Stochastization of Weighted Automata[⋆]

Guy Avni and Orna Kupferman

School of Computer Science and Engineering, The Hebrew University, Jerusalem, Israel.

Abstract. *Nondeterministic weighted finite automata* (WFAs) map input words to real numbers. Each transition of a WFA is labeled by both a letter from some alphabet and a weight. The weight of a run is the sum of the weights on the transitions it traverses, and the weight of a word is the minimal weight of a run on it. In *probabilistic weighted automata* (PWFAs), the transitions are further labeled by probabilities, and the weight of a word is the expected weight of a run on it. We define and study *stochastization* of WFAs: given a WFA $\mathcal{A}$, stochastization turns it into a PWFA $\mathcal{A}'$ by labeling its transitions by probabilities. The weight of a word in $\mathcal{A}'$ can only increase with respect to its weight in $\mathcal{A}$, and we seek stochastizations in which $\mathcal{A}'$ α-approximates $\mathcal{A}$ for the minimal possible factor $\alpha \geq 1$. That is, the weight of every word in $\mathcal{A}'$ is at most α times its weight in $\mathcal{A}$. We show that stochastization is useful in reasoning about the competitive ratio of randomized online algorithms and in approximated determinization of WFAs. We study the problem of deciding, given a WFA $\mathcal{A}$ and a factor $\alpha \geq 1$, whether there is a stochastization of $\mathcal{A}$ that achieves an α-approximation. We show that the problem is in general undecidable, yet can be solved in PSPACE for a useful class of WFAs.

1 Introduction

A recent development in formal methods for reasoning about reactive systems is an extension of the Boolean setting to a multi-valued one. The multi-valued component may originate from the system, for example when propositions are weighted or when transitions involve costs and rewards [16], and may also originate from rich specification formalisms applied to Boolean systems, for example when asking quantitative questions about the system [7] or when specifying its quality [1]. The interest in multi-valued reasoning has led to growing interest in *nondeterministic weighted finite automata* (WFAs), which map an input word to a value from a semi-ring over a large domain [12,23].

Many applications of WFAs use the tropical semi-ring $\langle \mathbb{R}^+ \cup \{\infty\}, \min, +, \infty, 0\rangle$. There, each transition has a *weight*, the weight of a run is the sum of the weights of the transitions taken along the run, and the weight of a word is the minimal weight of a run on it. Beyond the applications of WFAs over the tropical semi-ring in quantitative reasoning about systems, they are used also in text, speech, and image processing, where the costs of the WFA are used in order to account for the variability of the data and to rank alternative hypotheses [11,24].

A different kind of applications of WFAs uses the semi-ring $\langle \mathbb{R}^+ \cup \{\infty\}, +, \times, 0, 1\rangle$. There, the weight of a run is the product of the weights of the transitions taken along it,

[⋆] The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no 278410, and from The Israel Science Foundation (grant no 1229/10).

and the weight of a word is the sum of the weights of the runs on it. In particular, when the weights on the transitions are in $[0,1]$ and form a probabilistic transition function (that is, for every state q and letter σ, the sum of the weights of the σ-transitions from q is 1), we obtain a *probabilistic finite automaton* (PFA, for short). In fact, the probabilistic setting goes back to the 60's [25].

The theoretical properties of WFAs are less clean and more challenging than these of their Boolean counterparts. For example, not all WFAs can be determinized [23], and the problem of deciding whether a given WFA has an equivalent deterministic WFA is open. As another example, the containment problem is undecidable for WFAs [21]. The multi-valued setting also leads to new questions about automata and their languages, like approximated determinization [3] or discounting models [13].

By combining the tropical and the probability semi-rings, we obtain a *probabilistic weighted finite automaton* (PWFA, for short). There, each transition has two weights, which we refer to as the *cost* and the *probability*. The weight that the PWFA assigns to a word is then the expected cost of the runs on it. That is, as in the tropical semi-ring, the cost of each run is the sum the costs of the transitions along the run, and as in probabilistic automata, the contribution of each run to the weight of a word depends on both its cost and probability. While PFAs have been extensively studied (e.g., [6]), we are only aware of [20] in which PWFAs were considered.

We introduce and study *stochastization* of WFAs. Given a WFA $\mathcal{A}$, stochastization turns it into a PWFA $\mathcal{A}'$ by labeling its transitions with probabilities. Recall that in a WFA, the weight of a word is the minimal weight of a run on it. Stochastization of a WFA $\mathcal{A}$ results in a PWFA $\mathcal{A}'$ with the same set of runs, and the weight of a word is the expected cost of these runs. Accordingly, the weight of a word in $\mathcal{A}'$ can only increase with respect to its weight in $\mathcal{A}$. Hence, we seek stochastizations in which $\mathcal{A}'$ *α-approximates* $\mathcal{A}$ for the minimal possible factor $\alpha \geq 1$. That is, the weight of every word in $\mathcal{A}'$ is at most α times its weight in $\mathcal{A}$. We note that stochastization has been studied in the Boolean setting in [14], where a PFA is constructed from an NFA. [1] Before describing our contribution, we motivate stochastization further.

In [2], the authors describe a framework for using WFAs over the tropical semi-ring in order to reason about *online algorithms*. An online algorithm can be viewed as a reactive system: at each round, the environment issues a request, and the algorithm should process it. The sequence of requests is not known in advance, and the goal of the algorithm is to minimize the overall cost of processing the sequence. Online algorithms for many problems have been extensively studied [8]. The most interesting question about an online algorithm refers to its *competitive ratio*: the worst-case (with respect to all input sequences) ratio between the cost of the algorithm and the cost of an optimal solution – one that may be given by an *offline* algorithm, which knows the input sequence in advance. An online algorithm that achieves a competitive ratio α is said to be *α-competitive*.

[1] Beyond considering the Boolean setting, the work in [14] concerns the ability to instantiate probabilities so that at least one word is accepted with probability arbitrarily close to 1. Thus, the type of questions and motivations are very different from these we study here in the weighted setting.

The framework in [2] models optimization problems by WFAs, relates the "unbounded look ahead" of the optimal offline algorithm with nondeterminism, and relates the "no look ahead" of online algorithms with determinism. The framework has been broaden to online algorithms with an extended memory or a bounded lookahead, and to a competitive analysis that takes into an account assumptions about the environment [3]. An additional useful broadening of the framework would be to consider *randomized* online algorithms, namely ones that may toss coins in order to choose their actions. Indeed, it is well known that many online algorithms that use randomized strategies achieve a better competitive ratio [8]. As we elaborate in Section 3.1, this means that rather than pruning the WFA that models an optimization problem to a deterministic one, we consider its stochastization.

Recall that not all WFAs have equivalent or even α-approximating deterministic WFAs. Stochastization is thus useful in finding an approximate solution to problems that are intractable in the nondeterministic setting and are tractable in the probabilistic one. We describe two such applications. One is *reasoning about quantitative properties of probabilistic systems*. In the Boolean setting, while one cannot model check probabilistic systems, typically given by a Markov chain or a Markov decision process, with respect to a specification given by means of a nondeterministic automaton, it is possible to take the product of a probabilistic system with a deterministic or a probabilistic automaton, making model checking easy for them [26]. In the weighted setting, a quantitative specification may be given by a weighted automaton. Here too the product can be defined only with a deterministic or a probabilistic automaton. By stochastizating a WFA specification, we obtain a PWFA (a.k.a. a *rewarded Markov chain* in this context [17]) and can perform approximated model checking. A second application is *approximated determinization*. Existing algorithms for α-determinization [23,3] handle families of WFAs in which different cycles that can be traversed by runs on the same word cannot have weights that differ in more than an α multiplicative factor (a.k.a. "the α-twins property"). Indeed, cycles as above induce problematic cycles for subset-construction-type determinization constructions. As we show, stochastization can average such cycles, leading to an approximated-determinization construction that successfully α-determinizes WFAs that do not satisfy the α-twin property and thus could not be α-determinized using existing constructions.Let us note that another candidate application is *weighted language equivalence*, which is undecidable for WFAs but decidable for PWFA [20]. Unfortunately, however, weighted equivalence becomes pointless once approximation enters the picture.

Given a WFA $\mathcal{A}$ and a factor $\alpha \geq 1$, the *approximated stochastization problem* (AS problem, for short) is to decide whether there is a stochastization of $\mathcal{A}$ that α-approximates it. We study the AS problem and show that it is in general undecidable. Special tractable cases include two types of restrictions. First, restrictions on α: we show that when $\alpha = 1$, the problem coincides with determinization by pruning of WFAs, which can be solved in polynomial time. Then, restrictions on the structure of the WFA: we define the class of constant-ambiguous WFAs, namely WFAs whose degree of nondeterminism is a constant, and show that the AS problem for them is in PSPACE. On the other hand, the AS problem is NP-hard already for 7-ambiguous WFAs, namely WFAs that have at most 7 runs on each word. Even more restricted are tree-like WFAs, for which the problem can be solved in polynomial time, and so is the problem of finding a minimal approximation factor α. We show that these restricted classes are still expressive enough to model interesting optimization problems.

Due to the lack of space, some examples and proofs are omitted and can be found in the full version, in the authors' URLs.

2 Preliminaries

A *nondeterministic finite weighted automaton* on finite words (WFA, for short) is a tuple $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0, \tau \rangle$, where Σ is an alphabet, Q is a finite set of states, $\Delta \subseteq Q \times \Sigma \times Q$ is a total transition relation (i.e., for every $q \in Q$ and $\sigma \in \Sigma$, there is at least one state $q' \in Q$ with $\langle q, \sigma, q' \rangle \in \Delta$), $q_0 \in Q$ is an initial state, and $\tau : \Delta \to \mathbb{R}^+$ is a weight function that maps each transition to a non-negative real value, which is the cost of traversing this transition. If for every $q \in Q$ and $\sigma \in \Sigma$ there is exactly one $q' \in Q$ such that $\langle q, \sigma, q' \rangle \in \Delta$, then $\mathcal{A}$ is a *deterministic WFA* (DWFA, for short). We assume that all states are reachable from the initial state. Consider a transition $t = \langle q, \sigma, q' \rangle \in \Delta$. We use *source*$(t)$, *label*$(t)$, and *target*$(t)$, to refer to q, σ, and q', respectively. It is sometimes convenient to use a transition function rather than a transition relation. Thus, we use $\delta_{\mathcal{A}} : Q \times \Sigma \to 2^Q$, where for $q \in Q$ and $\sigma \in \Sigma$, we define $\delta_{\mathcal{A}}(q, \sigma) = \{p \in Q : \langle q, \sigma, p \rangle \in \Delta\}$. When $\mathcal{A}$ is clear from the context we do not state it implicitly.

A *run* of $\mathcal{A}$ on a word $w = w_1 \ldots w_n \in \Sigma^*$ is a sequence of transitions $r = r_1, \ldots, r_n$ such that *source*$(r_1) \in Q_0$, for $1 \leq i < n$ we have *target*$(r_i) =$ *source*(r_{i+1}), and for $1 \leq i \leq n$ we have *label*$(r_i) = w_i$. For a word $w \in \Sigma^*$, we denote by *runs*$(\mathcal{A}, w)$ the set of all runs of $\mathcal{A}$ on w. Note that since Δ is total, there is a run of $\mathcal{A}$ on every word in Σ^*, thus $|$*runs*$(\mathcal{A}, w)| \geq 1$, for all $w \in \Sigma^*$.[2] The value of the run, denoted $val(r)$, is the sum of costs of transitions it traverses. That is, $val(r) = \sum_{1 \leq i \leq n} \tau(r_i)$. We denote by *first*$(r)$ and *last*(r) the states in which r starts and ends, respectively, thus $start(r) =$ *source*(r_1) and *last*$(r) =$ *target*(r_n). Since $\mathcal{A}$ is nondeterministic, there can be more than one run on each word. We define the value that $\mathcal{A}$ assigns to the word w, denoted $val(\mathcal{A}, w)$, as the value of the minimal-valued run of $\mathcal{A}$ on w. That is, for every $w \in \Sigma^*$, we define $val(\mathcal{A}, w) = \min\{val(r) : r \in$ *runs*$(\mathcal{A}, w)\}$.

A *probabilistic finite weighted automaton* on finite words (PWFA, for short) is $\mathcal{P} = \langle \Sigma, Q, D, q_0, \tau \rangle$, where Σ, Q, q_0, and τ are as in WFAs, and $D : Q \times \Sigma \times Q \to [0, 1]$ is a *probabilistic transition function*. That is, it assigns for each two states $q, p \in Q$ and letter $\sigma \in \Sigma$ the probability of moving from q to p with letter σ. Accordingly, we have $\sum_{p \in Q} D(q, \sigma, p) = 1$, for every $q \in Q$ and $\sigma \in \Sigma$. We sometimes refer to a transition relation $\Delta_D \subseteq Q \times \Sigma \times Q$ induced by D. For two states $q, p \in Q$ and letter $\sigma \in \Sigma$, we have $\Delta_D(q, \sigma, p)$ iff $D(q, \sigma, p) > 0$. Then, $\tau : \Delta_D \to \mathbb{R}^+$ assigns positive weights to transitions with a positive probability. As in WFAs, we assume that all states are accessible from the initial state by path with a positive probability. Note that if for every $q \in Q$ and $\sigma \in \Sigma$, there is a state $p \in Q$ with $D(q, \sigma, p) = 1$, then $\mathcal{P}$ is a DWFA.

[2] A different way to define WFAs would be to designate a set of accepting states. Then, the language of a WFA is the set of words that have an accepting run, and it assigns values to words in its language. Since it is possible to model acceptance by weights, our definition simplifies the setting and all states can be thought of as accepting.

A run $r = r_1, \ldots, r_n$ of $\mathcal{P}$ on $w = w_1 \ldots w_n \in \Sigma^*$ is a sequence of transitions defined as in WFAs. The probability of r, denoted $\Pr[r]$, is $\prod_{1 \leq i \leq n} D(r_i)$. Similarly to WFAs, for $w \in \Sigma^*$, we denote by *runs*$(\mathcal{P}, w)$ the set of all runs of $\mathcal{P}$ on w with positive probability. We define $val(\mathcal{P}, w)$ to be the expected value of a run of $\mathcal{P}$ on w, thus $val(\mathcal{P}, w) = \sum_{r \in \textit{runs}(\mathcal{P},w)} \Pr[r] \cdot val(r)$.

We say that a WFA $\mathcal{A}$ is k-ambiguous, for $k \in \mathbb{N}$, if k is the minimal number such that for every word $w \in \Sigma^*$, we have $|runs(\mathcal{A}, w)| \leq k$. We say that a WFA $\mathcal{A}$ is *constant-ambiguous* (a CA-WFA, for short) if $\mathcal{A}$ is k-ambiguous from some $k \in \mathbb{N}$. The definitions for PWFAs are similar, thus CA-PWFAs have a bound on the number of possible runs with positive probability.

A *stochastization* of a WFA $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0, \tau \rangle$ is a construction of a PWFA that is obtained from $\mathcal{A}$ by assigning probabilities to its nondeterministic choices. Formally, it is a PWFA $\mathcal{A}^D = \langle \Sigma, Q, D, q_0, \tau \rangle$ obtained from $\mathcal{A}$ such that D is consistent with Δ. Thus, $\Delta_D = \Delta$. Note that since the transition function of $\mathcal{A}$ is total, there is always a stochastization of $\mathcal{A}$. Note also that if $\delta(q, \sigma)$ is a singleton $\{p\}$, then $D(q, \sigma, p) = 1$.

Recall that in a nondeterministic WFA, the value of a word is the minimal value of a run on it. Stochastization of a WFA $\mathcal{A}$ results in a PWFA $\mathcal{A}^D$ with the same set of runs, and the value of a word is some average of the values of these runs. Accordingly, the value of a word in $\mathcal{A}^D$ can only increase with respect to its value in $\mathcal{A}$. We would like to find a stochastization with which $\mathcal{A}^D$ *approximates* $\mathcal{A}$

Consider two weighted automata $\mathcal{A}$ and $\mathcal{B}$, and a factor $\alpha \in \mathbb{R}$ such that $\alpha \geq 1$. We say that $\mathcal{B}$ α-approximates $\mathcal{A}$ if, for every word $w \in \Sigma^*$, we have $\frac{1}{\alpha} \cdot val(\mathcal{A}, w) \leq val(\mathcal{B}, w) \leq \alpha \cdot val(\mathcal{A}, w)$. We denote the latter also by $\frac{1}{\alpha}\mathcal{A} \leq \mathcal{B} \leq \alpha\mathcal{A}$. When $\alpha = 1$, we say that $\mathcal{A}$ and $\mathcal{B}$ are *equivalent*. Note that $\mathcal{A}$ and $\mathcal{B}$ are not necessarily the same type of automata.

A decision problem and an optimization problem naturally arise from this definition:

- *Approximation stochastization* (AS, for short): Given a WFA $\mathcal{A}$ and a factor $\alpha \geq 1$, decide whether there is a distribution function D such that $\mathcal{A}^D$ α-approximates $\mathcal{A}$, in which case we say that $\mathcal{A}^D$ is an α-stochastization of $\mathcal{A}$.
- *Optimal approximation stochastization* (OAS, for short): Given a WFA $\mathcal{A}$, find the minimal $\alpha \geq 1$ such that there an α-stochastization of $\mathcal{A}$.

Recall that for every distribution function D, we have that $\mathcal{A} \leq \mathcal{A}^D$. Thus, in both the AS and OAS problems it is sufficient to require $\mathcal{A}^D \leq \alpha \cdot \mathcal{A}$.

Remark 1. **[Tightening the approximation by a square-root factor]** For a WFA $\mathcal{A}$ and $\beta \in [0, 1]$, let $\mathcal{A}_\beta$ be $\mathcal{A}$ with costs multiplied by β. It is easy to see that for all WFAs $\mathcal{A}$ and $\mathcal{B}$, we have $\frac{1}{\sqrt{\alpha}} \cdot \mathcal{A} \leq \mathcal{B}_{1/\sqrt{\alpha}} \leq \sqrt{\alpha} \cdot \mathcal{A}$ iff $\mathcal{A} \leq \mathcal{B} \leq \alpha \cdot \mathcal{A}$. In particular, taking $\mathcal{B}$ to be $\mathcal{A}^D$ for some distribution function D for $\mathcal{A}$, we have that $\mathcal{A}^D$ α-approximates $\mathcal{A}$ iff $\mathcal{A}^D_{1/\sqrt{\alpha}}$ $\sqrt{\alpha}$-approximates $\mathcal{A}$. It follows that when altering of weights is possible, we can tighten the approximation by a square-root factor. □

3 Motivation

In Section 1, we discussed the application of stochastization in reasoning about quantitative properties of probabilistic systems, reasoning about randomized online algorithms, and approximated determinization. Below we elaborate on the last two.

3.1 A WFA-Based Approach to Reasoning about Online Algorithms

In this section we describe [2]'s WFA-based approach to reasoning about online algorithms and extend it to account for randomized ones. An online algorithm with requests in Σ and actions in A corresponds to a function $g : \Sigma^+ \to A$ that maps sequences of requests (the history of the interaction so far) to an action to be taken. In general, the algorithm induces an infinite state space, as it may be in different states after processing different input sequences in Σ^*. For a finite set S of configurations, we say that g *uses memory* S, if there is a regular mapping of Σ^* into S such that g behaves in the same manner on identical continuations of words that are mapped to the same configuration.

We model the set of online algorithms that use memory S and solve an optimization problem P with requests in Σ and actions in A, by a WFA $\mathcal{A}_P = \langle \Sigma, S, \Delta, s_0, \tau \rangle$, such that Δ and τ describe transitions between configurations and their costs, and s_0 is an initial configuration. Formally, $\Delta(s, \sigma, s')$ if the set $A' \subseteq A$ of actions that process the request σ from configuration s by updating the configuration to s' is non-empty, in which case $\tau(\langle s, \sigma, s' \rangle)$ is the minimal cost of an action in A'.

An offline algorithm knows the sequence of requests in advance and thus can resolve nondeterminism to obtain a minimal cost. Accordingly, the cost that an offline algorithm with state space S assigns to a sequence of requests $w \in \Sigma^*$ is exactly $val(\mathcal{A}_P, w)$. On the other hand, an online algorithm is a DWFA $\mathcal{A}'_P$ obtained from $\mathcal{A}_P$ by pruning nondeterministic choices. The competitive ratio of the online algorithm, namely the ratio between its performance and that of the offline algorithm, on the sequence of requests that maximizes this ratio, is then the factor α such that $\mathcal{A}'_P$ α-approximates $\mathcal{A}_P$. A randomized online algorithm for P that uses state space S can be viewed as a function from S to a probability distribution on A, which induces a probabilistic transition function on top of $\mathcal{A}_P$. Consequently, we have the following:

Theorem 1. *Consider an online problem P and a set S of configurations. Let $\mathcal{A}_P$ be a WFA with state space S that models online algorithms for P that use memory S. For all $\alpha \geq 1$, there is a randomized online algorithm for P using memory S that achieves competitive ratio α iff $\mathcal{A}_P$ has an α-stochastization.*

Example 1. **The Ski-rental problem.** Assume that renting skis costs \$1 per day and buying skis has a one-time cost of $\$M$. The online ski-rental problem copes with the fact it is not known in advance how many skiing days are left. Given an input request "skiing continues today", the online algorithm should decide whether to buy or rent skis. Typically, it is also assumed that renting skis is only allowed for at most $m \geq M$ consecutive days.

The WFA $\mathcal{A}$ induced by the ski-rental problem with parameters M and m is depicted in Fig. 1. Formally, $\mathcal{A} = \langle \{a\}, \{1, \ldots, m, q_{own}\}, \Delta, 1, \tau \rangle$, where Δ and τ are described below. A state $1 \leq i < m$ has two outgoing transitions: $\langle i, a, i+1 \rangle$ with weight 1, corresponds to renting skis at day i, and $\langle i, a, q_{own} \rangle$ with weight M, corresponding to buying skis at day i. Finally, there are transitions $\langle m, a, q_{own} \rangle$ with weight M and $\langle q_{own}, a, q_{own} \rangle$ with weight 0. The optimal deterministic online algorithm is due to [19]; rent skis for $M-1$ consecutive days, and buy skis on the M-th day, assuming skiing continues. It corresponds to the DWFA obtained by pruning all transitions but $\langle i, a, i+1 \rangle$, for $1 \leq i < M$, and $\langle M, a, q_{own} \rangle$. This DWFA achieves an optimal approximation factor of $2 - \frac{1}{M}$.

We describe a simple probabilistic algorithm that corresponds to a stochastization of $\mathcal{A}$ that achieves a better bound of $2 - \frac{1.5}{M}$. Intuitively, before skiing starts, toss a coin. If it turns out "heads", buy skis on the $(M-1)$-th day, and if it turns out "tails", buy on the M-th day. The corresponding distribution function D is depicted in red in Fig. 1 It is not hard to see that the worst case of this stochastization is attained by the word a^M for which we have $val(\mathcal{A}, a^M) = M$ and $val(\mathcal{A}^D, a^M) = \frac{1}{2} \cdot (M - 2 + M) + \frac{1}{2} \cdot (M - 1 + M) = 2M - 1.5$, thus $val(\mathcal{A}^D, a^M) \leq (2 - \frac{1.5}{M}) \cdot val(\mathcal{A}, a^M)$. Finding the optimal distribution function takes care [18,9] and can achieve an approximation of $1 + \frac{1}{(1+1/M)^M - 1} \approx e/(e-1) \approx 1.582$ for $M \gg 1$. $\square$

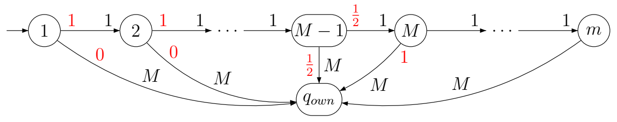

Fig. 1. The WFA that is induced by the ski-rental problem with parameters M and m.

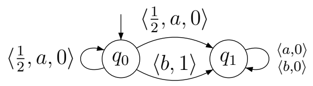

Fig. 2. A PWFA with no equivalent DWFA.

In the full version, we describe a WFA corresponds to the classical *paging* optimization problem. As we show there, it is sometimes useful to apply the stochastization after extending the state space of $\mathcal{A}_P$. In the case of paging with a cache of size k, determinization by pruning results in a k-approximation whereas stochastization results in an H_k-approximation, for $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{k} \approx \log(k)$. Thus, allowing the online algorithm to toss coins reduces the competitive ratio from k to H_k.

3.2 Approximated Determinization

Not all WFAs can be determinized. Since some applications require deterministic automata, one way to cope with WFAs that cannot be determinized is to α-*determinize* them, namely construct a DWFA that α-approximates them, for $\alpha \geq 1$. Our second application is an extension of the class of WFAs that can be approximately determinized.

In [23], Mohri describes a determinization construction for a subclass of WFAs – these that have the *twins property*. In [4], the authors define the α-*twins property*, for $\alpha \geq 1$, and describe an α-determinization construction for WFAs that satisfy it. We briefly define the properties below. Consider a WFA $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0, \tau \rangle$ and two states $q_1, q_2 \in Q$. We say that q_1 and q_2 are *pairwise reachable* if there is a word $u \in \Sigma^*$ such that there are runs of $\mathcal{A}$ on u that end in q_1 and q_2. Also, we say that q_1 and q_2 have the t-twins property if they are either not pairwise reachable, or, for every

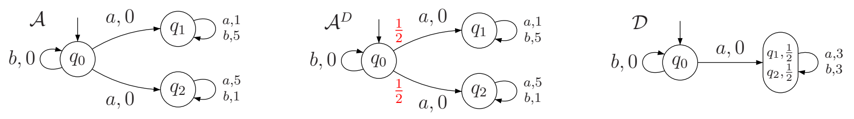

Fig. 3. An illustration of our algorithm for approximated determinization of WFAs.

word $v \in \Sigma^*$, if π_1 and π_2 are v-labeled cycle starting from q_1 and q_2, respectively, then $val(\pi_1) \leq t \cdot val(\pi_2)$. We say that $\mathcal{A}$ has the α-twins property iff every two states in $\mathcal{A}$ have the α-twins property. The α-twins property coincides with Mohri's twins property when $\alpha = 1$.

The α-twins property can be thought of as a *lengthwise* requirement; there might be many runs on a word, but there is a bound on how different the runs are. Our algorithm applies to CA-WFAs. Recall that such WFAs have a dual, *widthwise*, property: the number of runs on a word is bounded by some constant. The algorithm proceeds as follows. Given a CA-WFA $\mathcal{A}$, we first find an α-stochastization $\mathcal{A}^D$ of it. Since stochastization maintains constant ambiguity, we obtain a CA-PWFA. As we show in Theorem 7, CA-PWFA can be determinized, thus we find an α-determinization of $\mathcal{A}$.

Example 2. Consider the WFA $\mathcal{A}$ that is depicted in Fig. 3. Note that $\mathcal{A}$ is 2-ambiguous. The optimal stochastization of $\mathcal{A}$ is given by the distribution function D that assigns $D(q_0, a, q_1) = D(q_0, a, q_2) = \frac{1}{2}$. The resulting PWFA $\mathcal{A}^D$ is also depicted in the figure. Then, we construct the DWFA $\mathcal{D}$ by applying the determinization construction of Theorem 7. Clearly, the DWFA $\mathcal{D}$ 3-approximates $\mathcal{A}$.

We note that $\mathcal{A}$ has the 5-twins property, and this is the minimal t. That is, for every $t < 5$, $\mathcal{A}$ does not have the t-twins property. The DWFA $\mathcal{D}'$ that is constructed from $\mathcal{A}$ using the approximated determinization construction of [4] has the same structure as $\mathcal{D}$ only that the self loops that have weight 3 in $\mathcal{D}$, have weight 5 in $\mathcal{D}'$. Thus, $\mathcal{D}'$ 5-approximates $\mathcal{A}$.

4 Stochastization of General WFAs

In this section we study the AS and OAS problems for general WFA. We start with some good news, showing that the exact stochastization problem can be solved efficiently. Essentially, it follows from the fact that exact stochastization amounts to determinization by pruning, which can be solved in polynomial time [2]. See the full version for details.

Theorem 2. *The exact stochastization problem can be solved in polynomial time.*

We proceed to the bad news.

Theorem 3. *The AS problem is undecidable.*

Proof. In Section 1 we mentioned PFA, which add probabilities to finite automata. Formally, a PFA is $\mathcal{P} = \langle \Sigma, Q, P, q_0, F \rangle$, where $F \subseteq Q$ is a set of accepting states and the other components are as in PWFAs. Given a word $w \in \Sigma^*$, each run of $\mathcal{P}$ on

w has a probability. The value $\mathcal{P}$ assigns to w is the probability of the accepting runs. We say that $\mathcal{P}$ is *simple* if the image of P is $\{0, 1, \frac{1}{2}\}$. For $\lambda \in [0, 1]$, the λ-emptiness problem for PFAs gets as input a PFA $\mathcal{P}$, and the goal is to decide whether there is a word $w \in \Sigma^*$ such that $val(\mathcal{P}, w) > \lambda$. It is well known that the emptiness problem for PFAs is undecidable for $\lambda \in (0, 1)$ [6,22]. Furthermore, it is shown in [15] that the emptiness problem is undecidable for simple PFAs and $\lambda = \frac{1}{2}$. In the full version we construct, given a simple PFA $\mathcal{P}$, a WFA $\mathcal{A}$ such that $\mathcal{P}$ is $\frac{1}{2}$-empty iff there is an $\frac{2+\sqrt{7}}{3}$-stochastization of $\mathcal{A}$. $\square$

5 Stochastization of Constant Ambiguous WFAs

Recall that a CA-WFA has a bound on the number of runs on each word. We show that the AS problem becomes decidable for CA-WFAs. For the upper bound, we show that when the ambiguity is fixed, the problem is in PSPACE. Also, when we further restrict the input to be a *tree-like* WFAs, the OAS problem can be solved in polynomial time. We note that while constant ambiguity is a serious restriction, many optimization problems, including the ski-rental we describe here, induce WFAs that are constant ambiguous. Also, many theoretical challenges for WFAs like the fact that they cannot be determinized, apply already to CA-WFA. We start with a lower bound, which we prove in the full version by a reduction from 3SAT.

Theorem 4. *The AS problem is NP-hard for 7-ambiguous WFAs.*

Consider a k-ambiguous WFA $\mathcal{A}$, a factor α, and a distribution function D. When k is fixed, it is possible to decide in polynomial time whether $\mathcal{A}^D$ α-approximates $\mathcal{A}$. Thus, a tempting approach to show that the AS problem is in NP is to bound the size of the optimal distribution function. We show below that this approach is doomed to fail, as the optimal distribution function may involve irrational numbers even when the WFA has only rational weights.

Theorem 5. *There is a 4-ambiguous WFA with rational weights for which every distribution that attains the optimal approximation factor includes irrational numbers.*

Proof. In the full version we construct a WFA $\mathcal{A}$ that includes states q_1, q_2, and q_3, where for $i = 1, 2, 3$, the state q_i has nondeterministic choices t_i and t'_i. We define $\mathcal{A}$ so that for $i \neq j \in \{1, 2, 3\}$, an optimal distribution function D satisfies $D(t_i) \cdot D(t_j) = \frac{1}{2}$. Thus, $D(t_i) = \frac{1}{\sqrt{2}}$. $\square$

We now turn to show that the AS problem is decidable for CA-WFAs. Our proof is based on the following steps: (1) We describe a determinization construction for CA-PWFAs. The probabilities of transitions in the CA-PWFA affects the weights of the transitions in the obtained DWFA. (2) Consider a CA-WFA $\mathcal{A}$. We attribute each transition of $\mathcal{A}$ by a variable indicating the probability of taking the transition. We define constraints on the variables so that there is a correspondence between assignments and distribution functions. Let $\mathcal{A}'$ be the obtained CA-PWFA. Note that rather than being a standard probabilistic automaton, it is *parameterized*, thus it has the probabilities as variables. Since $\mathcal{A}'$ has the same structure as $\mathcal{A}$, it is indeed constant ambiguous. (3) We apply the determinization construction to $\mathcal{A}'$. The variables now appear in the weights of the obtained parameterized DWFA. (4) Given $\alpha \geq 1$, we add constraints on the variables that guarantee that the assignment results in an α-stochastization of $\mathcal{A}$. For that, we define a parameterized WFA $\mathcal{A}''$ that corresponds to $\alpha\mathcal{A} - \mathcal{A}'$ and the constraints ensure that it assigns a positive cost to all words. (5) We end up with a system of polynomial constraints, where the system is of size polynomial in $\mathcal{A}$. Deciding whether such a system has a solution is known to be in PSPACE.[3]

We now describe the steps in more detail.

Determinization of CA-WFAs Before we describe the determinization construction for CA-WFAs, we show that PWFAs are not in general determinizable. This is hardly surprising as neither WFAs nor PFAs are determinizable. We note, however, that the classic examples that show that WFAs are not determinizable use a 2-ambiguous WFA, and as we show below, 2-ambiguous PWFAs are determinizable.

Theorem 6. *There is a PWFA with no equivalent DWFA.*

Proof. Consider the PWFA depicted in Fig. 2. Assume towards contradiction that there is a DWFA $\mathcal{D}$ that is equivalent to $\mathcal{A}$. Let γ be the smallest absolute value on one of $\mathcal{B}$'s transitions. Then, $\mathcal{B}$ cannot assign values of higher precision than γ. In particular, for $n \in \mathbb{N}$ such that $2^{-n} < \gamma$, it cannot assign the value 2^{-n} to the word $a^n b$. □

Consider a PWFA $\mathcal{P} = \langle \Sigma, Q, D, q_0, \tau \rangle$. For a state $q \in Q$ and $\sigma \in \Sigma$, we say that $\mathcal{P}$ has a *σ-probabilistic choice* at q if there is a transition $\langle q, \sigma, q' \rangle \in \Delta_D$ such that $0 < D(\langle q, \sigma, q' \rangle) < 1$. This is indeed a choice as Δ_D must include a different σ-labeled outgoing transition from q with positive probability. We sometimes refer to such a choice simply as a *probabilistic choice*. We extend the definition to runs as follows. Consider a run $r = r_1, \ldots, r_n$ of $\mathcal{P}$ on some word. We say that r makes a probabilistic-choice at index $1 \leq i \leq n$ if there is a *label*(r_i)-choice at state *source*(r_i). We then say that r *chooses* the transition r_i. Note that when $\Pr[r] > 0$ and the probabilistic choices of r are $i_1, \ldots, i_\ell$, then $\Pr[r] = \prod_{1 \leq j \leq \ell} D(t_{i_j})$.

Given $\mathcal{P}$, we construct a DWFA $\mathcal{D} = \langle \Sigma, S, \Delta, q_0', \tau' \rangle$ equivalent to $\mathcal{P}$ as follows. The states of $\mathcal{D}$ are $S \subseteq 2^{Q \times [0,1]}$. Thus, a state in S is a set of pairs, each consisting of a state q in $\mathcal{P}$ and the probability of reaching q. Thus, the construction is similar to the subset construction used for determinizing NFWs, except that each state in $\mathcal{P}$ is paired with the probability of visiting it in $\mathcal{D}$. More formally, consider such a state $s = \{\langle q_1, p_1 \rangle, \ldots, \langle q_\ell, p_\ell \rangle\}$ and a run r on a word $w \in \Sigma^*$ that ends in s. Then, for $1 \leq i \leq \ell$, we have $p_i = \Pr[\{r \in runs(\mathcal{P}, w) : r \text{ end in } q_i\}]$. We define the transitions and their weights accordingly: There is a transition $t = \langle s, \sigma, s' \rangle \in \Delta$ iff for every pair $\langle q_i', p_i' \rangle \in s'$ we have $p_i' = \sum_{\langle q_j, p_j \rangle \in q} D(q_j, \sigma, q_i') \cdot p_j$ and $p_i' > 0$. For $s \in S$, the states of s are $st(s) = \{q \in Q : \langle q, p \rangle \in s\}$. The weight of t is $\tau'(t) = \sum_{t' = \langle q_i, \sigma, q_j' \rangle \in st(q) \times \{\sigma\} \times st(q')} \tau(t') \cdot p_j \cdot D(t')$.

While it is not hard to see that $\mathcal{D}$ is equivalent to $\mathcal{P}$, there is no a-priori bound on the size of $\mathcal{D}$. In the full version we show that when $\mathcal{P}$ is constant ambiguous, then $\mathcal{D}$

[3] The latter problem, a.k.a. the *existential theory of the reals* [10] is known to be NP-hard and in PSPACE. Improving its upper bound to NP would improve also our bound here.

has a finite number of states. Intuitively, we relate the probabilities that appear in the states of $\mathcal{D}$ with probabilistic choices that the runs of $\mathcal{P}$ perform. Then, we argue that a run of $\mathcal{P}$ on some word $w \in \Sigma^*$ performs at most $k-1$ probabilistic choices as every choice "spawns" another run of $\mathcal{P}$ on w and $|runs(\mathcal{P}, w)| \leq k$. Thus, we can bound the number of states in $\mathcal{D}$, implying the following.

Theorem 7. *Consider a k-ambiguous PWFA $\mathcal{P}$ with m transitions. There is a DWFA $\mathcal{D}$ that is equivalent to $\mathcal{P}$ with $m^{O(k^2)}$ states.*

An upper bound for the AS problem We are now ready to present the solution to the stochastization problem for CA-WFAs with weights in $\mathbb{Q}^+$. Given an input WFA $\mathcal{A}$ and a factor $\alpha \geq 1$, we construct a *polynomial feasibility problem*. The input to such a problem is a set of n variables X and polynomial constraints $P_i(\overline{x}) \leq 0$, for $1 \leq i \leq m$. The goal is to decide whether there is a point $\overline{p} \in \mathbb{R}^n$ that satisfies all the constraints, thus $P_i(\overline{p}) \leq 0$, for $1 \leq i \leq m$. The problem is known to be in PSPACE [10].

Theorem 8. *The AS problem for CA-WFAs is in PSPACE.*

Proof. We describe the intuition and the details can be found in the full version. Consider a k-ambiguous WFA $\mathcal{A} = \langle \Sigma, Q, \Delta, q_0, \tau \rangle$. We associate with $\mathcal{A}$ the following constraints. First, let $X_D = \{x_t : t \in \Delta\}$ be a set of variables, and let $\mathcal{A}^D$ be a parameterized stochastization of $\mathcal{A}$ in which the probability of a transition t is x_t. Next, let $\mathcal{D}^D$ be the parameterized DWFA obtained by applying to $\mathcal{A}^D$ the determinization construction of Theorem 7. Since the variables of $\mathcal{A}^D$ are the probabilities of the transitions and in the determinization construction the probabilities move to the transition weights, the variables in $\mathcal{D}^D$ appear only in the weights.

The first type of constraints are ones that ensure that the assignment to the variables in X_D forms a probabilistic transition function. The second type depends on α and ensures that $\mathcal{D}^D \leq \alpha \cdot \mathcal{A}$. This is done by applying constraints that ensures the emptiness of the parameterized WFA $\mathcal{A}'' = (\alpha\mathcal{A} - \mathcal{D}^D)$. the constraints are are based on *Bellman equations*, ensuring that the value of all words in $\mathcal{A}''$ is positive. The number of additional constraints is then polynomial in n and in the number of transitions of $\mathcal{A}''$. Thus, all in all we have polynomially many constraints and the numbers that appear in them are of size polynomial in the size of the weights of $\mathcal{A}$. Thus, the size of the program is polynomial in the size of $\mathcal{A}$, and we are done. □

Remark 2. A different way to specify emptiness by constraints is to consider all simple paths and cycles in the automaton, as was done in [5]. A polynomial treatment of the exponentially many linear constraints then involves a separation oracle, and the ellipsoid method. The method we use here has polynomially many linear constraints to start with, and its implementation is considerably more practical. □

Remark 3. **[Solving the OAS problem for tree-like WFAs]** We say that a WFA $\mathcal{A}$ is *tree-like* if the graph induced by its nondeterministic choices is a tree (note that $\mathcal{A}$ is constant ambiguous if the graph is acyclic). Thus, intuitively, for every nondeterministic choice t, there is one way to resolve other nondeterministic choices in order to reach t. Note that the WFA that corresponds to the ski-rental problem as well as the WFA from Example 2 are tree-like. In the full version we show that for every fixed $k \in \mathbb{N}$, the OAS problem can be solved in polynomial time for tree-like k-ambiguous WFAs. □

References

1. S. Almagor, U. Boker, and O. Kupferman. Formalizing and reasoning about quality. In *Proc. 40th ICALP*, LNCS 7966, pages 15 – 27, 2013.
2. B. Aminof, O. Kupferman, and R. Lampert. Reasoning about online algorithms with weighted automata. *TALG*, 6(2), 2010.
3. B. Aminof, O. Kupferman, and R. Lampert. Formal analysis of online algorithms. In *Proc. 9th ATVA*, LNCS 6996, pages 213–227, 2011.
4. B. Aminof, O. Kupferman, and R. Lampert. Rigorous approximated determinization of weighted automata. *TCS*, 480:104–117, 2013.
5. G. Avni and O. Kupferman. Parameterized weighted containment. *TOCL*, 16(1):6, 2014.
6. P. Azaria. *Introduction to Probabilistic Automata*. Academic Press, Inc., 1971.
7. U. Boker, K. Chatterjee, T.A. Henzinger, and O. Kupferman. Temporal specifications with accumulative values. In *Proc. 26th LICS*, pages 43–52, 2011.
8. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
9. N. Buchbinder, K. Jain, and J. Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *Proc. 15th ESA*, pages 253–264, 2007.
10. J. Canny. Some algebraic and geometric computations in PSPACE. In *Proc. STOC*, pages 460–467, 1988.
11. K. Culik and J. Kari. Digital images and formal languages. *Handbook of formal languages, vol. 3: beyond words*, pages 599–616, 1997.
12. M. Droste, W. Kuich, and H. Vogler (eds.). *Handbook of Weighted Automata*. 2009.
13. M. Droste and G. Rahonis. Weighted automata and weighted logics with discounting. *TCS*, 410(37):3481–3494, 2009.
14. N. Fijalkow, H. Gimbert, F. Horn, and Y. Oualhadj. Two recursively inseparable problems for probabilistic automata. In *Proc. 39th MFCS*, LNCS 8634, pages 267–278, 2014.
15. H. Gimbert and Y. Oualhadj. Probabilistic automata on finite words: Decidable and undecidable problems. In *Proc. 37th ICALP*, LNCS 6199, pages 527–538, 2010.
16. T.A. Henzinger. From Boolean to quantitative notions of correctness. In *Proc. 37th POPL*, pages 157–158, 2010.
17. R.A. Howard. Dynamic Probabilistic Systems, Volume II: Semi-Markov and Decision Processes. Vol 2, Courier Corporation, 2013.
18. A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. S. Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11(6):542–571, 1994.
19. A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:77–119, 1988.
20. S. Kiefer, A. S. Murawski, J. Ouaknine, B. Wachter, and J. Worrell. On the complexity of equivalence and minimisation for q-weighted automata. *LMCS*, 9(1), 2013.
21. D. Krob. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. *IJAC*, 4(3):405–425, 1994.
22. O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artif. Intell.*, 147(1-2):5–34, 2003.
23. M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.
24. M. Mohri, F.C.N. Pereira, and M. Riley. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88, 2002.
25. M. O. Rabin. Probabilistic automata. *I&C*, 6:230–245, 1963.
26. M.Y. Vardi. Probabilistic Linear-Time Model Checking: An Overview of the Automata-Theoretic Approach. *ARTS*, pages 265-276, 1999.