problems. The expected separation problem is a variant of the expected value problem, in which the input is enriched with numbers \(a, b\) such that \(b - a > 2\epsilon\) and the instance is such that \(E_M(A) \notin [a, b]\) and the question is whether \(E_M(A) < a\). In the distribution separation problem, the input is enriched with numbers \(a, b, c, d\) such that \(b - a > 2\epsilon\) and \(d - c > 2\epsilon\), the instance satisfies for all \(\lambda \in [c, d]\) we have \(D_{M, A}(\lambda) \notin [a, b]\), and we ask whether \(D_{M, A}(\frac{c + d}{2}) < a\). Note that having an algorithm computing one of the approximate problems (for the distribution or the expected value), we can use it to decide the separation question. Conversely, using the separation problem as an oracle, we can perform binary search on the domain to solve the corresponding approximation problem in polynomial time.

\begin{proposition}
The following conditions hold:
1. For any 0-preserving function \(f\), the expected separation problem for non-deterministic \(f\)-automata is PSpace-hard.
2. For any 0-preserving function \(f\) over infinite words, the distribution separation problem for non-deterministic \(f\)-automata over infinite words is PSpace-hard.
\end{proposition}

\textbf{Proof.} The proof is via reduction from the universality question for non-deterministic (unweighted) finite-word automata, which is PSPACE-complete \cite{23}.

\textit{The finite-word case.} We consider the uniform distribution over finite words. Given a non-deterministic finite-word automaton \(A\), we construct a finite-word \(f\)-automaton \(A_{\text{fin}}\) by labeling all transitions of \(A\) with 0. Observe that if there exists a word which is not accepted by \(A\) then the expected value of \(A_{\text{fin}}\) is \(\infty\). Otherwise, all words have value 0 and hence the expected value for \(A_{\text{fin}}\) is 0. The universality problem for \(A\) reduces to the expected separation problem for \(A_{\text{fin}}\).

\textit{The infinite-word case.} We consider the uniform distribution over infinite words. Given a non-deterministic finite-word automaton \(A\), we construct an infinite-word \(f\)-automaton \(A_{\infty}\) in the following way. We start with the automaton \(A\). First, we extend the input alphabet with a fresh letter \#\, which resets the automaton. More precisely, we add transitions labeled by \# between any final state of \(A\) and any initial state of \(A\). Finally, we label all transitions with 0. The resulting automaton is \(A_{\infty}\).

If there exists a finite word \(u\) rejected by \(A\), then for every infinite word \(w\) containing the infix \#\(u\)# the automaton \(A_{\infty}\) has no infinite run and hence it assigns value \(\infty\) to \(w\). Observe that the set of words containing \#\(u\)# has probability 1 (for any finite word \(u\)). Therefore, if \(A\) rejects some word, the expected value for \(A_{\infty}\) is \(\infty\) and the distribution of \(A_{\infty}\) for any \(\lambda \in \mathbb{R}\) is 0. Otherwise, if \(A\) accepts all words, the expected value of \(A_{\infty}\) is 0 and the distribution of \(A_{\infty}\) for any \(\lambda \geq 0\) is 1. The universality problem for \(A\) reduces to the separation problems for \(A_{\infty}\).

\textbf{Total automata.} Theorem \ref{thm:main} gives us a general hardness result, which is due to accepting conditions rather than values returned by weighted automata. In the following, we focus on weights and we assume that weighted automata are total, i.e., they accept all words (resp., almost all words in the infinite-word case). For \textsc{Sum}-automata under the totality assumption, the approximate probabilistic questions become \#\textsc{P}-complete. We additionally show that
the approximate distribution question for \( \text{Sum}\)-automata is in \#P regardless of totality assumption.

**Theorem 10.** The following conditions hold:
1. The approximate expected value and the approximate distribution questions for total non-deterministic total \( \text{Sum}\)-automata are \#P-complete.
2. The approximate distribution question for non-deterministic \( \text{Sum}\)-automata is \#P-complete.

**Proof.** \#P-hardness. Consider the problem of counting the number of satisfying assignment of a given propositional formula \( \varphi \) in Conjunctive Normal Form (CNF) \([31, 28]\). This problem is \#P-complete. We reduce it to the problem of approximation of the expected value for total \( \text{Sum}\)-automata. Consider a formula \( \varphi \) in CNF over \( n \) variables. Let \( \mathcal{M}_T \) be a terminating Markov chain over \( \{0, 1\} \), which at each step produces 0 and 1 with equal probability \( \frac{1}{2} \), and it terminates with probability \( \frac{1}{2} \). We define a total \( \text{Sum}\)-automaton \( \mathcal{A}_\varphi \) such that it assigns 0 to all words of length different than \( n \). For words \( u \in \{0, 1\}^n \), the automaton \( \mathcal{A}_\varphi \) regards \( u \) as an assignment for variables of \( \varphi \); \( \mathcal{A}_\varphi \) non-deterministically picks one clause of \( \varphi \) and returns 1 if that clause is satisfied and 0 otherwise. We can construct such \( \mathcal{A}_\varphi \) to have polynomial size in \( |\varphi| \). Observe that \( \mathcal{A}_\varphi(u) = 0 \) if some clause of \( \varphi \) is not satisfied by \( u \), i.e., \( \varphi \) is false under the assignment given by \( u \). Otherwise, if the assignment given by \( u \) satisfies \( \varphi \), then \( \mathcal{A}_\varphi(u) = 1 \). It follows that the expected value of \( \mathcal{A}_\varphi \) equals \( 3^{-n} \cdot C \), where \( 3^{-n} \) is the probability of generating a word of length \( n \) and \( C \) is the number of variable assignments satisfying \( \varphi \). Therefore, we can compute \( C \) by computing the expected value of \( \mathcal{A}_\varphi \) with any \( \epsilon \) less than \( 0.5 \cdot 3^{-(n+1)} \). Observe that the automaton \( \mathcal{A}_\varphi \) returns values 0 and 1 and hence the expected value \( E_{\mathcal{M}}(\mathcal{A}_\varphi) = 1 - D_{\mathcal{M}, \mathcal{A}_\varphi}(0) \), where \( 1 - D_{\mathcal{M}, \mathcal{A}_\varphi}(0) \) is the probability that \( \mathcal{A}_\varphi \) returns 1.

**Containment of the approximate distribution question in \#P.** Consider a terminating Markov chain \( \mathcal{M}_T \), a \( \text{Sum}\)-automaton \( \mathcal{A} \), and \( \epsilon \in \mathbb{Q}^+ \).

Let \( C \) be the smallest number such that every non-zero probability in \( \mathcal{M}_T \) is at least \( 2^{-C} \). Such \( C \) is polynomial in the input size. Consider \( N = C \cdot \text{len}(\epsilon) + 1 \) and let \( D_{\mathcal{M}_T, \mathcal{A}}(\lambda, N) \) be the distribution of \( \mathcal{A} \) over words up to length \( N \), i.e., \( \mathbb{P}_{\mathcal{M}_T}(\{w \mid |w| \leq N \land \mathcal{A}(w) = \lambda\}) \).

We show that the distribution of \( \mathcal{A} \) and the distribution of \( \mathcal{A} \) over words up to length \( N \) differ by less than \( \frac{\epsilon}{2} \), i.e.,

\[
\left| D_{\mathcal{M}_T, \mathcal{A}}(\lambda, n) - D_{\mathcal{M}_T, \mathcal{A}}(\lambda, n) \right| \leq \frac{\epsilon}{2}.
\]

To do so, let \( p_n \), for \( n \in \mathbb{N} \), be the probability that \( \mathcal{M}_T \) emits a word of the length greater than \( n \). From any state of \( \mathcal{M}_T \), the probability of moving to a terminating state is at least \( 2^{-C} \). We can (very roughly) bound the probability of generating a word of length greater than \( n \) by \( (1 - 2^{-C})^n \). This means that \( p_n \) decreases exponentially with \( n \). Since \( (1 - \frac{1}{2})^n \leq \frac{1}{2} \) for all \( n > 1 \), we obtained the desired inequality.

Let \( K = (N + 1) \cdot \log(|\Sigma|) \cdot \epsilon^{-1} + 1 \). We build a non-deterministic Turing machine \( H_1 \) such that on the input \( \mathcal{M}_T \), \( \mathcal{A} \), \( \epsilon \), and \( \lambda \) such that the number \( c_A \) of accepting computations of \( H_1 \) satisfies the following:

\[
\left| D_{\mathcal{M}_T, \mathcal{A}}(\lambda, n) - \frac{c_A}{2^K} \right| \leq \frac{\epsilon}{2}.
\]

To \( \epsilon \)-approximate \( D_{\mathcal{M}_T, \mathcal{A}}(\lambda) \), we need to compute \( c_A \) and divide it by \( 2^K \), which can be done in polynomial time.
The machine $H_1$ works as follows. Given the input $M^T$, $A$, $\epsilon$, it non-deterministically generates a string $ua$, where $u \in (\Sigma \cup \{\#\})^N$ is a word and $\alpha \in \{0, 1\}^K$ is a number written in binary. The machine rejects unless $u$ is of the form $ww$, where $w \in \Sigma^*$ and $v \in \{\#\}^*$. Then, the machine accepts if $L_A(w) \leq \lambda$ and $\alpha \leq 2^K \cdot P_{M^T}(w)$. Therefore, provided that $H_1$ generates $w$ with $L_A(w) \leq \lambda$, the number of accepting computations $c_A^w$ equals $[2^K \cdot P_{M^T}(w)]$. It follows that $c_A^w$ divided by $2^K$ is a $2^{-K}$-approximation of $P_{M^T}(w)$, i.e.,

$$P_{M^T}(w) - \frac{c_A^w}{2^K} < 2^{-K}.$$ 

The total number of accepting paths of $H_1$ is given by

$$c_A = \sum_{w: |w| \leq N \wedge L_A(w) \leq \lambda} c_A^w.$$ 

We estimate the difference between $D_{M^T,A}(\lambda, n)$ and the value $\frac{c_A}{2^K}$:

$$|D_{M^T,A}(\lambda, n) - \frac{c_A}{2^K}| \leq \sum_{w: |w| \leq N \wedge L_A(w) \leq \lambda} |P_{M^T}(w) - \frac{c_A^w}{2^K}| \leq |\Sigma|^{N+1} \cdot 2^{-K} < \frac{\epsilon}{2}.$$ 

**Containment of the approximate expected value question in $\#P$.** Assume that $A$ is total. For readability we assume that $A$ has only integer weights. If it does not, we can multiply all weights by least common multiple of all denominators of weights in $A$; this operation multiples the expected value by the same factor.

Recall that $C$ is the smallest number such that every non-zero probability in $M^T$ is at least $2^{-C}$. Let $W$ be the maximal absolute value of weights in $A$ and let $M = C \ln(\epsilon) \cdot \log(CW) + 1$ and $E_{M^T}(A, N)$ be the expected value of $M^T$ for words up to length $M$, i.e., computing only the finite sum from the definition of the expected value. We show that

$$|E_{M^T}(A) - E_{M^T}(A, M)| \leq \frac{\epsilon}{2}.$$ 

Recall that $p_n$ is the probability that $M^T$ emits a word of the length greater than $n$, and $p_n \leq (1 - 2^{-C})^n$. Since $A$ is total, the value of every word $w$ is finite and it belongs to the interval $[-|w| \cdot W, |w| \cdot W]$. The value of a word of the length bounded by $i$ is at most $i \cdot W$. Therefore, the expected value of $A$ over words of grater than $k$ is bounded from above by

$$\sum_{i > k} p_i \cdot i \cdot W \leq W \cdot (1 - 2^{-C})^k \cdot (k + 1).$$

W.L.O.G. we assume, that there are no transitions to the initial state in $A$. Next, we transform $A$ to an automaton $A'$ that returns natural numbers on all words of length at most $M$ by adding $W \cdot M$ to every transition from the initial state. Observe that $E_{A'} = E_A + W \cdot M$ and $D = 2 \cdot W \cdot M$ is an upper bound on values returned by the automaton $A'$ on words of length at most $M$.

Finally, we construct a Turing machine $H_2$, similar to $H_1$. Let $K = (D + 1) \cdot (N + 1) \cdot (|\Sigma| + 1) \cdot \epsilon^{-1} + 1$. $H_2$ non-deterministically chooses a word $ua$, where $u \in (\Sigma \cup \{\#\})^N$ is a word and $\alpha \in \{0, 1\}^K$ is a number written in binary, and also non-deterministically picks a natural number $\beta \in [0, D]$. The machine rejects unless $u$ is of the form $ww$, where $w \in \Sigma^*$ and $v \in \{\#\}^*$. Then $H_2$ accepts if and only if $L_A(w) \leq \beta$ and $\alpha \leq 2^K \cdot P_{M^T}(w)$. Then, provided that $H_2$ generates $w$, the number of accepting computations $c_A^w$ equals $[2^K \cdot P_{M^T}(w) \cdot L_A(w)]$. Therefore, using estimates similar to the distribution case, we obtain the desired inequality

$$|E_{M^T}(A', M) - \frac{c_A}{2^K}| \leq \frac{\epsilon}{2}.$$
Finally, we obtain that $\frac{c_A}{2K} - WM$ is an $\epsilon$-approximation of $E_{MT}(A)$, i.e.,

$$|E_{MT}(A) - \left(\frac{c_A}{2K} - WM\right)| \leq \epsilon.$$  

We show that the approximation problem for LimAvg-automata is PSPACE-hard over the class of total automata.

**Theorem 11.** The separation problems for non-deterministic total LimAvg-automata are PSPACE-hard.

**Proof.** We consider the uniform distribution over infinite words. Given a non-deterministic finite-word automaton $A$, we construct an infinite-word LimAvg-automaton $A_\infty$ from $A$ in the following way. We introduce an auxiliary symbol $#$ and we add transitions labeled by $#$ between any final state of $A$ and any initial state of $A$. Then, we label all transitions of $A_\infty$ with 0. Finally, we connect all non-accepting states of $A$ with an auxiliary state $q_{sink}$, which is a sink state with all transitions of weight 1. The automaton $A_\infty$ is total.

Observe that if $A$ is universal, then $A_\infty$ has a run of value 0 on every word. Otherwise, if $A$ rejects a word $w$, then upon reading a subword $#w#$, the automaton $A_\infty$ reaches $q_{sink}$, i.e., the value of the whole word is 1. Almost all words contain an infix $#w#$ and hence almost all words have value 1. Therefore, the universality problem for $A$ reduces to the problem deciding whether for almost all words $w$ we have $L_{A_\infty}(w) = 0$ or for almost all words $w$ we have $L_{A_\infty}(w) = 1$? The latter problem reduces to the expected separation problem as well as the distribution separation problem for $A_\infty$.

6 Approximating LimAvg-automata in exponential time

In this section we develop algorithms for the approximate expected value and approximate distribution questions for (non-deterministic) LimAvg-automata. The presented algorithms work in exponential time in the size of the automaton, polynomial time in the size of the Markov chain and the precision.

The case of LimAvg-automata is significantly more complex than the other cases and hence we present the algorithms in stages. First, we restrict our attention to recurrent LimAvg-automata and the uniform distribution over infinite words. Recurrent automata are strongly connected with an appropriate set of initial states. We show that deterministic LimAvg-automata with bounded look-ahead approximate recurrent automata. Next, in Section 6.4 we extend this result to non-uniform measures given by Markov chains. Finally, in Section 7 we show the approximation algorithms for all (non-deterministic) LimAvg-automata and measures given by Markov chains.

**Recurrent automata.** Let $A = (\Sigma, Q, Q_0, \delta)$ be a non-deterministic LimAvg-automaton and $\hat{\delta}$ be the extension of $\delta$ to all words $\Sigma^*$. The automaton $A$ is recurrent if and only if the following conditions hold:

1. for every state $q \in Q$ there is a finite word $u$ such that $\hat{\delta}(q, u) = Q_0$ ($\hat{\delta}$ is the transition relation extended to words), and
2. for every set $S \subseteq Q$, if $\hat{\delta}(Q_0, w) = S$ for some word $w$, then there is a finite word $u$ such that $\hat{\delta}(S, u) = Q_0$.

Intuitively, in recurrent automata $A$, if two runs deviate at some point, with high probability it is possible to synchronize them. More precisely, for almost all words $w$, if $\pi$ is a run on $w$, and $\rho$ is a finite run up to position $i$, then $\rho$ can be extended to an infinite
run that eventually coincides with \( \pi \). Moreover, we show that with high probability, they synchronize within doubly-exponential number of steps in \(|A|\) (Lemma 14).

**Example 12.** Consider the automaton depicted in Figure 1. This automaton is recurrent with the initial set of states \( Q_0 = \{q_x, q_a, q_b\} \). For condition (1) from the definition of recurrent automata, observe that for every state \( q \) we have \( \hat{\delta}(q, a,b) = Q_0 \). For condition (2), observe that \( \hat{\delta}(Q_0, b) = Q_0, \hat{\delta}(Q_0, a) = \{q_a, q_b\} \) and \( \hat{\delta}(\{q_a, q_b\}, a) = \hat{\delta}(\{q_a, q_b\}, b) = Q_0 \). The automaton would also be recurrent in the case of \( Q_0 = \{q_a, q_b\} \), but not in any other case.

Consider an automaton \( A \) depicted below:

![Automaton Diagram]

The automaton \( A \) is recurrent if the set of initial states is either \( \{q_L\} \) or \( \{q_R\} \), but not in the case of \( \{q_L, q_R\} \). Indeed, if we pick \( q_L \) (resp., \( q_R \)) we can never reach the whole set \( \{q_L, q_R\} \). This realizes our intuition that runs that start in \( q_L \) and \( q_R \) will never synchronize.

We discuss properties of recurrent automata. For every \( A \) that is strongly connected as a graph there exists a set of initial states \( T \) with which it becomes recurrent. Indeed, consider \( A \) as an unweighted \( \omega \)-automaton and construct a deterministic \( \omega \)-automaton \( A^D \) through the power-set construction applied to \( A \). Observe that \( A^D \) has a single bottom strongly-connected component (BSCC), i.e., a strongly connected component such that there are no transitions leaving that component. The set \( Q_0 \) belongs to that BSCC. Conversely, for any strongly connected automaton \( A \), if \( Q_0 \) belongs to the BSCC of \( A^D \), then \( A \) is recurrent.

Observe that for a recurrent automaton \( A \) the probability of words accepted by \( A \) is either 0 or 1. Now, for a word \( w \) consider a sequence of reachable sets of states \( \Pi_w \) defined as \( Q_0, \hat{\delta}(Q_0, w[1]), \hat{\delta}(Q_0, w[2]), \ldots \). Since \( A^D \) has a single BSCC containing \( Q_0 \), all sets of \( \Pi_w \) belong to that BSCC and hence either for almost all words \( w \), the sequence \( \Pi_w \) eventually contains only empty sets or for all words \( w \), the sequence \( \Pi_w \) consists of non-empty sets only. Observe that \( A \) has an infinite run on \( w \) if and only if \( \Pi_w \) consists of non-empty sets. It follows that the probability of the set of words having any infinite run in \( A \) is either 0 or 1.

While Markov chains generate words letter by letter, to define a run of a word of the minimal value we need to have the completely generated word, i.e., the optimal transition at some position \( i \) may depend on some positions \( j > i \) in the word. This precludes application of standard techniques for probabilistic verification, which rely on the fact that the word and the run on it are generated simultaneously \[33, 15, 4\].

**Key ideas.** Our main idea is to change the non-determinism to bounded look-ahead. This must be inaccurate, as the expected value of a deterministic automaton with bounded look-ahead is always rational, whereas Theorem 6 shows that the values of non-deterministic automata may be irrational. Nevertheless, we show that bounded look-ahead is sufficient to approximate the probabilistic questions for recurrent automata (Lemma 15). Furthermore, the approximation can be done effectively (Lemma 19), which in turn gives us an exponential-time approximation algorithm for recurrent automata (Lemma 21). Then, we comment on the extension to all distributions given by Markov chains (Section 6.4). Finally, we show the proof for all LIMAVG-automata over probability measures given by Markov chains (Theorem 22).
6.1 Nearly-deterministic approximations

Jumping runs. Let \( k > 0 \) and let \( N_k \) be the set of natural numbers not divisible by \( k \). A \( k \)-jumping run \( \xi \) of \( \mathcal{A} \) on a word \( w \) is an infinite sequence of states such that for every position \( i \in N_k \) we have \((\pi[i-1], w[i], \pi[i]) \in \delta\).

An \( i \)-th block of a \( k \)-jumping run is a sequence \( \xi[k, k(i+1) - 1] \); within a block the sequence \( \xi \) is consistent with transitions of \( \mathcal{A} \). The positions \( k, 2k, \ldots \notin N_k \) are jump positions, where the sequence \( \xi \) need not obey the transition relation of \( \mathcal{A} \).

The cost \( C \) of a transition of a \( k \)-jumping run \( \xi \) within a block is defined as usual, while the cost of a jump is defined as the minimal weight of \( \mathcal{A} \). The value of a \( k \)-jumping run \( \xi \) is defined as the limit average computed for such costs.

Optimal and block-deterministic jumping runs. We say that a \( k \)-jumping run \( \xi \) on a word \( w \) is optimal if its value is the infimum over values of all \( k \)-jumping runs on \( w \). We show that optimal \( k \)-jumping runs can be constructed nearly deterministically, i.e., only looking ahead to see the whole current block.

For every \( S \subseteq Q \) and \( u \in \Sigma^k \) we fix a run \( \xi_{S,u} \) on \( u \) starting in one of states of \( S \), which has the minimal average weight. Then, given a word \( w \in \Sigma^\omega \), we define a \( k \)-jumping run \( \xi \) as follows. We divide \( w \) into \( k \)-letter blocks \( u_1, u_2, \ldots \) and we put \( \xi = \xi_{S_0,u_1} \xi_{S_1,u_2} \cdots \), where \( S_0 = \{ q_0 \} \) and for \( i > 0 \), \( S_i \) is the set of states reachable from \( q_0 \) on the word \( u_1 \ldots u_i \). The run \( \xi \) is a \( k \)-jumping run and it is indeed optimal. We call such runs block-deterministic — they can be constructed based on finite memory — the set of reachable states \( S_i \) and the current block of the input word.

Since all runs of \( \mathcal{A} \) are in particular \( k \)-jumping runs, the value of (any) optimal \( k \)-jumping run on \( w \) is less or equal to \( \mathcal{A}(w) \). We show that for recurrent \( \text{LMIAVG-automata} \), the values of \( k \)-jumping runs on \( w \) converge to \( \mathcal{A}(w) \) as \( k \) tends to infinity. To achieve this, we construct a run of \( \mathcal{A} \) which tries to “follow” a given jumping run, i.e., after almost all jump positions it is able to synchronize with the jumping run quickly.

**Proof plan.** Let \( k > 0 \). Consider a word \( w \) and some optimal \( k \)-jumping run \( \xi_o \) on \( w \). We construct a run \( \pi_f \) of \( \mathcal{A} \) in the following way. Initially, both runs start in some initial state \( q_0 \) and coincide. However, at the first jump position \( \xi_o \), may take a move that is not a transition of \( \mathcal{A} \). The run \( \pi_f \) attempts to synchronize with \( \xi_o \), i.e., to be at the same position in the same state, and then repeat transitions of \( \xi_o \) until the end of the block. Then, in the next block, regardless of whether \( \pi_f \) managed to synchronize with \( \xi_o \) or not, we repeat the process. We say that a run \( \pi_f \) constructed in such a way is a run following \( \xi_o \).

In the following Lemma [13], we show that for \( m \in \mathbb{N} \) large enough, with high probability, the run \( \pi_f \) synchronizes with \( \xi_o \) within \( m \) steps. We then show that if \( m \) is large enough and \( k \) is much larger than \( m \), then the values of runs \( \pi_f \) and \( \xi_o \) differ by less than \( \epsilon \) (Lemma [15]).

Let \( q \) be a state of \( \mathcal{A} \) and \( u \) be a finite word. We say that a word \( v \) saturates the pair \((q, u)\), if the set of reachable states from \( q \) over \( v \) equals all the states reachable over \( uv \) from the initial states, i.e., \( \delta(Q_0, uv) = \delta(q, v) \).

**Example 13.** Consider the automaton from Figure [1] with \( Q_0 = Q \). For any \((q, u)\), any word that contains the infix \( abab \) saturates \((q, u)\), as \( \delta(Q_0, uv'abab) = \delta(q, v'abab) = Q_0 \) for any \( v' \).

Observe that in the above, the probability that a random word of a length \( 4\ell \) does not saturate \((q, u)\) is bounded by \( (1 - \frac{1}{m})^\ell \). So the probability that a random word \( v \) saturates \((q, u)\) quickly tends to 1 with \( |v| \). The next lemma shows that this is not a coincidence.
Lemma 14. Let $A$ be an NFA, $u$ be a finite word, and $q \in \delta(Q_0, u)$. For every $\Delta > 0$ there exists a natural number $\ell = 2^{2^{O(|A|)}} \log(\frac{1}{\Delta})$ such that over the uniform distribution on $\Sigma^\ell$ we have $P\{v \in \Sigma^\ell \mid v \text{ saturates } (q, u)\} \geq 1 - \Delta$.

Proof. First, observe that there exists a word $v$ saturating $(q, u)$. Let $S = \delta(Q_0, u)$. Then, $q \in S$. Since $A$ is recurrent, there exists a word $\alpha$ such that $Q_0 = \delta(q, \alpha)$. It follows that $S = \delta(q, \alpha u)$. Since $q \in S$, we have $\delta(q, \alpha u) \subseteq \delta(S, \alpha u)$. It follows that for $i \geq 0$ we have $\delta(q, (\alpha u)_i) = \delta(q, (\alpha u)^{i+1}) \subseteq \delta(S, (\alpha u)^{i+1})$. Therefore, for some $i > 0$ we have $\delta(q, (\alpha u)_i) = \hat{\delta}(S, (\alpha u)^i)$, i.e., the word $(\alpha u)^i$ saturates $(q, u)$.

Now, we observe that there exists a saturating word that is exponentially bounded in $|A|$. We start with the word $v_0$ equal $(\alpha u)^i$ and we pick any two positions $k < \ell$ such that $\delta(q, v_0[1, k]) = \delta(q, v_0[1, \ell])$ and $\hat{\delta}(S, v_0[1, k]) = \delta(S, v_0[1, \ell])$. Observe that for $v_1$ obtained from $v_0$ by removal of $v[k+1, \ell]$, the reachable sets do not change, i.e., $\delta(q, v_0) = \delta(q, v_1)$ and $\hat{\delta}(S, v_0) = \hat{\delta}(S, v_1)$. We iterate this process until there are no such positions. The resulting word $v'$ satisfies $\delta(S, v') = \delta(q, v')$. Finally, each position $k$ of $v'$ defines the unique pair $(\delta(q, v_0[1, k]), \hat{\delta}(S, v_0[1, k]))$ of subsets of $Q$. Therefore, the length of $v'$ is bounded by $2^{2^{O(|A|)}}$.

We have shown above that for every pair $(q, u)$ there exists a saturating word $v_{q,u}$ of length bounded by $N = 2^{2^{O(|A|)}}$. The probability of the word $v_{q,u}$ is $p_0 = 2^{-O(N)}$. Let $\ell = \frac{1}{p_0} \cdot \log(\frac{1}{\Delta})$; we show that the probability that $(q, u)$ is not saturated by a word from $\Sigma^\ell$ is at most $\Delta$. Consider a word $x \in \Sigma^\ell$. We can write it as $x = x_1 \ldots x_r$, where all words $x_k$ have length $N$. If $x_k$ saturates $(q, u x_1 \ldots x_{k-1})$, then $x_1 \ldots x_k$ (as well as $x$) saturates $(q, u)$. Therefore, the word $x$ does not saturate $(q, u)$ if and only if all $1 \leq k \leq \ell$, $x_k$ does not saturate $(q, u x_1 \ldots x_{k-1})$. The probability that $x \in \Sigma^\ell$ does not saturate $(q, u)$ is at most $\left(1 - p_0\right)^\ell \leq \left(\frac{1}{2}\right)^{\log(1/\Delta)} \leq \Delta$.

Finally, we show that for almost all words the value of an optimal $k$-jumping run approximates the values of the word.

Lemma 15. Let $A$ be a recurrent $\text{LIMAVG-automaton}$. For every $\epsilon \in \mathbb{Q}^+$, there exists $k$ such that for almost all words $w$, the value $A(w)$ and the value of an optimal $k$-jumping run on $w$ differ by at most $\epsilon$. The value $k$ is doubly-exponential in $|A|$ and polynomial in $\frac{1}{\epsilon}$.

Proof. By Lemma 14 for all $\Delta > 0$, $\ell = 2^{2^{O(|A|)}} \log(\frac{1}{\Delta})$, and all $k > \ell$, the probability that the run $\pi_f$ synchronizes with an optimal $k$-jumping run $\xi_o$ within $\ell$ steps in a block is at least $1 - \Delta$.

Consider some $k > \ell$ and an optimal $k$-jumping run $\xi_o$ that is block-deterministic. Observe that the run $\pi_f$ of $A$ following $\xi_o$ is also block-deterministic.

Consider a single block $\xi_o[i, i + k - 1]$. By Lemma 14, the probability that $\pi_f[i + \ell - 1] = \xi_o[i + \ell - 1]$ is at least $1 - \Delta$. In such a case, the sum of costs on that block of $\pi_f$ exceeds $\xi_o$ by at most $D \cdot \ell$, where $D$ is the difference between the maximal and the minimal weight in $A$. Otherwise, if $\pi_f$ does not synchronize, we bound the difference of the sums of values on that block by the maximal possible difference $D \cdot k$.

Since runs are block-deterministic, synchronization of $\pi_f$ and $\xi_o$ satisfies the Markov property; it depends only on the current block and the set of states $S$ reachable on the input word until the beginning of the current block. We observe that as $A$ is recurrent, the corresponding Markov chain, whose states are reachable sets of states $S$ of $A$, has only a single BSCC. Therefore, for almost all words, the average ratio of $k$-element blocks, in which $\pi_f$ synchronizes with $\xi_o$ within $\ell$ steps, is $1 - \Delta$. We then conclude that for almost all words the difference between $\pi_f$ and $\xi_o$ is bounded by $\gamma = \frac{(1-\Delta)D \cdot k + \Delta (D \cdot k)}{k}$. Observe that with $\Delta < \frac{\epsilon}{2D}$ and $k > \frac{2D \cdot \ell}{\epsilon}$, the value $\gamma$ is less than $\epsilon$. 

\begin{table}
Let $A$ be a recurrent LIMAVG-automaton and $k > 0$. (1) The functions $g[k]$ and $L_A$ are random variables. (2) For almost all words $w$ we have $g[k](w) = E(g[k])$ and $L_A(w) = E(L_A)$.

**Proof.** Since $A$ is recurrent, $M[k]$ has a single BSCC and hence $M[k]$ and $g[k]$ return the same value for almost all words [18]. This implies that the preimage through $g[k]$ of each set has measure 0 or 1, and hence $g[k]$ is measurable [17]. Lemma 15 implies that (measurable functions) $g[k]$ converge to $L_A$ with probability 1, and hence $L_A$ is measurable [17]. As the limit of $g[k]$, $L_A$ also has the same value for almost all words.

**Remark 17.** The automaton $A$ from the proof of Theorem 6 is recurrent (it resets after each $\epsilon$), so the value of $A$ on almost all words is irrational. Yet, for every ultimately periodic word $vw\omega$, the value of $A$ is rational. This means that while the expected value is realized by almost all words, it is not realized by any ultimately periodic word.

### 6.3 Approximation algorithms

We show that the expected value of $g[k]$ can be efficiently approximated. The approximation is exponential in the size of $A$, but only logarithmic in $k$ (which is doubly-exponential due to Lemma 15).

To approximate the expected value of $g[k]$ we need to compute the expected value of $A$ over $k$-letter blocks. Such blocks are finite and hence we consider $A$ as a finite-word automaton with the average value function $\text{Avg}$. More precisely, for $S$ being a subset of states of $A$, we define $A^\text{fin}_S$ as a $\text{Avg}$-automaton over finite words as $A$, which initial states set to $S$ and all states accepting. We can approximate the expected value of $A^\text{fin}_S$ over words $\Sigma^k$ in logarithmic time in $k$.

**Lemma 18.** Let $A$ be a recurrent LIMAVG-automaton, let $S, S'$ be subsets of states of $A$, and let $i > 0$. We can approximate the expected value $\mathbb{E}(\{A^\text{fin}_S(w) \mid |w| = 2^h \text{ and } \hat{\delta}(S, w) = S'\})$ within a given $\epsilon \in \mathbb{Q}^+$ in exponential time in $|A|$ and polynomial time in $k$ and $\frac{1}{\epsilon}$.

**Proof.** Let $h(q, w, q')$ be the infimum average weight over runs from $q$ to $q'$ over $w$. Consider $\epsilon_0 = \frac{1}{2\epsilon}$. Let $H = \{j : \epsilon_0 \mid j \in \mathbb{Z}\} \cap (-|A|, |A|)$ be a finite set and let $[x]_{\epsilon_0}$ stand for the greatest number from $H$ not exceeding $x$. 

Consider $i \in \{0, \ldots, k\}$ and let $N = 2^i$. We define a function $\tilde{h}^i : Q \times \Sigma^N \times Q \to H$ as follows. First, we define $\tilde{h}^0(q, w, q') = |h(q, w, q')|_{\epsilon_0}$. Then, inductively, we define

$$
\tilde{h}^{i+1}(q, w_1 w_2, q') = \left[ \min_{q'' \in Q} \frac{\tilde{h}^i(q, w_1, q'') + \tilde{h}^i(q'', w_2, q')}{2} \right]_{\epsilon_0}
$$

We show by induction on $i$ that for all $i$, $q$, $q'$, $N = 2^i$ and $w \in \Sigma^N$ we have $|h(q, w, q') - \tilde{h}^i(q, w, q')| \leq (i + 1)\epsilon_0$. First, we comment on the deteriorating precision. Notice that $|h(q, w, q') - \tilde{h}^i(q, w, q')| \leq \epsilon_0$ may not hold in general. Let us illustrate this with a simple toy example. Consider $\epsilon_0 = 1$, $x \in (0, 1)$ and $y \in (1, 2)$. Then $\frac{\epsilon_0 + \epsilon}{2} \in (\frac{1}{2}, \frac{3}{2})$, thus $\frac{\epsilon_0 + \epsilon}{2} \in \{0, 1\}$. However, knowing only $[x]_{\epsilon_0}$ and $[y]_{\epsilon_0}$, we cannot assess whether the answer should be 0 or 1. Therefore, when iterating the above-described procedure, we may lose some precision (up to one $\epsilon_0$ at each step); this is why we start with $\epsilon_0$ rather than $\epsilon$.

Now, we show by induction $|h(q, w, q') - \tilde{h}^i(q, w, q')| \leq (i + 1)\epsilon_0$. More precisely, we show that

1. $\tilde{h}^i(q, w, q') \leq h(q, w, q')$ and
2. $h(q, w, q') - \tilde{h}^i(q, w, q') \leq (i + 1)\epsilon_0$.

The case $i = 0$ follows from the definition of $\tilde{h}^0(q, w, q')$. Consider $i > 0$ and assume that for all words $w$ of length $2^i$ the induction hypothesis holds. Consider $w = w_1 w_2$ and states $q, q'$. There exists $q''$ such that $h(q, w, q') = h(q, w_1, q'') + h(q'', w_1, q')$. Then, due to induction assumption of (1) we have $h^{i-1}(q, w_1, q'') \leq h(q, w_1, q'')$ and $h^{i-1}(q'', w_1, q') \leq h(q'', w_1, q')$. In consequence, we get (1).

Now, to show (2), consider a state $s$ that realizes the minimum from the definition of $\tilde{h}^i(q, w, q')$. There are numbers $a, b \in \mathbb{Z}$ such that $\tilde{h}^{i-1}(q, w_1, s) = a\epsilon_0$, and $\tilde{h}^{i-1}(s, w_2, q') = b\epsilon_0$.

Then, $h(q, w, q') \leq \frac{h(q, w_1, s) + \epsilon_0}{2} + \frac{h(s, w_2, q) - \epsilon_0}{2}$ and we have

$$
\frac{h(q, w_1, s) + \epsilon_0}{2} + \frac{h(s, w_2, q) - \epsilon_0}{2} = \frac{h(q, w_1, s) + \epsilon_0}{2} + \epsilon_0
$$

Observe that $\frac{a\epsilon_0 - \epsilon_0}{2} = \frac{(a+b)\epsilon_0}{2}$ if $a + b$ is even and $\frac{a\epsilon_0 - \epsilon_0}{2} = \frac{(a+b)\epsilon_0}{2} - \frac{\epsilon_0}{2}$ otherwise. This gives us the following inequality

$$
\frac{h(q, w, q') - \tilde{h}^i(q, w, q')}{2} \leq \frac{h(q, w_1, s) - a\epsilon_0}{2} + \frac{h(s, w_2, q) - b\epsilon_0}{2} + \epsilon_0
$$

Due to the induction hypothesis (2) we have $h(q, w_1, q'') - a\epsilon_0 \leq \epsilon_0 h(q'', w_2, q') - b\epsilon_0 \leq \epsilon_0$ and it gives us (2).

We cannot compute the functions $\tilde{h}^i$ directly (in reasonable time), because there are too many words to be considered. However, we can compute them symbolically. Define the clustering function $c^i$ as follows. Let $N = 2^i$. For each function $f : Q \times \Sigma^N \times Q \to H$ we define $c^i(f) = \{ (w) \mid \forall q, q'.\tilde{h}^i(q, w, q') = f(q, w, q') \}$. Basically, for each function $f$, clustering counts the number of words realizing $f$ though functions $\tilde{h}^i(\cdot, w, \cdot)$.

The function $c^i$ can be computed directly. Then, $c^{i+1}(f) = \{ (f_1, f_2) \mid f = f_1 \ast f_2, h_1 \ast h_2(q, q'') = \left[ \min_{q'' \in F_2} \frac{h_1(q, q'') + h_2(q'', q'')}{2} \right]_{\epsilon_0} \}$. It follows that we can compute the $k$-clustering in time exponential in $|\mathcal{A}|$, polynomial in $\frac{1}{\epsilon}$ and $k$. The desired expected valued can be derived from the $k$-clustering in the straightforward way.
In consequence, we can approximate the expected value of $g[k]$ in exponential time in $|A|$ but logarithmic in $k$, which is important as $k$ may be doubly-exponential in $|A|$ (Lemma 15).

**Lemma 19.** Given a recurrent LIMAVG-automaton $A$, $k = 2^i$ and $\epsilon \in \mathbb{Q}^+$, the expected value $E(g[k])$ can be approximated up to $\epsilon$ in exponential time in $|A|$, logarithmic time in $k$ and polynomial time in $\frac{1}{\epsilon}$.

**Proof.** Recall that the expected values of $M[k]$ and $g[k]$ coincide. Observe that $M[k]$ can be turned into a weighted Markov chain $N[k]$ over the same set of states with one edge between any pair of states as follows. For an edge $(S,S')$, we define its probability as $\frac{1}{|\Sigma|}$ multiplied by the number of edges from $S$ to $S'$ with positive probability in $M[k]$ and the weight of $(S,S')$ in $N[k]$ is the average of the weights of all such the edges in $M[k]$, i.e., the weight of $(S,S')$ is $E((M_{S,S'}^k(w) \mid w \in \Sigma^k \text{ and } \delta(S,w) = S'))$ (see Lemma 18). Observe that the expected values of $M[k]$ and $N[k]$ coincide.

Having the Markov chain $N[k]$, we can compute its expected value in polynomial time [18]. Since $N[k]$ has the exponential size in $|A|$, we can compute it in exponential time in $|A|$. However, we need to show how to construct $N[k]$. In particular, computing $E((A_{S'}^k(w) \mid w \in \Sigma^k \text{ and } \delta(S,w) = S'))$ can be computationally expensive as $k$ can be doubly-exponential in $|A|$ (Lemma 15). Still, due to Lemma 18, we can approximate $E((A_{S'}^k(w) \mid w \in \Sigma^k \text{ and } \delta(S,w) = S'))$ in exponential time in $|A|$, logarithmic time in $k$ and polynomial time in $\epsilon$.

Therefore, we can compute a Markov chain $M^\infty$ with the same structure as $N[k]$ and such that for every edge $(S,S')$ the weight of $(S,S')$ in $M^\infty$ differs from the weight in $N[k]$ by at most $\epsilon$. Therefore, the expected values of $M^\infty$ and $N[k]$ differ by at most $\epsilon$.

Lemma 15 and Lemma 19 give us approximation algorithms for the expected value and the distribution of recurrent automata over the uniform distribution:

**Lemma 20.** Given a recurrent LIMAVG-automaton $A$, $\epsilon \in \mathbb{Q}^+$ and $\lambda \in \mathbb{Q}$, we can compute $\epsilon$-approximations of the distribution $D_A(\lambda)$ and the expected value $E(A)$ with respect to the uniform measure in exponential time in $|A|$ and polynomial time in $\frac{1}{\epsilon}$.

**Proof.** For uniform distributions, by Lemma 15 for every $\epsilon > 0$, there exists $k$ such that $|E(A) - E(g[k])| \leq \frac{\epsilon}{2}$. The value $k$ is doubly-exponential in $|A|$ and polynomial in $\frac{1}{\epsilon}$. Then, by Lemma 19 we can compute $\gamma$ such that $|\gamma - E(g[k])| \leq \frac{\epsilon}{2}$ in exponential time in $|A|$ and polynomial in $\frac{1}{\epsilon}$. Thus, $\gamma$ differs from $E(A)$ by at most $\epsilon$. Since almost all words have the same value, we can approximate $D_A(\lambda)$ by comparing $\lambda$ with $\gamma$, i.e., $1$ is an $\epsilon$-approximation of $D_A(\lambda)$ if $\lambda \leq \gamma$, and otherwise $0$ is an $\epsilon$-approximation of $D_A(\lambda)$.

### 6.4 Non-uniform measures

We briefly discuss how to adapt Lemma 20 to all measures given by Markov chains. We sketch the main ideas.

**Key ideas.** Assuming that (a variant of) Lemma 14 holds for any probability measure given by a Markov chain, the proofs of Lemmas 15, 19 and 21 can be easily adapted. Therefore we focus on adjusting Lemma 14.

Observe that if a Markov chain $M$ produces all finite prefixes $u \in \Sigma^*$ with non-zero probability, then the proof of Lemma 14 can be straightforwardly adapted. Otherwise, if some finite words cannot be produced by a Markov chain $M$, then Lemma 14 may be false. However, if there are words $w$ such that $A$ has an infinite run on $w$, but $M$ does not emit $w$, we can restrict $A$ to reject such words. Therefore, we assume that for every word $w$, if $A$ has an infinite run on $w$, then $M$ has an infinite path with non-zero probability on $w$ (M
24 Non-deterministic weighted automata evaluated over Markov chains

emits $w$). Then, the current proof of Lemma 14 can be straightforwardly adapted to the probability measure given by $\mathcal{M}$. In consequence, we can compute $\mathbb{E}_{\mathcal{M},A}(\lambda)$ and $\mathbb{E}_A(\lambda)$ in exponential time in $|A|$ and polynomial time in $|\mathcal{M}|$ and $\frac{1}{\epsilon}$.

More precisely, we first observe that we may enforce $\mathcal{M}$ to be “deterministic”, i.e., for all states $s$ and letters $a$ at most one outgoing transition labeled with $a$ has positive probability. We can determinise $\mathcal{M}$ by extending the alphabet $\Sigma$ to $\Sigma \times S$, where $S$ is the set of states of $\mathcal{M}$. The second component in $\Sigma \times S$ encodes the target state in the transition. Observe that $A$ can be extended to the corresponding automaton $A'$ over $\Sigma \times S$ by cloning transitions, i.e., for every transition $(q,a,q')$, the automaton $A'$ has transitions $(q,(a,s),q')$ for every $s \in S$ (i.e., $A'$ ignores the state of $\mathcal{M}$). For such a deterministic Markov chain $\mathcal{M}'$, we define a deterministic $\omega$-automaton $A_{\mathcal{M}'}$ that accepts words emitted by $\mathcal{M}'$. Finally, we consider the automaton $A^D = A_{\mathcal{M}'} \times A'$, which has infinite runs only on words that are emitted by $\mathcal{M}'$. Therefore, as we discussed, we can adapt the proof of Lemma 20 in such a case and compute $\mathbb{E}_{\mathcal{M}',A^D}(\lambda)$ and $\mathbb{E}_A(\lambda)$ (in exponential time in $|A|$, polynomial time in $|\mathcal{M}|$ and $\frac{1}{\epsilon}$; notice that $|A^D|$ is polynomial in $|A|$). Finally, observe that $\mathbb{E}_{\mathcal{M},A}(\lambda) = \mathbb{E}_{\mathcal{M}',A^D}(\lambda)$ and $\mathbb{E}_A(\lambda) = \mathbb{E}_{A^D}(\lambda)$. In consequence, we have the following:

Lemma 21. Given a recurrent LIMAVG-automaton $A$, Markov chain $\mathcal{M}$, $\epsilon \in \mathbb{Q}^+$ and $\lambda \in \mathbb{Q}$, we can compute $\epsilon$-approximations of the distribution $\mathbb{D}_{\mathcal{M},A}(\lambda)$ and the expected value $\mathbb{E}(A)$ in exponential time in $|A|$ and polynomial time in $|\mathcal{M}|$ and $\frac{1}{\epsilon}$.

7 Non-recurrent automata

We present the approximation algorithms for all non-deterministic LIMAVG-automata over measures given by Markov chains.

Theorem 22. (1) For a non-deterministic LIMAVG-automaton $A$ the function $\mathcal{L}_A : \Sigma^* \rightarrow \mathbb{R}$ is measurable. (2) Given a non-deterministic LIMAVG-automaton $A$, Markov chain $\mathcal{M}$, $\epsilon \in \mathbb{Q}^+$, $\lambda \in \mathbb{Q}$, we can $\epsilon$-approximate the distribution $\mathbb{D}_{\mathcal{M},A}(\lambda)$ and the expected value $\mathbb{E}(A)$ in exponential time in $|A|$ and polynomial time in $|\mathcal{M}|$ and $\frac{1}{\epsilon}$.

Proof. Consider $A$ as an $\omega$-automaton. It has no acceptance conditions and hence we can determinise it with the standard power-set construction to a deterministic automaton $A^D$. Then, we construct a Markov chain $\mathcal{M} \times A^D$, compute all its BSCCs $R_1, \ldots, R_k$ along with the probabilities $p_1, \ldots, p_k$ of reaching each of these sets. This can be done in polynomial time in $|\mathcal{M}| \times |A^D|$, and hence polynomial in $\mathcal{M}$ and exponential in $A$. Let $H_1, \ldots, H_k$ be sets of paths in $\mathcal{M} \times A^D$ such that for each $i$, all $\rho \in H_i$ eventually reach $R_i$ and stay there forever. Observe that each $H_i$ is a Borel set; the set $H_0 = \bigcup_{p \geq 2} H_i$ of paths that stay in $R_i$ past position $p$ is closed and $H_i = \bigcup_{p \geq 2} H_i$. It follows that each $H_i$ is measurable. We show how to compute an $\epsilon$-approximation of the conditional expected value $\mathbb{E}_A(A \mid H_i)$.

Consider a BSCC $R_i$. The projection of $R_i$ on the first component $R^1_i$ is a BSCC in $\mathcal{M}$ and the projection on the second component $R^2_i$ is an SCC of $A^D$. Let $(s,A) \in R_i$. If we fix $s$ as the initial state of $R^1_i$ and $A$ as the initial state of $R^2_i$, then $R_i$ are all reachable states of $R^1_i \times R^2_i$. The set $R^2_i$ consists of the states of $A^D$, which are subsets of states of $\mathcal{M}$. Therefore, the union $\bigcup R^2_i$ is a subset of states of $\mathcal{M}$ and it consists of some SCCs $S_1, \ldots, S_m$ of $A$. All these SCCs are reachable, but it does not imply that there is a run of $\mathcal{M}$ that stays in $S_j$ forever. We illustrate that in the following example.

Consider the automaton $A$ presented in Figure 3, where $q_I$ is the initial state, and a single-state Markov chain $\mathcal{M}$ generating uniform distribution. Then, all paths in $\mathcal{M} \times A^D$ are eventually contained in $\mathcal{M} \times \{Q\}$, i.e., the second component consists of all states of
\( a : 1, b : 1 \xrightarrow{q_f} a : 0, b : 0 \xrightarrow{q_f} b : 0 \)

**Figure 3** An automaton with a reachable SCC \( q_f \) such that almost no runs stay in \( q_f \) forever

\( \mathcal{A} \). Still, if a word \( w \) has infinitely many letters \( a \), then \( \mathcal{A} \) has no (infinite) run on \( w \) that visits the state \( q_f \). The set of infinite words that contain finitely many letters \( a \) is countable and hence has probability 0. Therefore, almost all words (i.e., all except for some set of probability 0) have no run that visits the state \( q_f \).

To avoid such pathologies, we divide SCCs into two types: permanent and transitory. More precisely, for a path \( \rho \) in \( \mathcal{M} \times \mathcal{A}^D \) let \( w_\rho \) be the word labeling \( \rho \). We show that for each SCC \( S_j \), one of the following holds:

- **\( S_j \) is permanent**, i.e., for almost all paths \( \rho \in H_i \) (i.e., the set of paths of probability 1),
  
  the automaton \( \mathcal{A} \) has a run on the word \( w_\rho \) that eventually stays in \( S_j \) forever, or

- **\( S_j \) is transitory**, i.e., for almost all paths \( \rho \in H_i \), the automaton \( \mathcal{A} \) has no run on \( w_\rho \) that eventually stays in \( S_j \).

Consider an SCC \( S_j \). If \( S_j \) is permanent, then it is not transitory. We show that if \( S_j \) is not permanent, then it is transitory. Suppose that \( S_j \) is not permanent and consider any \( (s, A) \in H_i \). Almost all paths in \( H_i \) visit \( (s, A) \) and since \( S_j \) is not permanent, there exists an infinite path \( \rho \) that visits \( (s, A) \) and \( \mathcal{A} \) has no run on \( w_\rho \) that stays in \( S_j \) forever. Let \( u \) be the suffix of \( w_\rho \) that labels \( \rho \) past some occurrence of \( (s, A) \). We observe that \( \delta_i(A \cap S_j, u) = \emptyset \) and hence for some finite prefix \( u' \) of \( u \) we have \( \delta_i(A \cap S_j, u') = \emptyset \). Let \( p \) be the probability that \( \mathcal{M} \times \mathcal{A}^D \) in the state \( (s, A) \) generates a path labeled by \( u' \). The probability that a path that visits \( (s, A) \) at least \( t \) times does not contain \( (s, A) \) followed by labels \( u' \) is at most \((1 - p)^t \). Observe that for almost all paths in \( H_i \), the state \( (s, A) \) is visited infinitely often and hence almost all paths contain \( (s, A) \) followed by labels \( u' \) upon which the path leaves \( S_j \). Therefore, \( S_j \) is transitory.

To check whether \( S_j \) is permanent or transitory, observe that for any \( (s, A) \in H_i \), in the Markov chain \( \mathcal{M} \times \mathcal{A}^D \), we can reach the set \( \mathcal{M} \times \{ \emptyset \} \) from \( (s, A \cap S_j) \) if and only if \( S_j \) is transitory. The former condition can be checked in polynomial space.

We mark each SCC \( S_1, \ldots, S_k \) as permanent or transitory and for every permanent SCC \( S_j \), we compute an \( \epsilon \)-approximation of \( \mathbb{E}_\mathcal{M}(A[S_j] \mid H_i) \), which is the expected value of \( \mathcal{A} \) under condition \( H_i \) with the restriction to runs that eventually stay in \( S_j \). Observe that an \( \epsilon \)-approximation of \( \mathbb{E}_\mathcal{M}(A[S_j] \mid H_i) \) can be computed using Lemma 21. Indeed, we pick \( (s, A) \in H_i \) and observe that \( \mathcal{A} \) restricted to states \( S_j \) is recurrent (with an appropriate initial states). Finally, we pick the minimum \( \gamma \) over the computed expected values \( \mathbb{E}_\mathcal{M}(A[S_j] \mid H_i) \) and observe that almost all words in \( H_i \) have value \( \gamma \). It follows that \( \mathbb{E}_\mathcal{M}(A \mid H_i) = \gamma \).

In each BSCC \( R_i \), almost all words have value \( \mathbb{E}_\mathcal{M}(A \mid H_i) \). As we discussed earlier, each \( H_i \) is measurable, and hence the function \( \mathcal{L}_A : \Sigma^* \to \mathbb{R} \) is measurable. Moreover, to approximate the distribution \( \mathbb{D}_{\mathcal{M}, A}(\lambda) \), we sum probabilities of \( p_i \) of reaching the BSCCs \( R_i \) over \( R_i \)'s such that the \( \epsilon \)-approximation of \( \mathbb{E}_\mathcal{M}(A \mid H_i) \) is less or equal to \( \lambda \). Finally, we compute an \( \epsilon \)-approximation of \( \mathbb{E}_\mathcal{M}(A) \) from \( \epsilon \)-approximations of conditional expected values \( \mathbb{E}_\mathcal{M}(A \mid H_i) \) using the identity \( \mathbb{E}_\mathcal{M}(A) = \sum_{i=1}^k p_i \cdot \mathbb{E}_\mathcal{M}(A \mid H_i) \).

8 Determinising and approximating Lim Avg-automata

For technical simplicity, we assume that the distribution of words is uniform. However, the results presented here extend to all distributions given by Markov chains.
Recall that for the LimAvg automata, the value of almost all words (i.e., all except for some set of words of probability 0) whose optimal runs end up in the same SSC, is the same. This means that there is a finite set of values (not greater than the number of SSCs of the automaton) such that almost all words have their values in this set.

LimAvg-automata are not determinisable [11]. We say that a non-deterministic LimAvg-automaton \( \mathcal{A} \) is weakly determinisable if there is a deterministic LimAvg-automaton \( \mathcal{B} \) such that \( \mathcal{A} \) and \( \mathcal{B} \) have the same value over almost all words. From [12] we know that deterministic automata return rational values for almost all words, so not all LimAvg-automata are weakly determinisable. However, we can show the following.

\textbf{Theorem 23.} A LimAvg-automaton \( \mathcal{A} \) is weakly determinisable if and only if it returns rational values for almost all words.

\textbf{Proof sketch.} Assume an automaton \( \mathcal{A} \) with SSCs \( C_1, \ldots, C_m \). For each \( i \) let \( v_i \) be defined as the expected value of \( \mathcal{A} \) when its set of initial states is \( C_i \) and the run is bounded to stay in \( C_i \). If \( \mathcal{A} \) has no such runs for some \( C_i \), then \( v_i = \infty \).

We now construct a deterministic automaton \( \mathcal{B} \) with rational weights using the standard power-set construction. We define the cost function such that the cost of any transition from a state \( Y \) is the minimal value \( v_i \) such that \( v_i \) is rational and \( Y \) contains a state from \( C_i \). If there are no such \( v_i \), then we set the cost to the maximal cost of \( \mathcal{A} \). Roughly speaking, \( \mathcal{B} \) tracks in which SSCs \( \mathcal{A} \) can be and the weight corresponds to the SSC with the lowest value.

To see that \( \mathcal{B} \) weakly determinises \( \mathcal{A} \) observe that for almost all words \( w \), a run with the lowest value over \( w \) ends in some SSC and its value then equals the expected value of this component, which is rational as the value of this word is rational.

A straightforward corollary is that every non-deterministic LimAvg-automaton can be weakly determinised by a LimAvg-automaton with real weights.

Theorem 23 does not provide an implementable algorithm for weak-determinisation, because of the hardness of computing the values \( v_i \). It is possible, however, to approximate this automaton. We say that a deterministic LimAvg-automaton \( \mathcal{B} \) \( \epsilon \)-approximates \( \mathcal{A} \) if for almost every word \( w \) we have that \( L_B(w) \in [L_A(w) - \epsilon, L_A(w) + \epsilon] \).

\textbf{Theorem 24.} For every \( \epsilon > 0 \) and a non-deterministic LimAvg-automaton \( \mathcal{A} \), one can compute in exponential time a deterministic LimAvg-automaton that \( \epsilon \)-approximates \( \mathcal{A} \).

The proof of this theorem is similar to the proof of Theorem 23, except now it is enough to approximate the values \( v_i \), which can be done in exponential time.

\textbf{Acknowledgements.} Our special thanks go to Günter Rote who pointed out an error in an earlier version of our running example.
4 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.
5 Christel Baier, Stefan Kiefer, Joachim Klein, Sascha Klüppelholz, David Müller, and James Worrell. Markov chains and unambiguous Büchi automata. In *CAV 2016*, pages 23–42. Springer, 2016.
6 Michael Benedikt, Gabriele Puppis, and Cristian Riveros. Regular repair of specifications. In *LICS 2011*, pages 335–344, 2011.
7 Alberto Bertoni, Giancarlo Mauri, and Mauro Torelli. Some recursively unsolvable problems relating to isolated cutpoints in probabilistic automata. In *ICALP*, pages 87–94. Springer, 1977.
8 Patrick Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2013.
9 Udi Boker and Thomas A. Henzinger. Approximate determinization of quantitative automata. In *FSTTCS 2012*, pages 362–373. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.
10 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. A survey of stochastic games with limsup and liminf objectives. In *ICALP 2009*, pages 1–15, 2009.
11 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Quantitative languages. *ACM TOCL*, 11(4):23, 2010.
12 Krishnendu Chatterjee, Thomas A. Henzinger, and Jan Otop. Quantitative automata under probabilistic semantics. In *LICS 2016*, pages 76–85. ACM, 2016.
13 Krishnendu Chatterjee, Mickael Randour, and Jean-François Raskin. Strategy synthesis for multi-dimensional quantitative objectives. In *CONCUR 2012*, pages 115–131, 2012.
14 E.M. Clarke, T.A. Henzinger, and H. Veith. *Handbook of Model Checking*. Springer International Publishing, 2016.
15 Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, 1995.
16 Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata*. Springer, 1st edition, 2009.
17 W. Feller. *An introduction to probability theory and its applications*. Wiley, 1971.
18 Jerzy Filar and Koos Vrieze. *Competitive Markov decision processes*. Springer, 1996.
19 Charles Miller Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Soc., 2012.
20 Ashutosh Gupta, Cornelius Popeea, and Andrey Rybalchenko. Predicate abstraction and refinement for verifying multi-threaded programs. In *POPL 2011*, pages 331–344, 2011.
21 Thomas A. Henzinger and Jan Otop. Model measuring for discrete and hybrid systems. *Nonlinear Analysis: Hybrid Systems*, 23:166 – 190, 2017.
22 Andrew Hinton, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. PRISM: A tool for automatic verification of probabilistic systems. In *TACAS 2006*, pages 441–444, 2006.
23 John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Adison-Wesley Publishing Company, Reading, Massachusetts, USA, 1979.
24 Alexander Kechris. *Classical descriptive set theory*, volume 156. Springer Science & Business Media, 2012.
25 Daniel Kröb. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *Int. J. Algebra. Comput.*, 4(3):405–426, 1994.
26 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Quantitative analysis with the probabilistic model checker PRISM. *Electr. Notes Theor. Comput. Sci.*, 153(2):5–31, 2006.
27 Jakub Michaliszyn and Jan Otop. Non-deterministic weighted automata on random words. In *CONCUR 2018*, pages 10:1–10:16, 2018.
28 Christos H Papadimitriou. *Computational complexity*. Wiley, 2003.
29 Mickael Randour, Jean-François Raskin, and Ocan Sankur. Percentile queries in multi-dimensional markov decision processes. In *CAV 2015*, pages 123–139, 2015.
30 Klaus Friedrich Roth. Rational approximations to algebraic numbers. *Mathematika*, 2(1):1–20, 1955.
Leslie G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8(2):189–201, 1979.

Gerco van Heerdt, Justin Hsu, Joël Ouaknine, and Alexandra Silva. Convex language semantics for nondeterministic probabilistic automata. In *ICTAC 2018*, pages 472–492, 2018.

Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *FOCS 1985*, pages 327–338. IEEE Computer Society, 1985.