# Determinizing Discounted-Sum Automata*

**Udi Boker**[1,2] **and Thomas A. Henzinger**[2]

1 **Hebrew University of Jerusalem**
2 **IST Austria**

**Abstract**

A discounted-sum automaton (NDA) is a nondeterministic finite automaton with edge weights, which values a run by the discounted sum of visited edge weights. More precisely, the weight in the $i$-th position of the run is divided by $\lambda^i$, where the discount factor $\lambda$ is a fixed rational number greater than 1. Discounted summation is a common and useful measuring scheme, especially for infinite sequences, which reflects the assumption that earlier weights are more important than later weights. Determinizing automata is often essential, for example, in formal verification, where there are polynomial algorithms for comparing two deterministic NDAs, while the equivalence problem for NDAs is not known to be decidable. Unfortunately, however, discounted-sum automata are, in general, not determinizable: it is currently known that for every rational discount factor $1 < \lambda < 2$, there is an NDA with $\lambda$ (denoted $\lambda$-NDA) that cannot be determinized.

We provide positive news, showing that every NDA with an integral factor is determinizable. We also complete the picture by proving that the integers characterize exactly the discount factors that guarantee determinizability: we show that for every rational factor $\lambda \notin \mathbb{N}$, there is a nondeterminizable $\lambda$-NDA. Finally, we prove that the class of NDAs with integral discount factors enjoys closure under the algebraic operations min, max, addition, and subtraction, which is not the case for general NDAs nor for deterministic NDAs. This shows that for integral discount factors, the class of NDAs forms an attractive specification formalism in quantitative formal verification. All our results hold equally for automata over finite words and for automata over infinite words.

**1998 ACM Subject Classification** F.4.3 Formal Languages

**Keywords and phrases** Discounted-sum automata, determinization, quantitative verification

**Digital Object Identifier** 10.4230/LIPIcs.CSL.2011.82

## 1 Introduction

Discounting the influence of future events is a key paradigm in economics and it is studied in game theory (e.g. [12, 1]), Markov decision processes (e.g. [10, 9]), and automata theory (e.g. [5, 8, 2, 3, 4]). Discounted summation formalizes the concept that an immediate reward is better than a potential one in the far-away future, as well as that a potential problem in the future is less troubling than a current one.

A discounted-sum automaton (NDA) is a nondeterministic automaton with rational weights on the transitions, where the value of a run is the discounted summation of the weights along it. Each automaton has a fixed discount-factor $\lambda$, which is a rational number bigger than 1, and the weight in the $i$th position of a run is divided by $\lambda^i$. The value of a word is the minimal value of the automaton runs on it. Hence, an NDA realizes a function from words to real numbers. Two automata are equivalent if they realize the same function, namely if they assign the same value to every word.

* The research was supported by the FWF NFN Grant No S11407-N23 (RiSE), the EU STREP Grant COMBEST, the ERC Advanced Grant QUAREM, and the EU NOE Grant ArtistDesign.

Discounted summation is of special interest for automata over infinite words. There are two common ways to adjust standard summation for handling infinite sequences: discounting and limit-averaging. The latter, which relates to the input suffixes, has been studied a lot in mean-payoff games and, more recently, in limit-average automata [2, 6]; the former, which relates more to the input prefixes, has received comparatively little attention.

Automata are widely used in formal verification, for which automata comparison is fundamental. Specifically, one usually considers the following three questions, ordered from the most difficult one to the simplest one: general comparison (language inclusion), universality, and emptiness. In the Boolean setting, where automata assign Boolean values to the input words, the three questions, with respect to automata $\mathcal{A}$ and $\mathcal{B}$, are whether $\mathcal{A} \subseteq \mathcal{B}$, $\mathcal{A} = True$, and $\mathcal{A} = False$. In the quantitative setting, where automata assign numeric values to the input words, the universality and emptiness questions relate to a constant threshold, usually 0. Thus, the three questions are whether $\mathcal{A} \leq \mathcal{B}$, $\mathcal{A} \leq 0$, and $\mathcal{A} \geq 0$.

A central problem with these quantitative automata is that only the emptiness question is known to be solvable. For limit-average automata, the two other questions are undecidable [6]. For NDAs, it is an open question whether universality and comparison are decidable. This is not the case with DDAs, for which all three questions have polynomial solutions [12, 1, 2]. Unfortunately, NDAs cannot, in general, be determinized. It is currently known that for every rational discount-factor $1 < \lambda < 2$, there is a $\lambda$-NDA that cannot be determinized [2].

It turns out, quite surprisingly, that discounting by an integral factor forms a "well behaved" class of automata, denoted "integral NDAs", allowing for determinization (Section 3) and closed under the algebraic operations min, max, addition and subtraction (Section 5). The above closure is of special interest, as neither NDAs nor DDAs are closed under the max operation (Theorem 9). Furthermore, the integers, above 1, characterize exactly the set of discount factors that guarantee determinizability (Section 4). That is, for every rational factor $\lambda \notin \mathbb{N}$, there is a non-determinizable $\lambda$-NDA.

The discounted summation intuitively makes NDAs more influenced by word-prefixes than by word-suffixes, suggesting that some basic properties are shared between automata over finite words and over infinite words. Indeed, all the above results hold for both models. Yet, the equivalence relation between automata over infinite words is looser than the one on finite words. That is, if two automata are equivalent with respect to finite words then they are also equivalent with respect to infinite words, but not vice versa (Lemma 3).

The above results relate to complete automata; namely, to automata in which every state has at least one transition over every alphabet letter. For incomplete automata or, equivalently, for automata with $\infty$-weights, no discount factor can guarantee determinization (Section 4.2).

Our determinization procedure, described in Section 3.1, is an extension of the subset construction, keeping a "recoverable-gap" value to each element of the subset. Intuitively, the "gap" of a state $q$ over a finite word $u$ stands for the extra cost of reaching $q$, compared to the best possible value so far. This extra cost is multiplied, however, by $\lambda^{|u|}$, to reflect the $\lambda^{|u|}$ division in the value-computation of the suffixes. A gap of $q$ over $u$ is "recoverable" if there is a suffix $w$ that "recovers" it, meaning that there is an optimal run over $uw$ that visits $q$ after reading $u$. Due to the discounting of the future, once a gap is too large, it is obviously not recoverable. Specifically, for every $\lambda$, we have that $\sum_{i=0}^{\infty}(\frac{1}{\lambda^i}) = \frac{1}{1-\frac{1}{\lambda}} = \frac{\lambda}{\lambda-1} \leq 2$. Hence, our procedure only keeps gaps that are smaller than twice the maximal difference between the automaton weights.

The determinization procedure may be used for an arbitrary $\lambda$-NDA, always providing an equivalent $\lambda$-DDA, if terminating. Yet, it is guaranteed to terminate for a $\lambda$-NDA with

$\lambda \in \mathbb{N}$, while it might not terminate in the case that $\lambda \in \mathbb{Q} \setminus \mathbb{N}$.

For integral NDAs, the key observation is that there might only be finitely many recoverable gaps (Lemma 2). More precisely, for an integral NDA $\mathcal{A}$, there might be up to $m$ recoverable gaps, where $m$ is the maximal difference between the weights in $\mathcal{A}$, multiplied by the minimal common divider of all weights. Accordingly, our determinization procedure generates a DDA with up to $m^n$ states, where $n$ is the number of states in $\mathcal{A}$. We show that this state blow-up is tight, using a rich alphabet of size exponential in the number of states (Theorem 6). The unavoidable state blow-up for the case that the alphabet size is linear in the number of states is left as an open problem.

For nonintegral NDAs, the key observation is that the recoverable gaps might be arbitrarily dense (Theorem 7). Hence, the bound on the maximal value of the gaps cannot guarantee a finite set of recoverable gaps. Different gaps have, under the appropriate setting, suffixes that distinguish between them, implying that an equivalent deterministic automaton must have a unique state for each recoverable-gap (Lemma 5). Therefore, an automaton that admits infinitely many recoverable gaps cannot be determinized.

It turns out that closure under algebraic operation is also closely related to the question of whether the set of recoverable gaps is finite. Considering the operations of addition, subtraction, minimization, and maximization, the latter is the most problematic one, as the value of a word is defined to be the minimal value of the automaton runs on it. For two NDAs, $\mathcal{A}$ and $\mathcal{B}$, one may try to construct an automaton $\mathcal{C} = \max(\mathcal{A}, \mathcal{B})$, by taking the product of $\mathcal{A}$ and $\mathcal{B}$, while maintaining the recoverable gaps of $\mathcal{A}$'s original states, compared to $\mathcal{B}$'s original states. This approach indeed works for integral NDAs (Theorem 10). Note that determinizability is not enough, as neither NDAs nor DDAs are closed under the max operation. Furthermore, we show, in Theorem 9, that there are two DDAs, $\mathcal{A}$ and $\mathcal{B}$, such that there is no NDA $\mathcal{C}$ with $\mathcal{C} = \max(\mathcal{A}, \mathcal{B})$. For precluding the existence of such a nondeterministic automaton $\mathcal{C}$, we cannot make usage of Lemma 5, and thus use a more involved, "pumping-style", argument with respect to recoverable gaps.

**Related work.** Weighted automata are often handled as *formal power series*, mapping words to a *semiring* [7]. By this view, the weight of a run is the semiring-multiplication of the transition weights along it, while the weight of a word is the semiring-addition of its possible run weights. For this setting, there are numerous works, including results on determinization [11, 7]. However, discounted-sum automata do not fall into this setting, as discounted summation cannot be described as the multiplication operation of a semiring. The latter is required to have an identity element $\bar{1}$, such that for every element $e$, $\bar{1}e = e\bar{1} = e$. One can check that discounted summation cannot allow for an identity element, which is, in a sense, the core reason for its different behavior. Formal power series are generalized, in [8], for handling discounted summation. The weight of a run is defined to be a "skewed multiplication" of the weights along it, where this "skewing" corresponds to the discounting operation. Yet, [8] mainly considers the equivalence between recognizable series and rational series, and does not handle automata determinization.

Discounted Markov decision processes (e.g. [10, 9]) and discounted games (e.g. [12, 1]) generalize, in some sense, deterministic discounted-sum automata. The former adds probabilities and the latter allows for two player choices. However, they do not cover nondeterministic automata. One may note that nondeterminism relates to "blind games", in which each player cannot see the other player's moves, whereas in standard games the players have full information on all moves. Indeed, for a discounted-game, one can always compute an optimal strategy [12], while a related question on nondeterministic discounted-sum automata, of whether the value of all words is below 0, is not known to be decidable.

The discounted-sum automata used in [2] are the same as ours, with only syntactic differences – they use the discount-factor $\lambda$ as a multiplying factor, rather than as a dividing one, and define the value of a word as the maximal value of the automaton runs on it, rather than the minimal one. The definitions are analogous, replacing $\lambda$ with $\frac{1}{\lambda}$ and multiplying all weights by $(-1)$. In [2], it is shown that for every rational discount-factor $1 < \lambda < 2$, there is a $\lambda$-NDA that cannot be determinized. We generalize their proof approach, in Theorem 7, extending the result to every $\lambda \in \mathbb{Q} \setminus \mathbb{N}$.

## 2 Discounted-Sum Automata

We consider discounted-sum automata with rational weights and rational discount factors over finite and infinite words.

Formally, given an alphabet $\Sigma$, a *word* over $\Sigma$ is a finite or infinite sequence of letters in $\Sigma$, with $\varepsilon$ for the empty word. We denote the concatenation of a finite word $u$ and a finite or infinite word $w$ by $u{\cdot}w$, or simply by $uw$.

A discounted-sum automaton (NDA) is a tuple $\mathcal{A} = \langle \Sigma, Q, q_{in}, \delta, \gamma, \lambda \rangle$ over a finite alphabet $\Sigma$, with a finite set of states $Q$, an initial state $q_{in} \in Q$, a transition function $\delta \subseteq Q \times \Sigma \times Q$, a weight function $\gamma : \delta \to \mathbb{Q}$, and a discount factor $1 < \lambda \in \mathbb{Q}$. We write $\lambda$-NDA to denote an NDA with a discount factor $\lambda$, for example $\frac{5}{2}$-NDA, and refer to an "integral NDA" when $\lambda$ in an integer. For an automaton $\mathcal{A}$ and a state $q$ of $\mathcal{A}$, we denote by $\mathcal{A}^q$ the automaton that is identical to $\mathcal{A}$, except for having $q$ as its initial state.

Intuitively, $\{q' \mid (q, \sigma, q') \in \delta\}$ is the set of states that $\mathcal{A}$ may move to when it is in the state $q$ and reads the letter $\sigma$. The automaton may have many possible transitions for each state and letter, and hence we say that $\mathcal{A}$ is *nondeterministic*. In the case where for every $q \in Q$ and $\sigma \in \Sigma$, we have that $|\{q' \mid (q, \sigma, q') \in \delta\}| \leq 1$, we say that $\mathcal{A}$ is *deterministic*, denoted DDA.

In the case where for every $q \in Q$ and $\sigma \in \Sigma$, we have that $|\{q' \mid (q, \sigma, q') \in \delta\}| \geq 1$, we say that $\mathcal{A}$ is *complete*. Intuitively, a complete automaton cannot get stuck at some state. In this paper, we only consider complete automata, except for Section 4.2, handling incomplete automata.

A run of an automaton is a sequence of states and letters, $q_0, \sigma_1, q_1, \sigma_2, q_2, \ldots$, such that $q_0 = q_{in}$ and for every $i$, $(q_i, \sigma_{i+1}, q_{i+1}) \in \delta$. The length of a run, denoted $|r|$, is $n$ for a finite run $r = q_0, \sigma_1, q_1, \ldots, \sigma_n, q_n$, and $\infty$ for an infinite run.

The value of a run $r$ is $\gamma(r) = \sum_{i=0}^{|r|-1} \frac{\gamma(q_i, \sigma_{i+1}, q_{i+1})}{\lambda^i}$. The value of a word $w$ (finite or infinite) is $\mathcal{A}(w) = \inf\{\gamma(r) \mid r \text{ is a run of } \mathcal{A} \text{ on } w\}$. A run $r$ of $\mathcal{A}$ on a word $w$ is said to be *optimal* if $\gamma(r) = \mathcal{A}(w)$. By the above definitions, an automaton $\mathcal{A}$ over finite words realizes a function from $\Sigma^*$ to $\mathbb{Q}$ and over infinite words from $\Sigma^\omega$ to $\mathbb{R}$. Two automata, $\mathcal{A}$ and $\mathcal{A}'$, are *equivalent* if they realize the same function.

Next, we provide some specific definitions, to be used in the determinization construction and in the non-determinizability proofs.

The *cost* of reaching a state $q$ of an automaton $\mathcal{A}$ over a finite word $u$ is $\text{cost}(q, u) = \min\{\gamma(r) \mid r \text{ is a run of } \mathcal{A} \text{ on } u \text{ ending in } q\}$, where $\min \emptyset = \infty$. The *gap* of a state $q$ over a finite word $u$ is $\text{gap}(q, u) = \lambda^{|u|}(\text{cost}(q, u) - \mathcal{A}(u))$. Note that when $\mathcal{A}$ operates over infinite words, we interpret $\mathcal{A}(u)$, for a finite word $u$, as if $\mathcal{A}$ was operating over finite words.

Intuitively, the gap of a state $q$ over a word $u$ stands for the weight that a run starting in $q$ should save, compared to a run starting in $u$'s optimal ending state, in order to make $q$'s path preferable. A gap of a state $q$ over a finite word $u$ is said to be *recoverable* if there is a suffix that makes this path optimal; that is, if there is a word $w$, such that

$\text{cost}(q,u) + \frac{\mathcal{A}^q(w)}{\lambda^{|u|}} = \mathcal{A}(uw)$. The suffix $w$ should be finite/infinite, depending on whether $\mathcal{A}$ operates over finite/infinite words.

Notes on notation-conventions: The discount factor $\lambda$ is often used in the literature as a multiplying factor, rather than as a dividing factor, thus taking the role of $\frac{1}{\lambda}$, compared to our definitions. Another convention is to value a word as the maximal value of its possible runs, rather than the minimal value; the two definitions are analogous, and can be interchanged by multiplying all weights by $(-1)$.

## 3 Determinizability of Integral Discounted-Sum Automata

In this section, we show that all complete NDAs with an integral factor are determinizable. Formally, we provide the following result.

▶ **Theorem 1.** *For every complete $\lambda$-NDA $\mathcal{A}$ with an integral factor $\lambda \in \mathbb{N}$, there is an equivalent complete $\lambda$-DDA with up to $m^n$ states, where $m$ is the maximal difference between the weights in $\mathcal{A}$, multiplied by the minimal common divider of all weights, and $n$ is the number of states in $\mathcal{A}$.*

**Proof.** Lemmas 2-4, given in the subsections below, constitute the proof. ◀

Theorem 1 stands for both automata over finite words and over infinite words.

The determinization procedure extends the subset construction, by keeping a recoverable-gap value to each element of the subset. It resembles the determinization procedure of non-discounting sum automata over finite words [11, 7], while having two main differences: the weight-differences between the reachable states is multiplied at every step by $\lambda$, and differences that exceed some threshold are removed.

The procedure may be used for an arbitrary $\lambda$-NDA, always providing an equivalent $\lambda$-DDA, if terminating. It is guaranteed to terminate for a $\lambda$-NDA with $\lambda \in \mathbb{N}$, which is not the case for $\lambda \in \mathbb{Q} \setminus \mathbb{N}$.

The state blow-up involved in the construction is shown to be tight for a rich alphabet of size exponential in the number of states (Theorem 6). The unavoidable blow-up for an alphabet of size linear in the number of states is left as an open problem.

We start, in Subsection 3.1, with the determinization procedure, continue, in Subsection 3.2, with its termination and correctness proofs, and conclude, in Subsection 3.3, showing that the involved state blow-up is tight for a rich alphabet.

### 3.1 The Construction

Consider an NDA $\mathcal{A} = \langle \Sigma, Q, q_{in}, \delta, \gamma, \lambda \rangle$. We inductively construct an equivalent DDA $\mathcal{D} = \langle \Sigma, Q', q'_{in}, \delta', \gamma', \lambda \rangle$. (An example is given in Figure 4.)

Let $T$ be the maximal difference between the weights in $\mathcal{A}$. That is, $T = \max\{|x-y| \bigm| x, y \in \text{range}(\gamma)\}$. Since $\sum_{i=0}^{\infty}(\frac{1}{\lambda^i}) = \frac{1}{1-\frac{1}{\lambda}} = \frac{\lambda}{\lambda-1} \leq 2$, we define the set $G = \{v \bigm| v \in \mathbb{Q} \text{ and } 0 \leq v < 2T\} \cup \{\infty\}$ of possible recoverable-gaps. The $\infty$ element denotes a non-recoverable gap, and behaves as the standard infinity element in the arithmetic operations that we will be using. Note that our discounted-sum automata do not have infinite weights; it is only used as an internal element of the construction.

A state of $\mathcal{D}$ extends the standard subset construction by assigning a gap to each state of $\mathcal{A}$. That is, for $Q = \{q_1, \ldots, q_n\}$, a state $q' \in Q'$ is a tuple $\langle g_1, \ldots, g_n \rangle$, where $g_h \in G$ for every $1 \leq h \leq n$. Intuitively, the gap $g_h$ of a state $q_h$ stands for the extra cost of reaching $q_h$, compared to the best possible value so far. This extra cost is multiplied, however, by $\lambda^l$, for a finite run of length $l$, to reflect the $\lambda^l$ division in the value-computation of the suffixes. Once a gap is obviously irreducible, by being larger than or equal to $2T$, it is set to be $\infty$.

In the case that $\lambda \in \mathbb{N}$, the construction only requires finitely many elements of $G$, as shown in Lemma 2 below, and thus it is guaranteed to terminate.

For simplicity, we assume that $q_{in} = q_1$ and extend $\gamma$ with $\gamma(\langle q_i, \sigma, q_j\rangle) = \infty$ for every $\langle q_i, \sigma, q_j\rangle \notin \delta$. The initial state of $\mathcal{D}$ is $q'_{in} = \langle 0, \infty, \ldots, \infty\rangle$, meaning that $q_{in}$ is the only relevant state and has a 0 gap.

We inductively build $\mathcal{D}$ via the intermediate automata $\mathcal{D}_i = \langle \Sigma, Q'_i, q'_{in}, \delta'_i, \gamma'_i, \lambda\rangle$. We start with $\mathcal{D}_1$, in which $Q'_1 = \{q'_{in}\}$, $\delta'_1 = \emptyset$ and $\gamma'_1 = \emptyset$, and proceed from $\mathcal{D}_i$ to $\mathcal{D}_{i+1}$, such that $Q'_i \subseteq Q'_{i+1}$, $\delta'_i \subseteq \delta'_{i+1}$ and $\gamma'_i \subseteq \gamma'_{i+1}$. The construction is completed once $\mathcal{D}_i = \mathcal{D}_{i+1}$, finalizing the desired deterministic automaton $\mathcal{D} = \mathcal{D}_i$.

In the induction step, $\mathcal{D}_{i+1}$ extends $\mathcal{D}_i$ by (possibly) adding, for every state $q' = \langle g_1, \ldots, g_n\rangle \in Q'_i$ and letter $\sigma \in \Sigma$, a state $q''$, a transition $\langle q', \sigma, q''\rangle$ and a weight $\gamma_{i+1}(\langle q', \sigma, q''\rangle) = c$, as follows:

- For every $1 \le h \le n$, $c_h := \min\{g_j + \gamma(\langle q_j, \sigma, q_h\rangle) \mid 1 \le j \le n\}$
- $c := \min_{1 \le h \le n}(c_h)$
- For every $1 \le h \le n$,
  - $x_h := \lambda(c_h - c)$;
  - if $x_h \ge 2T$ then $x_h := \infty$
- $q'' := \langle x_1, \ldots, x_n\rangle$
- $Q'_{i+1} := Q'_{i+1} \cup q''$
- $\delta'_{i+1} := \delta'_{i+1} \cup \langle q', \sigma, q''\rangle$
- $\gamma'_{i+1}(\langle q', \sigma, q''\rangle) = c$

## 3.2 Termination and Correctness

We prove below that the above procedure always terminates for a discount factor $\lambda \in \mathbb{N}$, while generating an automaton that is equivalent to the original one. We start with the termination proof.

▶ **Lemma 2.** *The above determinization procedure always terminates for a complete integral $\lambda$-NDA $\mathcal{A}$. The resulting deterministic automaton has up to $m^n$ states, where $m$ is the maximal difference between the weights in $\mathcal{A}$, multiplied by the minimal common divider of all weights, and $n$ is the number of states in $\mathcal{A}$.*

**Proof.** The induction step of the construction, extending $\mathcal{D}_i$ to $\mathcal{D}_{i+1}$, only depends on $\mathcal{A}$, $\Sigma$ and $Q'_i$. Furthermore, for every $i \ge 0$, we have that $Q'_i \subseteq Q'_{i+1}$. Thus, for showing the termination of the construction, it is enough to show that there is a general bound on the size of the sets $Q'_i$. We do it by showing that the inner values, $g_1, \ldots, g_n$, of every state $q'$ of every set $Q'_i$ are from the finite set $\bar{G}$, defined below.

Let $d \in \mathbb{N}$ be the minimal common divider of the weights in $\mathcal{A}$, and let $m \in \mathbb{N}$ be the maximal difference between the weights, multiplied by $d$. That is, $m = d \times \max\{|x - y| \mid x, y \in \text{range}(\gamma)\}$. We define the set $\bar{G} = \{\frac{\lambda c}{d} \mid \frac{2m}{\lambda} > c \in \mathbb{N}\} \cup \{\infty\}$

We start with $Q'_1$, which satisfies the property that the inner values, $g_1, \ldots, g_n$, of every state $q' \in Q'_1$ are from $\bar{G}$, as $Q'_1 = \{\langle 0, \infty, \ldots, \infty\rangle\}$. We proceed by induction on the construction steps, assuming that $Q'_i$ satisfies the property. By the construction, an inner value of a state $q''$ of $Q'_{i+1}$ is derived by four operations on elements of $\bar{G}$: addition, subtraction ($x - y$, where $x \ge y$), multiplication by $\lambda \in \mathbb{N}$, and minimization.

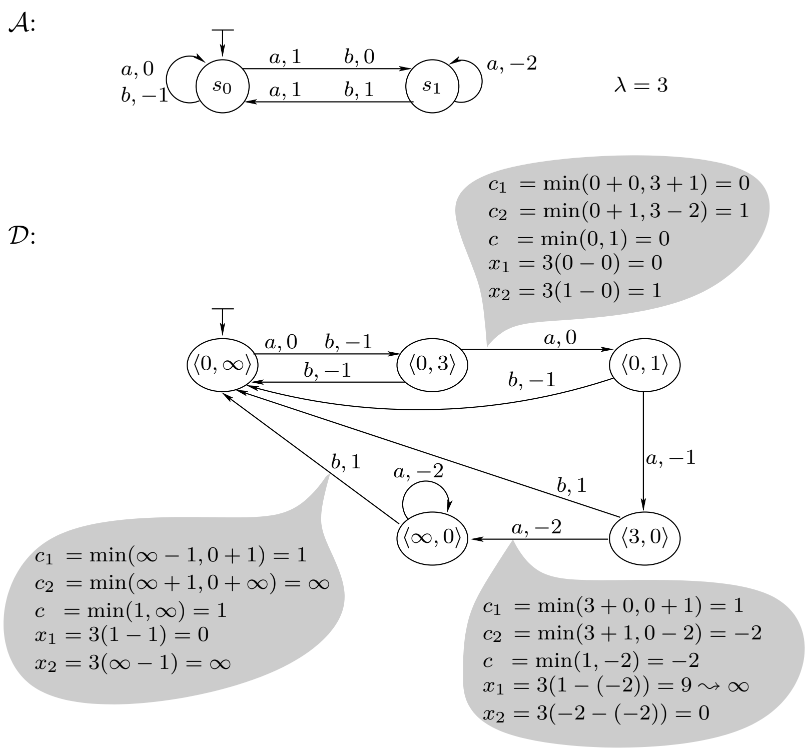


**Figure 1** Determinizing the 3-NDA $\mathcal{A}$ into the 3-DDA $\mathcal{D}$. The gray bubbles detail some of the intermediate calculations of the determinization procedure.

One may verify that applying these four operations on $\infty$ and numbers of the form $\frac{\lambda c}{d}$, where $\lambda, c \in \mathbb{N}$, results in $\infty$ or in a number $\frac{v}{d}$, where $v \in \mathbb{N}$. Since the last operation in calculating an inner value of $q''$ is multiplication by $\lambda$, we have that $v$ is divided by $\lambda$. Once an inner value exceeds $\frac{2m}{d}$, it is replaced with $\infty$. Hence, all the inner values are in $\bar{G}$.

Having up to $m$ possible values to the elements of an $n$-tuple, provides the $m^n$ upper bound for the state space of the resulting deterministic automaton. ◀

Before proceeding to the correctness proof, we show that equivalence of automata over finite words implies their equivalence over infinite words. Note that the converse need not hold.

▶ **Lemma 3.** *If two NDAs, $\mathcal{A}$ and $\mathcal{B}$, are equivalent with respect to finite words then they are also equivalent with respect to infinite words. The converse need not hold.*

**Proof.** Assume, by contradiction, two NDAs, $\mathcal{A}$ and $\mathcal{B}$, that are equivalent with respect to finite words and not equivalent with respect to infinite words. Then there is an infinite word $w$ and a constant number $c \neq 0$, such that $\mathcal{A}(w) - \mathcal{B}(w) = c$. Let $m$ be the maximal difference between a weight in $\mathcal{A}$ and a weight in $\mathcal{B}$. Since for every $1 < \lambda$, $\sum_{i=0}^{\infty}(\frac{1}{\lambda^i}) = \frac{1}{1-\frac{1}{\lambda}} = \frac{\lambda}{\lambda-1} \leq 2$, it follows that the difference between the values that $\mathcal{A}$ and $\mathcal{B}$ assign to any word is smaller or equal to $2m$. Hence, the difference between the values of their runs on suffixes of $w$, starting at a position $p$, is smaller or equal to $\frac{2m}{\lambda^p}$.

Now, since $\mathcal{A}$ and $\mathcal{B}$ are equivalent over finite words, it follows that they have equally-valued optimal runs over every prefix of $w$. Thus, after a long enough prefix, of length $p$ such that $\frac{2m}{\lambda^p} < c$, the difference between the values of $\mathcal{A}$'s and $\mathcal{B}$'s optimal runs on $w$ must be smaller than $c$, leading to a contradiction.

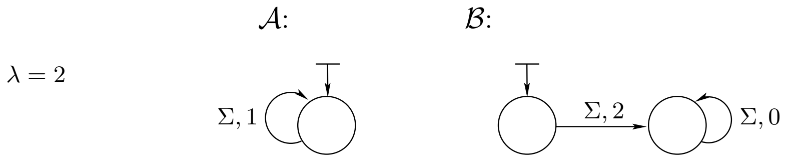


**Figure 2** The automata $\mathcal{A}$ and $\mathcal{B}$ are equivalent with respect to infinite words, while not equivalent with respect to finite words.

A counter example for the converse is provided in Figure 2. ◀

We proceed with the correctness proof. By Lemma 3, it is enough to prove the correctness for automata over finite words.

Note that the correctness holds for arbitrary discount factors, not only for integral ones. For the latter, the determinization procedure is guaranteed to terminate (Lemma 2), which is not the case in general. Yet, in all cases that the procedure terminates, it is guaranteed to be correct.

▶ **Lemma 4.** *Consider a $\lambda$-NDA $\mathcal{A}$ over $\Sigma^*$ and a DDA $\mathcal{D}$, constructed from $\mathcal{A}$ as above. Then, for every $w \in \Sigma^*$, $\mathcal{A}(w) = \mathcal{D}(w)$.*

**Proof.** Consider an NDA $\mathcal{A} = \langle \Sigma, Q, q_{in}, \delta, \gamma, \lambda \rangle$ and the DDA $\mathcal{D} = \langle \Sigma, Q', q'_{in}, \delta', \gamma', \lambda \rangle$ constructed from $\mathcal{A}$ as above. Let $T$ be the maximal difference between the weights in $\mathcal{A}$. That is, $T = \max\{|x - y| \mid x, y \in \text{range}(\gamma)\}$.

For a word $w$, let $q'_w = \langle g_1, \ldots, g_n \rangle \in Q'$ be the last state of $\mathcal{D}$'s run on $w$. We show by induction on the length of the input word $w$ that:

- For every $1 \leq h \leq n$, $g_h = \text{gap}(q_h, w)$ if $\text{gap}(q_h, w) < 2T$ and $\infty$ otherwise.
- $\mathcal{A}(w) = \mathcal{D}(w)$.

The assumptions obviously hold for the initial step, where $w$ is the empty word. As for the induction step, we assume they hold for $w$ and show that for every $\sigma \in \Sigma$, they hold for $w{\cdot}\sigma$. Let $q'_{w\sigma} = \langle x_1, \ldots, x_n \rangle \in Q'$ be the last state of $\mathcal{D}$'s run on $w{\cdot}\sigma$.

For every $1 \leq h \leq n$, as long as $g_h < 2T$, the value that the determinization-construction assigns to $x_h$, as well as the weight that is set on the transition from $q'_w$ to $q'_{w\sigma}$, directly follows the gap definitions, and accordingly satisfy the required properties. Therefore, it is left to show that if $g_h \geq 2T$ then $\text{gap}(q_h, w\sigma) \geq 2T$. Indeed, $\text{gap}(q_h, w\sigma) = \lambda^{i+1}(\text{cost}(q_h, w\sigma) - \mathcal{A}(w\sigma)) > \lambda^{i+1}(\text{cost}(q_h, w) - \mathcal{A}(w) - (\frac{1}{\lambda}^{i})T) > \lambda(2T - T) = \lambda(T) \geq 2T$.

◀

## 3.3 State Complexity

For an integral NDA $\mathcal{A}$, the deterministic automaton constructed as in Subsection 3.1 has up to $m^n$ states, where $m$ is the maximal difference between the weights in $\mathcal{A}$, multiplied by the minimal common divider of all weights, and $n$ is the number of states in $\mathcal{A}$ (Lemma 2).

We show below that the above state blow-up is asymptotically tight, using a rich alphabet of size in $O(m^n)$. For an alphabet of size linear in $m$ and $n$, the unavoidable state blow-up is left as an open problem.

A family of automata $\mathcal{A}_{m,n}$, with which we provide the lower bound, is illustrated in Figure 3. Intuitively, the rich alphabet allows to set every gap in $\{0, 1, 2, \ldots, m+1\}$ to each of the $n$ states. Two different gaps have, under the appropriate setting, two suffixes that distinguish between them. Hence, an equivalent deterministic automaton must have a unique state for each recoverable-gap, yielding at least $m^n$ states.

We start by providing a sufficient condition, under which two different gaps must be associated with two different states of a deterministic automaton. The lemma below generalizes an argument given in [2].

▶ **Lemma 5.** *Consider an NDA $\mathcal{A}$ for which there is an equivalent DDA $\mathcal{D}$. If there is a state $q$ of $\mathcal{A}$, finite words $u$ and $u'$, and words $w$ and $z$, such that:*

*i.* *$\mathcal{A}$ has runs on $u$ and on $u'$ ending in $q$;*

*ii.* $\text{gap}(q,u) \neq \text{gap}(q,u')$*;*

*iii.* *The gaps of $q$ over both $u$ and $u'$ are recoverable with $w$, that is, $\mathcal{A}(uw) = \text{cost}(q,u)+\frac{\mathcal{A}^q(w)}{\lambda^{|u|}}$ and $\mathcal{A}(u'w) = \text{cost}(q,u') + \frac{\mathcal{A}^q(w)}{\lambda^{|u'|}}$; and*

*iv.* *$\mathcal{A}$ is indifferent to concatenating $z$ to $u$ and to $u'$, that is $\mathcal{A}(uz) = \mathcal{A}(u)$ and $\mathcal{A}(u'z) = \mathcal{A}(u')$*

*then the runs of $\mathcal{D}$ on $u$ and on $u'$ end in different states.*

*The words $w$ and $z$ should be finite for automata over finite words and infinite for automata over infinite words. In the former case, $z$ is redundant as it can always be $\varepsilon$.*

**Proof.** Consider the above setting. Then, we have that $\mathcal{A}(uw) - \mathcal{A}(uz) = \frac{\text{gap}(q,u)+\mathcal{A}^q(w)}{\lambda^{|u|}}$ and $\mathcal{A}(u'w) - \mathcal{A}(u'z) = \frac{\text{gap}(q,u')+\mathcal{A}^q(w)}{\lambda^{|u'|}}$ . Thus,

$$(I)\;\; \text{gap}(q,u) = \lambda^{|u|}[\mathcal{A}(uw) - \mathcal{A}(uz)] - \mathcal{A}^q(w); \quad \text{gap}(q,u') = \lambda^{|u'|}[\mathcal{A}(u'w) - \mathcal{A}(u'z)] - \mathcal{A}^q(w)$$

Now, assume, by contradiction, a single state $p$ of $\mathcal{D}$ in which the runs of $\mathcal{D}$ on both $u$ and $u'$ end. Then, we have that

$$(II)\;\; \mathcal{D}(uw) - \mathcal{D}(uz) = \frac{\mathcal{D}^p(w)}{\lambda^{|u|}}; \quad \mathcal{D}(u'w) - \mathcal{D}(u'z) = \frac{\mathcal{D}^p(w)}{\lambda^{|u'|}}$$

Since $\mathcal{A}$ and $\mathcal{D}$ are equivalent, we may replace between $[\mathcal{A}(uw)-\mathcal{A}(uz)]$ and $[\mathcal{D}(uw)-\mathcal{D}(uz)]$ as well as between $[\mathcal{A}(u'w) - \mathcal{A}(uz')]$ and $[\mathcal{D}(u'w) - \mathcal{D}(u'z)]$. Making the replacements in equations (I) above, we get:

$$(I\&II)\;\; \text{gap}(q,u) = \lambda^{|u|}\frac{\mathcal{D}^p(w)}{\lambda^{|u|}} - \mathcal{A}^q(w); \quad \text{gap}(q,u') = \lambda^{|u'|}\frac{\mathcal{D}^p(w)}{\lambda^{|u'|}} - \mathcal{A}^q(w)$$

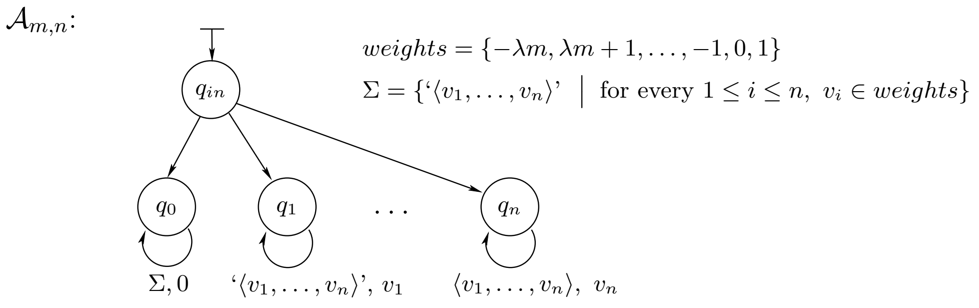
Therefore, $\text{gap}(q,u) = \text{gap}(q,u')$, leading to a contradiction. ◀

We continue with the tightness proof.

▶ **Theorem 6.** *For every $\lambda, m, n \in \mathbb{N}$, there is a complete $\lambda$-NDA with $n+2$ states and weights in $\{-\lambda m, -\lambda m+1, \ldots, -1, 0, 1\}$, such that every equivalent DDA has at least $m^n$ states.*

**Proof.** For every $\lambda, m, n \in \mathbb{N}$, we define the NDA $\mathcal{A} = \langle \Sigma, Q, q_{in}, \delta, \gamma, \lambda\rangle$, illustrated in Figure 3 as $\mathcal{A}_{m,n}$, where:

- $\Sigma = \{\langle v_1, \ldots v_n\rangle \;\big|\; \text{for every } 1 \leq i \leq n, v_i \in \{-\lambda m, -\lambda m+1, \ldots, -1, 0, 1\}\}$
- $Q = \{q_{in}, q_0, q_1, \ldots, q_n\}$
- $\delta = \{\langle q_{in}, \sigma, q_i\rangle, \langle q_i, \sigma, q_i\rangle \;\big|\; 0 \leq i \leq n \text{ and } \sigma \in \Sigma\}$
- For every $\sigma = \langle v_1, \ldots v_n\rangle \in \Sigma$ and $0 \leq i \leq n$: $\gamma(\langle q_{in}, \sigma, q_0\rangle) = 0$, $\gamma(\langle q_{in}, \sigma, q_i\rangle) = 0$, $\gamma(\langle q_0, \sigma, q_0\rangle) = 0$ and $\gamma(\langle q_i, \sigma, q_i\rangle) = v_i$



$\mathcal{A}_{m,n}$:
$weights = \{-\lambda m, \lambda m + 1, \ldots, -1, 0, 1\}$
$\Sigma = \{`\langle v_1, \ldots, v_n \rangle' \mid \text{for every } 1 \le i \le n,\ v_i \in weights\}$
$q_{in}$
$q_0$
$q_1$
$q_n$
$\Sigma, 0$
$`\langle v_1, \ldots, v_n \rangle', v_1$
$\langle v_1, \ldots, v_n \rangle,\ v_n$

**Figure 3** The family of integral NDAs, where for every $m$ and $n$, a deterministic automaton equivalent to $\mathcal{A}_{m,n}$ must have at least $m^n$ states.

Note that, for simplicity, we define the alphabet letters of $\Sigma$ as tuples of numbers.

Consider a DDA $\mathcal{D}$ equivalent to $\mathcal{A}$. We will show that there is a surjective mapping between $\mathcal{D}$'s states and the set of vectors $V = \{\langle g_1, \ldots, g_n \rangle \mid \text{for every } 1 \le i \le n, 1 \le g_i \le m\}$.

We call an $n$-vector of gaps, $G = \langle g_1, \ldots, g_n \rangle$, a *combined-gap*, specifying the gaps of $q_1, \ldots, q_n$, respectively. Due to the rich alphabet, for every combined-gap $G \in V$, there is a finite word $u_G$, such that for every $1 \le i \le n$, $\text{gap}(q_i, u_G) = g_i$.

Every two different combined gaps, $G$ and $G'$, are different in at least one dimension $j$ of their $n$-vectors. Thus, $\mathcal{A}$ satisfies the conditions of Lemma 5, by having $u = u_G$, $u' = u_{G'}$, $z =`\langle 0, \ldots, 0 \rangle'^{\omega}$, and $w =`\langle 0, \ldots 0, -\lambda m, 0, \ldots 0 \rangle'^{\omega}$, where the repeated word in $w$ has 0 in all dimensions except for $-\lambda m$ in the $j$'s dimension. Hence, $\mathcal{A}$ has two different states corresponding to each two different vectors in $V$, and we are done.

◀

## 4 Nondeterminizability of Nonintegral Discounted-Sum Automata

The discount-factor $\lambda$ plays a key role in the question of whether a complete $\lambda$-NDA is determinizable. In Section 3, we have shown that an integral factor guarantees the automaton's determinizabilty. In Subsection 4.1 below, we show the converse for every nonintegral factor.

In the whole paper, except for Subsection 4.2 below, we only consider complete automata. In Subsection 4.2, we show that once allowing incomplete automata or, equivalently, adding infinite weights, there is a non-determinizable automaton for every discount-factor $\lambda$, including integral ones.

### 4.1 Complete Automata

We show below that for every noninntegral discount factor $\lambda$, there is a complete $\lambda$-NDA that cannot be determinized. The proof generalizes the approach taken in [2], where the case of $1 < \lambda < 2$ was handled.

Intuitively, for a discount factor that is not a whole number, a nondeterministic automaton might have arbitrarily dense recoverable-gaps. Two different gaps have, under the appropriate setting, two suffixes that distinguish between them (Lemma 5). Hence, an equivalent deterministic automaton must have a unique state for each recoverable-gap, which is impossible for infinitely many gaps.

▶ **Theorem 7.** *For every nonintegral discount factor $\lambda$, there is a complete $\lambda$-NDA for which there is no equivalent DDA (with any discount factor).*

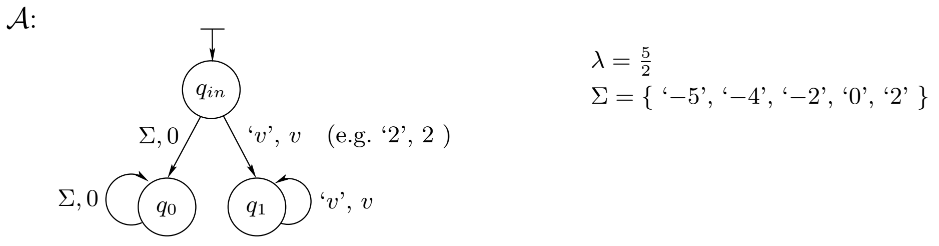


**Figure 4** The non-determinizable $\frac{5}{2}$-NDA $\mathcal{A}$.

**Proof.** For every $1 < \lambda \in \mathbb{Q} \setminus \mathbb{N}$, we define a complete $\lambda$-NDA $\mathcal{A} = \langle \Sigma, Q, q_{in}, \delta, \gamma, \lambda \rangle$ and show that $\mathcal{A}$ is not determinizable. Let $\lambda = \frac{h}{k}$, where $h$ and $k$ are mutually prime, and define:

- $\Sigma = \{-jk \mid j \in \mathbb{N} \text{ and } jk < h\} \cup \{-h, k\}$
- $Q = \{q_{in}, q_1, q_2\}$
- $\delta = \{\langle q_{in}, \sigma, q_1\rangle, \langle q_{in}, \sigma, q_2\rangle, \langle q_1, \sigma, q_1\rangle, \langle q_2, \sigma, q_2\rangle \mid \sigma \in \Sigma\}$
- For every $\sigma \in \Sigma$ and $q \in Q$: $\gamma(\langle q, \sigma, q_1\rangle) = 0$ and $\gamma(\langle q, \sigma, q_2\rangle) = \sigma$

Note that, for simplicity, we define the alphabet letters of $\Sigma$ as numbers. The NDA $\mathcal{A}$ for $\lambda = \frac{5}{2}$ is illustrated in Figure 1.

We show that $\mathcal{A}$ cannot be determinized by providing an infinite word $w$, such that $q_2$ has a unique recoverable gap for each of $w$'s prefixes. By Lemma 5, such a word $w$ implies that $\mathcal{A}$ cannot be determinized.

We inductively define $w$, denoting its prefix of of length $i$ by $w_i$, as follows: the first letter is $k$ and the $i+1$'s letter is '$-jk$', such that $0 \leq \text{gap}(q_2, w_i)\frac{h}{k} - jk \leq k$. Intuitively, each letter is chosen to almost compensate on the gap generated so far, by having the same value as the gap up to a difference of $k$.

We show that $w$ has the required properties, by the following steps:

1. The word $w$ is infinite and $q_2$ has a recoverable-gap for each of its prefixes.
2. There is no prefix of $w$ for which $q_2$'s gap is 0.
3. There are no two different prefixes of $w$ for which $q_2$ has the same gap.

Indeed:

1. Since $\gamma(\langle q_2, -h, q_2\rangle) = -h$, a gap $g$ of $q_2$ is obviously recoverable if $g \leq h$. We show by induction on the length of $w$'s prefixes that for every $i \geq 1$, we have that $\text{gap}(q_2, w_i) \leq h$. It obviously holds for the initial step, as $w_1 =$'$k$' and $\text{gap}(q_2, w_1) = k\frac{h}{k} = h$. Assuming that it holds for the $i$'s prefix, we can choose the $i+1$'s letter to be some '$-jk$' $\in \Sigma$, such that $0 \leq \text{gap}(w_i) - jk \leq k$. Hence, we get that $\text{gap}(w_{i+1}) = (\text{gap}(w_i) - jk)\frac{h}{k} \leq k$.
2. Assume, by contradiction, a prefix of $w$ of length $n+1$ whose recoverable-gap is 0. We have then that:
$$(((k\frac{h}{k} - j_1k)\frac{h}{k} - j_2k)\frac{h}{k} \ldots - j_nk)\frac{h}{k} = 0$$
for some $j_1, \ldots, j_n \in \mathbb{N}$. Simplifying the equation, we get that
$$\frac{h^n - j_1kh^{n-1} - j_2k^2h^{n-2} - \ldots - j_nk^n}{k^{n-1}} = 0$$
Therefore, $h^n = j_1kh^{n-1} + \ldots + j_nk^n$. Now, since $k$ divides $j_1kh^{n-1} + \ldots + j_nk^n$, it follows that $k$ divides $h^n$, which leads to a contradiction, as $h$ and $k$ are mutually prime.
3. Assume, by contradiction, that $q_2$ has the same gap $x$ for two prefixes, $n \geq 1$ steps apart. We have then that:
$$((((x - j_1k)\frac{h}{k} - j_2k)\frac{h}{k} - j_3k)\frac{h}{k} \ldots - j_nk)\frac{h}{k} = x$$

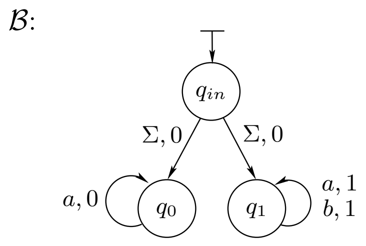


**Figure 5** The incomplete automaton $\mathcal{B}$ is not determinizable with respect to any discount-factor.

for some $j_1, \ldots, j_n \in \mathbb{N}$. Simplifying the equation, we get that

$$\frac{xh^n - j_1 kh^n - j_2 k^2 h^{n-1} - \ldots - j_n k^n h}{k^n} = x$$

Thus,

$$xh^n - xk^n = j_1 kh^n + j_2 k^2 h^{n-1} + \ldots + j_n k^n h$$

Hence, $k$ divides $x(h^n - k^n)$. Now, since there is no prefix for which $q_2$ has a zero gap, it follows that $k$ does not divide $x$. (Otherwise, $q_2$ would have had a zero gap for the prefix right after the one with $x$). Therefore, $k$ divides $h^n - k^n$. But, since $k$ divides $k^n$, it follows that $k$ also divides $h^n$, which leads to a contradiction.

◀

## 4.2 Incomplete Automata

Once considering incomplete automata or, equivalently, automata with $\infty$-weights, no discount factor can guarantee determinization. The reason is that there is no threshold above which a gap becomes irrecoverable – no matter how (finitely) bad some path is, it might eventually be essential, in the case that the other paths get stuck.

Formally:

▶ **Theorem 8.** *For every rational discount factor $\lambda$, there is an incomplete $\lambda$-NDA for which there is no equivalent DDA (with any discount factor).*

**Proof.** Consider the incomplete automaton $\mathcal{B}$ presented in Figure 5 with a discount factor $\lambda \in \mathbb{Q}$.

For every $n \in \mathbb{N}$, we have that $\text{gap}(q_2, a^n) = \sum_{i=0}^{n} \lambda^i$. Since $q_1$ has no transition for the letter $b$, it follows that all these gaps are recoverable. Hence, for every $i, j \in \mathbb{N}$ such that $i \neq j$, we satisfy the conditions of Lemma 5 with $u = a^i$, $u' = a^j$, $z = a^\omega$ and $w = b^\omega$ (for automata over finite words, $z = \varepsilon$ and $w = b$). Therefore, an equivalent deterministic automaton must have infinitely many states, precluding its existence. ◀

## 5 Closure Properties

Discounted-sum automata realize a function from words to numbers. Hence, one may wish to consider their closure under arithmetic operations. The operations are either between two automata, having the same discount factor, as minimization and addition, or between an automaton and a scalar, as multiplication by a positive rational number $c$.

We consider the class of complete NDAs, as well as two of its subclasses: DDAs and integral NDAs.

| Class $\diagdown$ Operation | min | max | $+$ | $-$ | $\times c \geq 0$ | $\times(-1)$ |
|---|---|---|---|---|---|---|
| NDAs | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ |
| DDAs | ✗ | | ✓ | | | |
| Integral NDAs | ✓ | | | | | |

**Table 1** Closure of discounted-sum automata under arithmetic operations.

The closure properties, summarized in Table 1, turn out to be the same for automata over finite words and over infinite-words. By arguments similar to those of Lemma 3's proof, it is enough to prove the positive results with respect to automata over finite words and the negative results with respect to automata over infinite words.

Some of the positive results are straightforward, as follows.

- Nondeterministic: Minimization is achieved by taking the union of the input automata, addition by taking the product of the input automata and adding the corresponding weights, and multiplication by a positive scalar $c$ is achieved by multiplying all weights by $c$.
- Deterministic: Addition/subtraction is achieved by taking the product of the input automata and adding/subtracting the corresponding weights. Multiplication by (positive or negative) scalar $c$ is achieved by multiplying all weights by $c$.
- Discount factor $\in \mathbb{N}$: Since these automata can always be determinized, they obviously enjoy the closure properties of both the deterministic and non-deterministic classes.

All the negative results can be reduced to the max operation, as follows. Closure under subtraction implies closure under $(-1)$-multiplication, by subtracting the given automaton from a constant 0 automaton. For nondeterministic automata, closure under $(-1)$-multiplication implies closure under the max operation, by multiplying the original automata by $(-1)$ and taking their minimum. As for deterministic automata, closure under the min and max operations are reducible to each other due to the closure under $(-1)$-multiplication.

It is left to show the results with respect to the max operation. We start with the classes of deterministic and nondeterministic automata.

▶ **Theorem 9.** *NDAs and DDAs are not closed under the* max *operation.*

**Proof.** We prove a stronger claim, showing that there are two DDAs, $\mathcal{A}$ and $\mathcal{B}$, defined in Figure 6, for which there is no NDA equivalent to $\max(\mathcal{A}, \mathcal{B})$. Intuitively, we show that the recoverable-gap between $\mathcal{A}$ and $\mathcal{B}$ can be arbitrarily small, and therefore, by pumping-arguments, an NDA for $\max(\mathcal{A}, \mathcal{B})$ cannot be of a finite size.

Assume, by contradiction, an NDA $\mathcal{C}$ with $n$ states equivalent to $\max(\mathcal{A}, \mathcal{B})$. The value of $\mathcal{A}$ over every word is obviously 0. Thus, for every infinite word $w$, $\mathcal{C}(w) = \mathcal{B}(w)$ if $\mathcal{B}(w) > 0$ and 0 otherwise.

For a finite word $u$, we shall refer to $\lambda^{|u|}\mathcal{B}(u)$ as the *gap* of $\mathcal{B}$ over $u$, denoted $\text{gap}(\mathcal{B}, u)$. Intuitively, this gap stands for the weight that $\mathcal{B}$ should save over a suffix $z$ for having a negative value over the whole word. That is, $\mathcal{B}(uz) < 0$ if and only if $\mathcal{B}(z) < -\text{gap}(\mathcal{B}, u)$. Within this proof, $\lambda$ is fixed to $\frac{5}{2}$.

A key observation is that the gap of $\mathcal{B}$ can be arbitrarily small. Specifically, we show that for every natural numbers $k \geq 3$ and $j \leq \lceil \frac{2^k}{5} \rceil$, there is a finite word $u_{j,k}$ such that $\text{gap}(\mathcal{B}, u_{j,k}) = \frac{5j}{2^k}$. It goes by induction on $k$. For $k = 3$, it holds with $u_{0,3} =$'0', $u_{1,3} =$'$\frac{2}{5}$' '$-\frac{1}{2}$' '$-1$', and $u_{2,3} =$'$\frac{2}{5}$' '$-\frac{1}{2}$'. As for the induction step, consider a number $0 \leq j \leq \lceil \frac{2^{k+1}}{5} \rceil$.

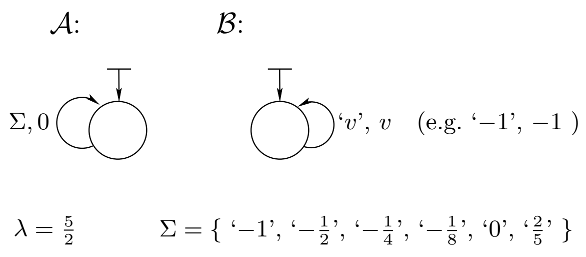

$\mathcal{A}$:
$\mathcal{B}$:
$\Sigma, 0$
'$v$', $v$ (e.g. '$-1$', $-1$ )
$\lambda = \frac{5}{2}$
$\Sigma = \{$ '$-1$', '$-\frac{1}{2}$', '$-\frac{1}{4}$', '$-\frac{1}{8}$', '0', '$\frac{2}{5}$' $\}$

**Figure 6** The DDAs $\mathcal{A}$ and $\mathcal{B}$, for which there is no NDA equivalent to $\max(\mathcal{A}, \mathcal{B})$.

One may verify that multiplying $2^k$ by each of $1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$, and 0, provides a different reminder when divided by 5. Hence, there is a number $v \in \{-1, -\frac{1}{2}, -\frac{1}{4}, -\frac{1}{8}, 0\}$ and a natural number $j' \leq \lceil \frac{2^k}{5} \rceil$ such that $j' = \frac{j - v2^k}{5}$. Thus, we can have, by the induction assumption, the required word $u_{j,k+1}$, by $u_{j,k+1} = u_{j',k}$'$v$' , as $\text{gap}(\mathcal{B}, u_{j,k+1}) = \frac{5}{2}(\text{gap}(\mathcal{B}, u_{j',k}) + v) = \frac{5}{2}(\frac{5j'}{2^k} + v) = \frac{5}{2}(\frac{j-v2^k}{2^k} + v) = \frac{5j}{2^{k+1}}$.

By the above observation, there is a finite word $u$, such that $\frac{1}{\lambda^{2n}} < \text{gap}(\mathcal{B}, u) < \frac{1}{\lambda^n}$. We define the infinite word $w = u$'0'$^n$'$-1$''0'$^\omega$. Since a 0-weighted letter multiplies the gap by $\lambda$, we get that $0 < \text{gap}(\mathcal{B}, u\text{'0'}^n) < 1$, and therefore $\mathcal{B}(w) < 0$ and $\mathcal{C}(w) = 0$.

Let $r$ be an optimal run of $\mathcal{C}$ on $z$. Since $\mathcal{C}$ has only $n$ states, there is a state $q$ of $\mathcal{C}$ and two positions $|u| < p_1 < p_2 < |u| + n$, such that $r$ visits $q$ on both $p_1$ and $p_2$. Let $w_1$ and $w_2$ be the prefixes of $w$ of lengths $p_1$ and $p_2$, respectively. Let $z$ be the suffix of $w$ after $w_2$, that is $w = w_2 z$. Let $r_1$, $r_2$ and $r_z$ be the portions of $r$ on $w_1$, $w_2$ and $z$, respectively.

Let $v_1$ and $v_2$ be the values of $r_1$ and $r_2$, respectively, and define $g_1 = \lambda^{p1} v_1$ and $g_2 = \lambda^{p2} v_2$. Let $v_z$ be the value of a run equivalent to $r_z$. Since the value of $r$ is 0, we have that $v_2 + \frac{v_z}{\lambda^{p2}} = 0$, and therefore, $v_z = -g_2$.

We shall reach a contradiction by showing that $g_1 \not< g_2$, $g_1 \not> g_2$, and $g_1 \neq g_2$. Indeed:

- If $g_1 < g_2$ then there is a run $r' = r_1 r_z$ of $\mathcal{C}$ on the word $w' = w_1 z$, whose value is $v_1 + \frac{v_z}{\lambda^{p2}} = \frac{g_1 + v_z}{\lambda^{p2}}$. However, since $g_1 < g_2 = -v_z$, it follows that the value of $\mathcal{C}$ on $w'$ is negative, which leads to a contradiction.
- If $g_1 > g_2$ then there is a negative-valued run of $\mathcal{C}$ on the word $w_1$'0'$^{2(p_2 - p_1)} z$, analogously to the previous case.
- If $g_1 = g_2$ then there is a 0-valued run of $\mathcal{C}$ on the word $w' = w_1$'0'$^{2n} z$, however $\mathcal{B}(w') > 0$, leading to a contradiction.

◀

We continue with the class of automata with an integral factor.

▶ **Theorem 10.** *For every $\lambda \in \mathbb{N}$, the class of $\lambda$-NDAs is closed under the* max *operation.*

**Proof.** Consider a discount-factor $1 < \lambda \in \mathbb{N}$ and two $\lambda$-NDAs, $\mathcal{A}$ and $\mathcal{B}$. By Theorem 1, $\mathcal{A}$ and $\mathcal{B}$ can be determinized to equivalent $\lambda$-DDAs. Thus, we may only consider deterministic automata. Since deterministic automata are closed under $(-1)$-multiplication, we may also consider the min operation rather than the max operation.

The construction of a DDA $\mathcal{C}$ equivalent to $\min(\mathcal{A}, \mathcal{B})$ is analogous to the determinization construction of Section 3.1, with the difference of extending automata-product rather than the subset-construction. Namely, we iteratively construct the product of $\mathcal{A}$ and $\mathcal{B}$, where a state of $\mathcal{C}$ contains a state of $\mathcal{A}$ and a state of $\mathcal{B}$, together with their recoverable-gaps. That is, for a state $p$ of $\mathcal{A}$ and a state $q$ of $\mathcal{B}$, a state $c$ of $\mathcal{C}$ is of the form $c = \langle \langle p, g_p \rangle, \langle q, g_q \rangle \rangle$. When $\mathcal{A}$ and $\mathcal{B}$ read a finite word $u$ and reach the states $p$ and $q$, respectively, we have that

$g_p = \lambda^{|u|}(\mathcal{A}(u) - \min(\mathcal{A}(u), \mathcal{B}(u)))$ and $g_q = \lambda^{|u|}(\mathcal{B}(u) - \min(\mathcal{A}(u), \mathcal{B}(u)))$. Once a gap is too large, meaning that it is bigger than twice the maximal difference between a weight in $\mathcal{A}$ and a weight in $\mathcal{B}$, it is changed to $\infty$.

The termination and correctness proofs of the above construction are analogous to the proofs of Lemmas 2 and 4. ◀

## 6 Conclusions

Recently, there has been a considerable effort to extend formal verification from the Boolean setting to a quantitative one. Automata theory plays a key role in formal verification, and therefore quantitative automata, mainly limit-average automata and discounted-sum automata, have a central role in quantitative formal verification. Yet, a bothering problem is that among the basic automata questions underlying a verification task, namely emptiness, universality, and inclusion, only emptiness is known to be solvable for these automata. The other questions are either undecidable, with limit-average automata, or not known to be decidable, with discounted-sum automata.

We showed that discounted-sum automata with an integral factor form a robust class, having algorithms for all the above questions, closed under natural composition relations, as min, max, addition and subtraction, and always allowing for determinization. Hence, we find this class of integral discounted-sum automata a promising direction in the development of formal quantitative verification.

## References

1 D. Andersson. An improved algorithm for discounted payoff games. In *ESSLLI Student Session*, pages 91–98, 2006.

2 K. Chatterjee, L. Doyen, and T. A. Henzinger. Quantitative languages. In *Proc. of CSL*, LNCS 5213, pages 385–400. Springer, 2008.

3 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Alternating weighted automata. In *FCT*, volume 5699 of *LNCS*, pages 3–13, 2009.

4 Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Expressiveness and closure properties for quantitative languages. *Logical Methods in Computer Science*, 6(3), 2010.

5 Luca de Alfaro, Thomas A. Henzinger, and Rupak Majumdar. Discounting the future in systems theory. In *ICALP*, volume 2719 of *LNCS*, pages 1022–1037, 2003.

6 Aldric Degorre, Laurent Doyen, Raffaella Gentilini, Jean-François Raskin, and Szymon Torunczyk. Energy and mean-payoff games with imperfect information. In *CSL*, pages 260–274, 2010.

7 M. Droste, W. Kuich, and H. Vogler. *Handbook of Weighted Automata*. Springer Publishing Company, Incorporated, 2009.

8 Manfred Droste and Dietrich Kuske. Skew and infinitary formal power series. *Theor. Comput. Sci.*, 366(3):199–227, 2006.

9 Hugo Gimbert and Wieslaw Zielonka. Limits of multi-discounted markov decision processes. In *LICS*, pages 89–98. IEEE Computer Society, 2007.

10 Omid Madani, Mikkel Thorup, and Uri Zwick. Discounted deterministic markov decision processes and discounted all-pairs shortest paths. *ACM Transactions on Algorithms*, 6(2), 2010.

11 M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23:269–311, 1997.

12 U. Zwick and M.S. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.