Quantitative Languages

KRISHNENDU CHATTERJEE
IST Austria (Institute of Science and Technology Austria)
LAURENT DOYEN
LSV, ENS Cachan & CNRS, France
and
THOMAS A. HENZINGER
IST Austria (Institute of Science and Technology Austria) and EPFL, Switzerland

Quantitative generalizations of classical languages, which assign to each word a real number instead of a boolean value, have applications in modeling resource-constrained computation. We use weighted automata (finite automata with transition weights) to define several natural classes of quantitative languages over finite and infinite words; in particular, the real value of an infinite run is computed as the maximum, limsup, liminf, limit average, or discounted sum of the transition weights. We define the classical decision problems of automata theory (emptiness, universality, language inclusion, and language equivalence) in the quantitative setting and study their computational complexity. As the decidability of the language-inclusion problem remains open for some classes of weighted automata, we introduce a notion of quantitative simulation that is decidable and implies language inclusion. We also give a complete characterization of the expressive power of the various classes of weighted automata. In particular, we show that most classes of weighted automata cannot be determinized.

Categories and Subject Descriptors: F.4.3 [**MATHEMATICAL LOGIC AND FORMAL LANGUAGES**]: Formal Languages

General Terms: Quantitative Verification, Weighted Automata

Additional Key Words and Phrases: Model Checking, Expressiveness

Authors' addresses: Krishnendu Chatterjee, IST, Austria. Email: Krishnendu.Chatterjee@ist.ac.at

Laurent Doyen, LSV, ENS Cachan & CNRS, 61 Av. du Président Wilson, 94230 Cachan, France. Email: ldoyen@ulb.ac.be

Thomas A. Henzinger, IST, Austria and EPFL, Switzerland. Email: tah@epfl.ch

A preliminary version of this paper appeared in the Proceedings of the 17th International Conference on Computer Science Logic (CSL), Lecture Notes in Computer Science, Springer, 2008.

This research is supported in part by the NSF grants CCR-0132780, CNS-0720884, and CCR-0225610, by the Swiss National Science Foundation, and by the European COMBEST project.

1. INTRODUCTION

The automata-theoretic approach to verification is boolean. To check that a system satisfies a specification, we construct a finite automaton A to model the system and a finite (usually nondeterministic) automaton B for the specification. The language $L(A)$ of A contains all behaviors of the system, and $L(B)$ contains all behaviors allowed by the specification. The language of an automaton A can be seen as a boolean function L_A that assigns 1 (or true) to words in $L(A)$, and 0 (or false) to words not in $L(A)$. The verification problem "does the system satisfy the specification?" is then formalized as the language-inclusion problem "is $L(A) \subseteq L(B)$?", or equivalently, "is $L_A(w) \leq L_B(w)$ for all words w?". We investigate a natural generalization of this framework: a *quantitative language* L is a function that assigns a real-numbered value $L(w)$ to each (finite or infinite) word w. With quantitative languages, systems and specifications can be formalized more accurately. For example, a system may use a varying amount of some resource (e.g., memory consumption, or power consumption) depending on its behavior, and a specification may assign a maximal amount of available resource to each behavior, or fix the long-run average available use of the resource. The quantitative language-inclusion problem "is $L_A(w) \leq L_B(w)$ for all words w?" can then be used to check, say, if for each behavior, the peak power used by the system lies below the bound given by the specification; or if for each behavior, the long-run average response time of the system lies below the specified average response requirement.

In the boolean automaton setting, the value of a word w in $L(A)$ is the maximal value of a run of A over w (if A is nondeterministic, then there may be many runs of A over w), and the value of a run is a function that depends on the class of automata: for automata over finite words, the value of a run is true if the last state of the run is accepting; for Büchi automata, the value is true if an accepting state is visited infinitely often; etc. To define quantitative languages, we use automata with weights on transitions. We again set the value of a word w as the maximal value of all runs over w, and the value of a run r is a function of the (finite or infinite) sequence of weights that appear along r. This approach is well-known from the theory of weighted automata and in this work we consider several new ways for computing the values of runs. We consider functions, such as Max and Sum of weights for finite runs, and Sup, LimSup, LimInf, limit average, and discounted sum of weights for infinite runs. For example, peak power consumption can be modeled as the maximum of a sequence of weights representing power usage; energy use can be modeled as the sum; average response time as the limit average [Chakrabarti et al. 2005; Chakrabarti et al. 2003]. Quantitative languages have also been used to specify and verify reliability requirements: if a special symbol $\perp$ is used to denote failure and has weight 1, while the other symbols have weight 0, one can use a limit-average automaton to specify a bound on the rate of failure in the long run [Chatterjee et al. 2008]. Alternatively, the discounted sum can be used to specify that failures happening later are less important than those happening soon [de Alfaro et al. 2003]. It should be noted that LimSup and LimInf automata generalize Büchi and coBüchi automata, respectively. Functions such as limit average (or mean payoff) and discounted sum are classical in game theory [Shapley 1953]; they have been studied extensively as quantitative objectives in

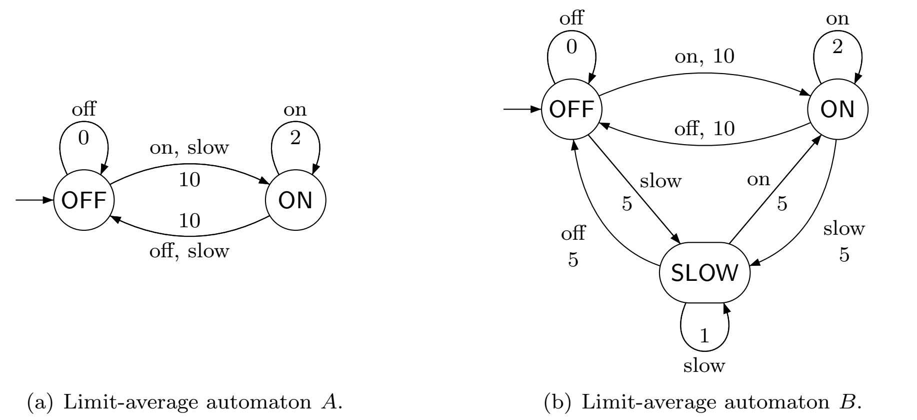


Fig. 1. Specifications for the energy consumption of a motor (B refines A).

the branching-time context of games played on graphs [Ehrenfeucht and Mycielski 1979; Condon 1992; Chakrabarti et al. 2003; Gimbert 2006].

The linear-time setting of automata and languages provides a uniform way to describe quantitative specifications (e.g., quantitative objectives as monitors in games) using the above functions, and allows to compare their expressive power and study their reducibility relationship. It is therefore natural to consider the same functions in the linear-time context of automata and languages that have been widely studied in the branching-time context of games.

Example. We illustrate the use of limit-average automata to model the energy consumption of a motor. Energy-aware design has emerged as an important topic in the recent years, and our work could be used in that direction, as illustrated by the example. Since we consider energy consumption in the long-run, it is natural to accumulate the weights as limit-average (the total energy consumed is the sum of the amounts of consumed energy). The automaton A in Figure 1(a) specifies the maximal allowed energy consumption to maintain the motor on or off, and the maximal consumption for a mode change. The specification abstracts away that a mode change can occur smoothly with the *slow* command. A refined specification B is given in Figure 1(b) where the effect of slowing down is captured by a third state. One can check that B refines A, i.e. $L_B(w) \leq L_A(w)$ for all words $w \in \{on, off, slow\}^\omega$, hence the limit-average consumption in B always satisfies the bound specified by A.

We make the following remarks about this example. First, to check that B refines A, it would not be sufficient to check locally that transitions in B have smaller weight than corresponding transitions in A. For instance, the word $slow \cdot on^\omega$ visits the sequences of weights $10, 2, 2, 2, \dots$ in A and $5, 5, 2, 2, \dots$ in B, the second weight in the sequences being larger in B than in A. The algorithmic problem of deciding whether refinement holds is discussed in Section 3 (for instance, Theorem 5 shows that refinement can be decided in polynomial time for deterministic limit-average automata). Second, if we would assign to an infinite run the supremum of its weights instead of the limit-average, then the automaton B would still refine A as the word off^ω would have value 0 in both A and B, and for all other words w, we would have $L_A(w) = 10$ and $L_B(w) \leq 10$. Now, if we assign weight 4 to the transition from ON to OFF in A, then the refinement of A by B still holds if we use Sup-automata (by exactly the same argument as before), but it fails for limit-average (e.g., for $w = (\mathit{on} \cdot \mathit{off})^\omega$, we have $L_B(w) = 10$ and $L_A(w) = 7 < 10$).

We attempt a systematic study of quantitative languages defined by weighted automata. The main novelties concern quantitative languages of infinite words, and especially those that have no boolean counterparts (i.e., limit-average and discounted-sum languages). In the first part of this paper, we consider generalizations of the boolean decision problems of emptiness, universality, language inclusion, and language equivalence. The quantitative emptiness problem asks, given a weighted automaton A and a rational number ν, whether there exists a word w such that $L_A(w) \geq \nu$. This problem can be reduced to a one-player game with a quantitative objective and is therefore solvable in polynomial time. The quantitative universality problem asks whether $L_A(w) \geq \nu$ for all words w. This problem can be formulated as a two-player game (one player choosing input letters and the other player choosing successor states) with imperfect information (the first player, whose goal is to construct a word w such that $L_A(w) < \nu$, is not allowed to see the state chosen by the second player). The problem is PSPACE-complete for simple functions like Sup, LimSup, and LimInf, but we do not know if it is decidable for limit-average or discounted-sum automata (the corresponding games of imperfect information are not known to be decidable either). The same situation holds for the quantitative language-inclusion and language-equivalence problems, which ask, given two weighted automata A and B, if $L_A(w) \leq L_B(w)$ (resp. $L_A(w) = L_B(w)$) for all words w. Therefore we introduce a notion of quantitative simulation between weighted automata, which generalizes boolean simulation relations, is decidable, and implies language inclusion. Simulation corresponds to a weaker version of the above game, where the first player has perfect information about the state of the game. In particular, this implies that quantitative simulation can be decided in NP $\cap$ coNP for limit-average and discounted-sum automata.

In the second part of this paper, we present a complete characterization of the expressive power of the various classes of weighted automata, by comparing the classes of quantitative languages they can define. The complete picture relating the expressive powers of weighted automata is shown in Figure 9 and Table III. For instance, the results for LimSup and LimInf are analogous to the special boolean cases of Büchi and coBüchi (nondeterminism is strictly more expressive for LimSup, but not for LimInf). In the limit-average and discounted-sum cases, nondeterministic automata are strictly more expressive than their deterministic counterparts. Also, one of our results shows that nondeterministic limit-average automata are not as expressive as deterministic Büchi automata (and vice versa). It may be noted that deterministic Büchi languages are complete for the second level of the Borel hierarchy [Thomas 1997], and deterministic limit-average languages are complete for the third level [Chatterjee 2007a]; so there is a Wadge reduction [Wadge 1984] from deterministic Büchi languages to deterministic limit-average languages. Our result shows that Wadge reductions are not captured by automata, and in particular, that the Wadge reduction from Büchi to limit-average languages is not regular.

Related works. In the literature, there is a wealth of results on weighted automata on finite and infinite words. We now describe the differences to the setting presented in this paper. The lattice automata of [Kupferman and Lustig 2007] map finite words to values from a finite lattice. Roughly speaking, the value of a run is the meet (greatest lower bound) of its transition weights, and the value of a word w is the join (least upper bound) of the values of all runs over w. This corresponds to Min and Inf automata in our setting, and for infinite words, the Büchi lattice automata of [Kupferman and Lustig 2007] are analogous to our LimSup automata. However, the other classes of weighted automata (Sum, limit-average, discounted-sum) cannot be defined using operations on finite lattices. The complexity of the emptiness and universality problems for lattice automata is given in [Kupferman and Lustig 2007] (and implies our results for LimSup automata), while their generalization of language inclusion differs from ours. They define the *implication value* $v(A, B)$ of two lattice automata A and B as the meet over all words w of the join of $\neg L_A(w)$ and $L_B(w)$, while we would define implication value as $v(A, B) = \min_w(L_B(w) - L_A(w))$ since min is the meet operation (and defining negation as multiplication by -1, but using $+$ instead of join), and say that B refines A if $v(A, B) \geq 0$.

In classical weighted automata [Schützenberger 1961; Mohri 1997] and semiring automata [Kuich and Salomaa 1986], the value of a finite word is defined using the two algebraic operations $+$ and $\cdot$ of a semiring as the sum of the product of the transition weights of the runs over the word. In that case, quantitative languages are called *formal power series*. Over infinite words, weighted automata with discounted sum were first investigated in [Droste and Kuske 2003]. Researchers have also considered other quantitative generalizations of languages over finite words [Droste and Gastin 2007], over trees [Droste et al. 2008], and using finite lattices [Gurfinkel and Chechik 2003]. However, these works do not address the quantitative decision problems, nor do they compare the relative expressive powers of weighted automata over infinite words, as we do here. The work of [Culik and Karhumäki 1994] studies the decision problems for weighted automata but for different notion of behaviors (different value functions). The works of [Karianto 2005; Seidl et al. 2004] consider quantitative counting properties, and the works of [Klaedtke and Rueß 2003] consider the cardinality properties in monadic second order logic, but the value functions we consider are different from the counting and cardinality properties. The works [Seidl et al. 2003; Dal-Zilio and Lugiez 2003] consider numerical properties of documents such as XML that are very different from the quantitative properties we consider. In [Chakrabarti et al. 2005], a quantitative generalization of languages is defined by discrete functions (the value of a word is an integer) and the decision problems only involve the extremal value of a language, which corresponds to emptiness.

In models that use transition weights as probabilities, such as probabilistic *Rabin automata* [Paz 1971], one does not consider values of individual infinite runs (which would usually have a value, or measure, of 0), but only measurable sets of infinite runs (where basic open sets are defined as extensions of finite runs). Our quantitative setting is orthogonal to the probabilistic framework: we assign quantitative values (e.g., peak power consumption, average response time, failure rate) to individual infinite behaviors, not probabilities to finite behaviors.

2. BOOLEAN AND QUANTITATIVE LANGUAGES

We recall the classical automata-theoretic description of boolean languages, and introduce an automata-theoretic description of several classes of quantitative languages.

2.1 Boolean Languages

A *boolean language* over a finite alphabet Σ is either a set $L \subseteq \Sigma^*$ of finite words or a set $L \subseteq \Sigma^\omega$ of infinite words. Alternatively, we can view these sets as functions in $[\Sigma^* \to \{0,1\}]$ and $[\Sigma^\omega \to \{0,1\}]$, respectively.

Boolean automata. A *(finite) automaton* is a tuple $A = \langle Q, q_I, \Sigma, \delta \rangle$ where:

—Q is a finite set of states, and $q_I \in Q$ is the initial state;
—Σ is a finite alphabet;
—$\delta \subseteq Q \times \Sigma \times Q$ is a finite set of labeled transitions.

The automaton A is *total* if for all $q \in Q$ and $\sigma \in \Sigma$, there exists $(q, \sigma, q') \in \delta$ for at least one $q' \in Q$. The automaton A is *deterministic* if for all $q \in Q$ and $\sigma \in \Sigma$, there exists $(q, \sigma, q') \in \delta$ for exactly one $q' \in Q$. We sometimes call automata *nondeterministic* to emphasize that they are not necessarily deterministic.

A *run* of A over a finite (resp. infinite) word $w = \sigma_1 \sigma_2 \dots$ is a finite (resp. infinite) sequence $r = q_0 \sigma_1 q_1 \sigma_2 \dots$ of states and letters such that (i) $q_0 = q_I$, and (ii) $(q_i, \sigma_{i+1}, q_{i+1}) \in \delta$ for all $0 \leq i < |w|$. When the run r is finite, we denote by $\mathsf{Last}(r)$ the last state in r. When r is infinite, we denote by $\mathsf{Inf}(r)$ the set of states that occur infinitely many times in r. The prefix of length i of an infinite run r is the prefix of r that contains the first i states.

Given a set $F \subseteq Q$ of final (or accepting) states, the *finite-word language* defined by the pair $\langle A, F \rangle$ is $L_A^{\mathrm{f}} = \{w \in \Sigma^* \mid$ there exists a run r of A over w such that $\mathsf{Last}(r) \in F\}$. The *infinite-word languages* defined by $\langle A, F \rangle$ are as follows: if $\langle A, F \rangle$ is interpreted as a Büchi automaton, then $L_A^{\mathrm{b}} = \{w \in \Sigma^\omega \mid$ there exists a run r of A over w such that $\mathsf{Inf}(r) \cap F \neq \varnothing\}$, and if $\langle A, F \rangle$ is interpreted as a coBüchi automaton, then $L_A^{\mathrm{c}} = \{w \in \Sigma^\omega \mid$ there exists a run r of A over w such that $\mathsf{Inf}(r) \subseteq F\}$. In the sequel, we assume that the set F is given with the description of the finite automaton A, and we often omit the superscripts in the notation L_A^{f}, L_A^{b}, and L_A^{c}, assuming that automata have a type (finite-word, Büchi, or coBüchi) that determines which language it defines.

Boolean decision problems. We recall the classical decision problems for automata, namely, emptiness, universality, language inclusion and language equivalence. Given a finite automaton A, the *boolean emptiness problem* asks whether $L_A = \varnothing$, and the *boolean universality problem* asks whether $L_A = \Sigma^*$ (for finite-word language) or $L_A = \Sigma^\omega$ (for infinite-word language). Given two finite automata A and B, the *boolean language-inclusion problem* asks whether $L_A \subseteq L_B$, and the *boolean language-equivalence problem* asks whether $L_A = L_B$. It is well-known that for both finite- and infinite-word languages, the emptiness problem is solvable in polynomial time, while the universality, inclusion, and equivalence problems are

PSPACE-complete [Meyer and Stockmeyer 1972; Sistla et al. 1987] (see also Table I).

2.2 Quantitative Languages

A *quantitative language* L over a finite alphabet Σ is either a mapping $L : \Sigma^+ \to \mathbb{R}$ or a mapping $L : \Sigma^\omega \to \mathbb{R}$, where $\mathbb{R}$ is the set of real numbers.

Weighted automata. A *weighted automaton* is a tuple $A = \langle Q, q_I, \Sigma, \delta, \gamma \rangle$ where:

—$\langle Q, q_I, \Sigma, \delta \rangle$ is a total finite automaton, and

—$\gamma : \delta \to \mathbb{Q}$ is a *weight* function, where $\mathbb{Q}$ is the set of rational numbers.

Given a finite (resp. infinite) run $r = q_0\sigma_1 q_1 \sigma_2 \dots$ of A over a finite (resp. infinite) word $w = \sigma_1\sigma_2 \dots$, let $\gamma(r) = v_0 v_1 \dots$ be the sequence of weights defined by $v_i = \gamma(q_i, \sigma_{i+1}, q_{i+1})$ for all $0 \leq i < |w|$.

Given a *value function* $\mathsf{Val} : \mathbb{Q}^+ \to \mathbb{R}$ (resp. $\mathsf{Val} : \mathbb{Q}^\omega \to \mathbb{R}$), the Val-automaton A defines the quantitative language L_A such that for all words $w \in \Sigma^+$ (resp. $w \in \Sigma^\omega$), we have $L_A(w) = \sup\{\mathsf{Val}(\gamma(r)) \mid r \text{ is a run of } A \text{ over } w\}$. We assume that $\mathsf{Val}(v)$ is bounded when the numbers in v are taken from a finite set (namely, the set of weights in A), and since weighted automata are total, $L_A(w)$ is not infinite. All value functions we consider in this paper satisfy this boundedness assumption.

Note that for boolean automata, if we assign value 1 to the accepting runs (either those that end up in an accepting state, or visit an accepting state infinitely often, or eventually visit accepting states only) and value 0 to the other runs, then the function L_A would be the characteristic function of the boolean language defined by A. Hence, the sup operator is a natural generalization to the quantitative setting of the way nondeterminism is dealt with in boolean automata. Other definitions can be considered [Chatterjee et al. 2009], like choosing inf instead of sup which would correspond to the so-called *universal* automata in the boolean case [Kupferman and Vardi 2001].

In the sequel, we denote by n the number of states and by m the number of transitions of a given automaton. We assume that rational numbers are given as pairs of integers, encoded in binary. All time bounds we give in this paper assume that the largest size of an integer in the input is a constant p. Without this assumption, most complexity results would involve a polynomial factor in p, as they require polynomially many operations of addition, multiplication, and comparison of rational numbers, which are quadratic in p.

Quantitative decision problems. We now present quantitative generalizations of the classical decision problems for automata. Given two quantitative languages L_1 and L_2 over Σ, we write $L_1 \sqsubseteq L_2$ if $L_1(w) \leq L_2(w)$ for all words $w \in \Sigma^+$ (resp. $w \in \Sigma^\omega$). Given a weighted automaton A and a rational number $\nu \in \mathbb{Q}$, the *quantitative emptiness problem* asks whether there exists a word $w \in \Sigma^+$ (resp. $w \in \Sigma^\omega$) such that $L_A(w) \geq \nu$, and the *quantitative universality problem* asks whether $L_A(w) \geq \nu$ for all words $w \in \Sigma^+$ (resp. $w \in \Sigma^\omega$). Given two weighted automata A and B, the *quantitative language-inclusion problem* asks whether $L_A \sqsubseteq L_B$, and the *quantitative language-equivalence problem* asks whether $L_A = L_B$, that is, whether $L_A(w) = L_B(w)$ for all $w \in \Sigma^+$ (resp. $w \in \Sigma^\omega$). All

	$\exists$	$\forall$	$\subseteq$	$=$
L^{f}	PTIME	PSPACE	PSPACE	PSPACE
L^{b}	PTIME	PSPACE	PSPACE	PSPACE
L^{c}	PTIME	PSPACE	PSPACE	PSPACE

Table I. Complexity's upper bound for boolean decision problems ($\exists$) emptiness, ($\forall$) universality, ($\subseteq$) inclusion, and ($=$) equivalence.

results that we present in this paper also hold for the decision problems defined above with inequalities replaced by strict inequalities.

Our purpose is the study of the quantitative decision problems for infinite-word languages and the expressive power of weighted automata that define infinite-word languages. We start with a brief overview of the corresponding results for finite-word languages, most of which follow from classical results in automata theory.

Finite words. For finite words, we consider the value functions Last, Max, and Sum such that for all finite sequences $v = v_1 \dots v_n$ of rational numbers,

$$\mathsf{Last}(v) = v_n, \qquad \mathsf{Max}(v) = \max\{v_i \mid 1 \leq i \leq n\}, \qquad \mathsf{Sum}(v) = \sum_{i=1}^{n} v_i.$$

Note that Last generalizes the classical boolean acceptance condition for finite words. One could also consider the value function $\mathsf{Min} = \min\{v_i \mid 1 \leq i \leq n\}$, which roughly corresponds to lattice automata [Kupferman and Lustig 2007].

THEOREM 1. *The quantitative emptiness problem can be solved in linear time for* Last *and* Max*-automata, and in quadratic time for* Sum*-automata. The quantitative language-inclusion problem is PSPACE-complete for* Last*- and* Max*-automata.*

Proof. We show that the quantitative language-inclusion problem is PSPACE-complete for Last- and Max-automata.

1. Given two Last-automata A and B, it is easy to construct (in polynomial time) for each weight v appearing in A or B the (boolean) finite automata $A^{\geq v}$ and $B^{\geq v}$ that accept the finite words with a value at least v according to L_A and L_B respectively. Then the quantitative language inclusion problem for A and B is equivalent to check that $L^{\mathrm{f}}_{A^{\geq v}} \subseteq L^{\mathrm{f}}_{B^{\geq v}}$ for each weight v appearing in A or B, which can be done in polynomial space.

The hardness result is obtained by a straightforward reduction of the boolean language inclusion problem for finite automata which is PSPACE-complete.

2. Given two Sup-automata $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1 \rangle$, and $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2 \rangle$, we construct a (boolean) finite automaton C whose language is empty if and only if $L_A \sqsubseteq L_B$. The (N)PSPACE algorithm will explore this automaton on-the-fly. We assume for $i = 1, 2$ that $\gamma_i(e) = \perp$ for all $e \notin \delta_i$, and let V_1, V_2 be the sets of weights appearing on transitions of A and B respectively. We define $C = \langle Q, q_I, \Sigma, \delta \rangle$ as follows:

—$Q = 2^{Q_1} \times \Gamma_1 \times 2^{Q_2} \times \Gamma_2$ where Γ_i is the set of functions $f : Q_i \to V_i$ for $i = 1, 2$;
—$q_I = (\{q_I^1\}, f_1, \{q_I^2\}, f_2)$ where $f_i(q) = \min(V_i)$ for all $q \in Q_i$ and $i = 1, 2$;
—δ contains all tuples $\langle (s_1, f_1, s_2, f_2), \sigma, (s_1', f_1', s_2', f_2') \rangle$ such that (for $i = 1, 2$):

—$s'_i = \{q' \in Q_i \mid \exists q \in s_i : (q, \sigma, q') \in \delta_i\}$;
—for all $q' \in Q_i$, $f'_i(q') = \max\{f_i(q) + \gamma_i(q, \sigma, q') \mid q \in s_i\}$ with the assumptions that $\max \varnothing = \perp$ and $n + \perp = \perp$ for all $n \in \mathbb{Q}$;

The set of accepting states of C is $F_C = \{(s_1, f_1, s_2, f_2) \mid \exists q_1 \in s_1 \cdot \forall q_2 \in s_2 : f_1(q_1) > f_2(q_2)\}$. It is easy to see that $L^{\mathrm{f}}_C \neq \varnothing$ if and only if there is a finite word w such that $L_A(w) > L_B(w)$.

The hardness result is obtained by a straightforward reduction of the boolean language inclusion problem for finite automata which is PSPACE-complete. ■

The quantitative language-inclusion problem is undecidable for Sum-automata [Krob 1992]. However, the quantitative language-inclusion problem for deterministic Sum-automata can be solved in polynomial time using a product construction. This naturally raises the question of the power of nondeterminism, which we address through translations between weighted automata.

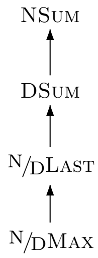
Expressiveness. A class $\mathcal{C}$ of weighted automata *can be reduced* to a class $\mathcal{C}'$ of weighted automata if for every $A \in \mathcal{C}$ there exists $A' \in \mathcal{C}'$ such that $L_A = L_{A'}$. In particular, a class of weighted automata *can be determinized* if it can be reduced to its deterministic counterpart. All reductions that we present in this paper are constructive: when $\mathcal{C}$ can be reduced to $\mathcal{C}'$, we always show how to construct an automaton $A' \in \mathcal{C}'$ that defines the same quantitative language as a given automaton $A \in \mathcal{C}$. We say that the *cost* of a reduction is $O(f(n, m))$ if for all automata $A \in \mathcal{C}$ with n states and m transitions, the constructed automaton $A' \in \mathcal{C}'$ has at most $O(f(n, m))$ many states. For all reductions we present, the size of the largest transition weight in A' is linear in the size p of the largest weight in A (however, the time needed to compute these weights may be quadratic in p).

THEOREM 2 (see also [MOHRI 1997]). *Last- and Max-automata can be determinized in $O(2^n)$ time; Sum-automata cannot be determinized. Deterministic Max-automata can be reduced to deterministic Last-automata in $O(n \cdot m)$ time; deterministic Last-automata can be reduced to deterministic Sum-automata in $O(n \cdot m)$ time. Deterministic Sum-automata cannot be reduced to Last-automata; deterministic Last-automata cannot be reduced to Max-automata.*

Results about determinizable sub-classes of Sum-automata can be found in [Mohri 1997; Kirsten and Mäurer 2005]. The results of Theorem 2 are summarized in Figure 2.

Proof of Theorem 2. It is easy to show that Last- and Max-automata can be determinized using a subset construction.

To show that Sum-automata cannot be determinized, consider the language L_N over $\Sigma = \{a, b\}$ that assigns to each finite word $w \in \Sigma^+$ the number $\max\{L_a(w), L_b(w)\}$ where $L_\sigma(w)$ is the number of occurrences of σ in w (for $\sigma \in \{a, b\}$). Clearly L_N is definable by a nondeterministic Sum-automaton. To obtain a contradiction, assume that L_N is defined by a deterministic Sum-automaton A with n states. Consider the word $w = a^n$ and let $r = q_0 a q_1 a \ldots a q_n$ be the unique run of A over w. There must exist $0 \leq i < j \leq n$ such that $q_i = q_j$, and thus $L_A(a^i) = L_A(a^j)$ since $L_A(a^i b^n) = L_A(a^j b^n) = n$. This is a contradiction since $L_N(a^i) = i \neq j = L_N(a^j)$.



	N/DMax	N/DLast	DSum	NSum
N/DMax	·	✓	✓	✓
N/DLast	×	·	✓	✓
DSum	×	×	·	✓
NSum	×	×	×	·

Fig. 2. Reducibility relation. $\mathcal{C}$ is reducible to $\mathcal{C}'$ if there is a path from $\mathcal{C}$ to $\mathcal{C}'$ in the graph, and if the entry $(\mathcal{C}, \mathcal{C}')$ is ✓in the table.

We reduce Max-automata to Last-automata as follows. Given a deterministic Max-automaton $A = \langle Q, q_I, \Sigma, \delta, \gamma \rangle$, we construct the deterministic Last-automaton $A' = \langle Q', q'_I, \Sigma, \delta', \gamma' \rangle$ as follows:

—$Q' = Q \times V$ where V is the set of weights that appear on transitions of A;

—$q'_I = (q_I, v_{\min})$ where $v_{\min}$ is the minimal weight in V;

—δ' contains all tuples $\langle (q, v), \sigma, (q', v') \rangle$ such that:
  —$(q, \sigma, q') \in \delta$, and
  —$v' = v$ if $\gamma(q, \sigma, q') \leq v$, and $v' = \gamma(q, \sigma, q')$ otherwise;

—$\gamma'(\langle (q, v), \sigma, (q', v') \rangle) = v'$ for all $\langle (q, v), \sigma, (q', v') \rangle \in \delta'$.

It is easy to see that the Last-automaton A' defines the same language as A. To show that the class of Last-automata is not reducible to the class of Max-automata, observe that the language L defined by a Max-automaton is such that $L(w_1) \leq L(w_1.w_2)$ for all $w_1, w_2 \in \Sigma^+$. It is easy to construct a Last-automaton that violates this property (consider a transition with weight v, followed by a transition with weight $v' < v$).

We reduce Last-automata to Sum-automata as follows. Given a deterministic Last-automaton $A = \langle Q, q_I, \Sigma, \delta, \gamma \rangle$, we construct the deterministic Sum-automaton $A' = \langle Q', q'_I, \Sigma, \delta', \gamma' \rangle$ as follows:

—$Q' = Q \times (V \cup \{0\})$ where V is the set of weights that appear on transitions of A;

—$q'_I = (q_I, 0)$;

—δ' contains all tuples $\langle (q, v), \sigma, (q', v') \rangle$ for each $v \in V \cup \{0\}$ such that: $(q, \sigma, q') \in \delta$ and $v' = \gamma(q, \sigma, q')$;

—$\gamma'(\langle (q, v), \sigma, (q', v') \rangle) = v' - v$ for all $\langle (q, v), \sigma, (q', v') \rangle \in \delta'$.

It is easy to see that the Sum-automaton A' defines the same language as A. To show that the class of Sum-automata is not reducible to the class of Last-automata, observe that a Sum-automaton can define a language with infinitely many different values (e.g. counting the number of a's in words), while Last-automata can only assign finitely many different values. ■

Infinite words. For infinite words, we consider the following classical value functions from $\mathbb{Q}^\omega$ to $\mathbb{R}$. Given an infinite sequence $v = v_0 v_1 \ldots$ of rational numbers taken from a finite set V (i.e., $v_i \in V$ for all $i \geq 0$), define

- $\mathsf{Sup}(v) = \sup\{v_n \mid n \geq 0\}$;
- $\mathsf{LimSup}(v) = \limsup_{n\to\infty} v_n = \lim_{n\to\infty} \sup\{v_i \mid i \geq n\}$;
- $\mathsf{LimInf}(v) = \liminf_{n\to\infty} v_n = \lim_{n\to\infty} \inf\{v_i \mid i \geq n\}$;
- $\mathsf{LimAvg}(v) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i$;
- given a discount factor $0 < \lambda < 1$, $\mathsf{Disc}_\lambda(v) = \sum_{i=0}^{\infty} \lambda^i \cdot v_i$.

Intuitively for a sequence $v = v_0 v_1 \ldots$ of rational numbers from the finite set V the Sup function chooses the maximal number that appear in v; the LimSup function chooses the maximal number that appear infinitely often in v; the LimInf function chooses the maximal number ℓ such that from some point on all numbers that are visited are at least ℓ; the LimAvg functions gives the long-run average of the numbers in v; and the Disc_λ gives the discounted sum of the numbers in v. For decision problems, we always assume that the discount factor λ is a rational number. Note that $\mathsf{LimAvg}(v)$ is defined using $\liminf$ and is therefore well-defined; all results of this paper hold also if the limit average of v is defined instead as $\limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i$. One could also consider the value function $\inf\{v_n \mid n \geq 0\}$ and obtain results analogous to the Sup value function.

Significance of value functions. The value functions provide natural generalizations of the classical boolean languages, they are complete for different levels of the Borel hierarchy, and they have been well studied in the context of game theory.

(1) The Sup value function is the natural quantitative generalization of the reachability condition and is complete for the first level of the Borel hierarchy.
(2) The LimSup and LimInf objectives are the natural quantitative generalizations of the classical Büchi and coBüchi conditions. Moreover, the LimSup and LimInf objectives are complete for the second level of the Borel hierarchy, and hence important and canonical quantitative functions.
(3) The LimAvg and Disc_λ value functions have been studied in many different contexts in game theory. Discounted functions on graph games were introduced in the seminal work of Shapley [Shapley 1953], and have been extensively studied in economics. Discounted conditions have also been studied for discounting the future in systems theory [de Alfaro et al. 2003]. The LimAvg function has also been studied extensively in the context of games on graphs: the works of Everett [Everett 1957], Liggett-Lippman [Liggett and Lippman 1969], Hopfman-Karp [Hoffman and Karp 1966], Ehrenfeucht-Mycielski [Ehrenfeucht and Mycielski 1979], Mertens-Neyman [Mertens and Neyman 1981], Zwick-Paterson [Zwick and Paterson 1996] have studied different classes of games

	$\exists$	$\forall$	$\subseteq$	$=$
Sup	PTime	PSpace	PSpace	PSpace
LimSup	PTime	PSpace	PSpace	PSpace
LimInf	PTime	PSpace	PSpace	PSpace
LimAvg	PTime	?	?	?
Deterministic LimAvg	PTime	PTime	PTime	PTime
Disc_λ	PTime	?	?	?
Deterministic Disc_λ	PTime	PTime	PTime	PTime

Table II. Complexity's upper bound for quantitative decision problems ($\exists$) emptiness, ($\forall$) universality, ($\subseteq$) inclusion and ($=$) equivalence.

with LimAvg objective. Also see the books [Filar and Vrieze 1997; Puterman 1994] for applications of discounted and limit-average value functions in the context of games on graphs. Moreover, the LimAvg value function is complete for the third level of the Borel hierarchy.

Hence the value functions considered are classical, canonical, and well-studied in the branching-time framework of games on graphs, and we study them in the linear-time framework of weighted automata.

Notation. Classes of weighted automata over infinite words are denoted with acronyms of the form xy where x is either N(ondeterministic), D(eterministic), or N/D (when deterministic and nondeterministic automata have the same expressiveness), and y is one of the following: Sup, Lsup (LimSup), Linf (LimInf), Lavg (LimAvg), or Disc. For Büchi and coBüchi condition, we use BW and CW respectively.

3. THE COMPLEXITY OF QUANTITATIVE DECISION PROBLEMS

We study the complexity of the quantitative decision problems for weighted automata over infinite words. The results are summarized in Table II.

Emptiness. The quantitative emptiness problem can be solved by reduction to the problem of finding the maximal value of an infinite path in a graph. This is decidable because pure memoryless strategies for resolving nondeterminism exist for all quantitative objectives that we consider [Filar and Vrieze 1997; Karp 1978; Andersson 2006].

Theorem 3. *The quantitative emptiness problem is solvable in $O(m+n)$ time for* Sup-*,* LimSup-*, and* LimInf-*automata; in $O(n \cdot m)$ time for* LimAvg-*automata; and in $O(n^2 \cdot m)$ time for* Disc-*automata.*

Proof sketch. Given a quantitative function Val and a weighted automaton A, let

$$\mathsf{Val}(A) = \sup\{\mathsf{Val}(\gamma(r)) \mid w \in \Sigma^\omega, r \text{ is a run of } A \text{ over } w\}.$$

It follows that the answer to the emptiness question is "Yes" iff $\mathsf{Val}(A) \geq \nu$. Given the quantitative function Val is Sup, LimSup, LimInf, LimAvg, or Disc_λ, the following assertion holds: there exists a word $\hat{w}$ such that $\hat{w} = w_1 \cdot (w_2)^\omega$, for finite words w_1 and w_2 of length at most $|Q|$ (i.e., $\hat{w}$ is a *lasso* word), such that $\mathsf{Val}(A) =$

$\sup\{\mathsf{Val}(\gamma(r)) \mid r$ is a run of A over $\hat{w}\}$. This assertion is proved as follows: from the automaton A we obtain a graph G with the same set of states and edges in the graph represent the transitions of A, rewards on edges of the graph represent the weight function. Then we use the following property about graphs with the value functions. Given a graph with rewards on edges, and objectives defined by Sup, LimSup, LimInf, LimAvg, or Disc_λ, pure memoryless optimal strategies exist. The existence of pure memoryless optimal strategy implies that for graphs with the Sup, LimSup, LimInf, LimAvg, or Disc_λ value functions there is an infinite path that consists of a finite path in the graph that is cycle free, and then repeating a cycle for ever, and the path yields the best possible value for the respective value function. The result for existence of pure memoryless optimal strategies for Sup, LimSup, and LimInf objectives in graphs are obtained by extending the result for reachability, Büchi and coBüchi objectives, respectively, and the result for LimAvg and Disc_λ can be obtained as a special case of the result known for Markov decision processes [Filar and Vrieze 1997]. Given a weighted automaton A, let n and m denote the size of Q and δ, respectively. Then the graph G obtained has n states and m edges, and we obtain the answer to the decision problem by solving the corresponding problem on graphs. The algorithms for computation of $\mathsf{Val}(A)$ is as follows:

—If Val is Sup, then $\mathsf{Val}(A)$ can be computed in $O(m+n)$ time, by classical reachability of weights greater than (or equal to) ν;

—If Val is LimSup or LimInf, then $\mathsf{Val}(A)$ can be computed in $O(m+n)$ time, by the same algorithm as for Büchi and coBüchi (computing the maximal strongly connected components, and reachability to strongly connected components) where the "accepting" edges are those with a weight greater than (or equal to) ν;

—If Val is LimAvg, then $\mathsf{Val}(A)$ can be computed in time $O(nm)$ time by applying the maximum mean cycle algorithm [Karp 1978];

—If Val is Disc_λ, then $\mathsf{Val}(A)$ can be computed in $O(n^2m)$ time by applying the algorithm to solve discounted payoff objectives in graphs with rewards on edges [Andersson 2006].

Hence the desired result follows. ■

Language inclusion. The following theorem relies on the analogous result for finite automata.

THEOREM 4. *The quantitative language-inclusion problem is PSPACE-complete for* Sup-, LimSup-, *and* LimInf-*automata.*

To prove Theorem 4, we need the following lemmas.

LEMMA 1. *Given a* Sup-*automaton* $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1 \rangle$, *we can construct in linear time a* LimSup-*automaton* $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2 \rangle$ *such that* $L_A(w) = L_B(w)$ *for all* $w \in \Sigma^\omega$.

Proof. Let $V_1 = \{\gamma_1(e) \mid e \in \delta_1\}$ be the (finite) set of weights that occur on some transitions in A. The construction is as follows:

—$Q_2 = Q_1 \cup (Q_1 \times V_1)$;
—$q_I^2 = q_I^1$;
—$\delta_2 = \delta_1 \cup \{(q, \sigma, (q', v)) \mid (q, \sigma, q') \in \delta_1 \text{ and } \gamma_1(q, \sigma, q') = v\} \cup \{((q, v), \sigma, (q, v)) \mid q \in Q_1, v \in V_1 \text{ and } \sigma \in \Sigma\}$;
—$\gamma_2(e) = \gamma_1(e)$ for all $e \in \delta_1$, and $\gamma_2(q_2, \sigma, (q, v)) = v$ for all $q_2 \in Q_2$, $\sigma \in \Sigma$ and $(q, v) \in Q_1 \times V_1$.

For every run r_1 of A, we can easily construct a run r_2 of B on the same word such that $\mathsf{Sup}(\gamma(r_1)) = \mathsf{LimSup}(\gamma(r_2))$ by looping through a state of the form (q, v) where $v = \mathsf{Sup}(\gamma(r_1))$ is the maximal value occurring in r_1. Thus we have $L_A(w) \leq L_B(w)$ for all $w \in \Sigma^\omega$.

Similarly, for every run r_2 of B, we can construct a run r_1 of A on the same word such that $\mathsf{LimSup}(\gamma(r_2)) \leq \mathsf{Sup}(\gamma(r_1))$. Indeed, either r_2 is also a run of A or it is looping through a state $(q, v) \in Q_1 \times V_1$. In each case, r_2 has a finite prefix which can be executed in A and contains a transition with weight $v = \mathsf{LimSup}(\gamma(r_2))$. We obtain r_1 by prolonging this prefix in A. Hence, we have $L_B(w) \leq L_A(w)$ for all $w \in \Sigma^\omega$. ■

Lemma 2. *Given a* LimSup*-automaton* $A = \langle Q, q_I, \Sigma, \delta, \gamma\rangle$ *and a rational number* v*, we can construct in linear time a finite automaton* $A^{\geq v}$ *with accepting states* $F^{\geq v}$ *such that* $L^b_{A^{\geq v}} = \{w \in \Sigma^\omega \mid L_A(w) \geq v\}$.

Proof. We construct $A^{\geq v} = \langle Q^{\geq v}, q_I^{\geq v}, \Sigma, \delta^{\geq v}\rangle$ as follows:

—$Q^{\geq v} = Q \times \{0, 1\}$;
—$q_I^{\geq v} = (q_I, 0)$;
—$\delta^{\geq v}$ contains all tuples $((q, i), \sigma, (q', j))$ such that $(q, \sigma, q') \in \delta$ and
  ◦ $i = 0$ and $j = 0$ and $\gamma(q, \sigma, q') < v$, or
  ◦ $i = 0$ and $j = 1$ and $\gamma(q, \sigma, q') \geq v$, or
  ◦ $i = 1$ and $j = 0$.
—$F^{\geq v} = Q \times \{1\}$.

It is easy to see that for all infinite words $w \in \Sigma^\omega$, A has a run r over w with $\mathsf{Val}(\gamma(r)) \geq v$ if and only if $A^{\geq v}$ has an accepting run over w. ■

Proof of Theorem 4. We show that the quantitative language inclusion problem for LimSup-automata is PSPACE-complete.

(1) *In PSPACE.* Let $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1\rangle$ and $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2\rangle$ be two LimSup-automata. Let V_1 be the (finite) set of weights that occur on some transitions in δ_1. Clearly, we have $L_A(w) \in V_1$ for all words $w \in \Sigma^\omega$. Consider the finite automaton $B^{\geq v}$ of Lemma 2. The quantitative language inclusion problem $A \sqsubseteq B$ is equivalent to check that $L^{\mathrm{b}}_{A^{\geq v}} \subseteq L^{\mathrm{b}}_{B^{\geq v}}$ for all $v \in V_1$, and thus it is in PSPACE. The proof for LimInf automata is similar, and the upper bound for Sup-automata follows from Lemma 1.

(2) *PSPACE-hardness.* We present the PSPACE-hardness for Sup-automata, and the PSPACE-hardness for LimSup-automata follows from Lemma 1. The hardness result for LimInf-automata is obtained in an analogous way. The hardness result for Sup-automata is obtained by a simple reduction of the boolean language inclusion problem for finite automata which is PSPACE-complete [Meyer and Stockmeyer 1972]. It suffices to assign weight 1 to the transitions entering an accepting state, and weight 0 to the other transitions. Thus the PSPACE-hardness for the quantitative language inclusion problem for Sup-, LimSup- and LimInf-automata follows.

The desired result follows. ■

We do not know if the quantitative language-inclusion problem is decidable for LimAvg- or Disc-automata. The special cases of deterministic automata are solved using a product construction (see Theorem 5).

THEOREM 5. *The quantitative language-inclusion problems $L_A \sqsubseteq L_B$ for* LimAvg- *and* Disc-*automata are decidable in polynomial time when B is deterministic.*

To prove Theorem 5, we need the following lemma.

LEMMA 3. *For all sequences $(a_n)_{n\geq 0}$ and $(b_n)_{n\geq 0}$ of real numbers, we have:*

- $\limsup\limits_{n\to\infty} \; a_n + b_n \;\leq\; \limsup\limits_{n\to\infty} \; a_n \;+\; \limsup\limits_{n\to\infty} \; b_n$
- $\limsup\limits_{n\to\infty} \; a_n - b_n \;\geq\; \limsup\limits_{n\to\infty} \; a_n \;-\; \limsup\limits_{n\to\infty} \; b_n$
- $\liminf\limits_{n\to\infty} \; a_n + b_n \;\geq\; \liminf\limits_{n\to\infty} \; a_n \;+\; \liminf\limits_{n\to\infty} \; b_n$
- $\liminf\limits_{n\to\infty} \; a_n - b_n \;\leq\; \liminf\limits_{n\to\infty} \; a_n \;-\; \liminf\limits_{n\to\infty} \; b_n$

Proof. The first statement is well known. The second statement is obtained by substituting in the first a_n with b'_n, and b_n with $a'_n - b'_n$. The last two statements follow from the first two by the equality $\limsup\limits_{n\to\infty} a_n = -\liminf\limits_{n\to\infty} -a_n$. ■

Proof of Theorem 5. Given two weighted automata $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1\rangle$ and $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2\rangle$, we define the product weighted automaton as follows: $A \times B = \langle Q_1 \times Q_2, (q_I^1, q_I^2), \Sigma, \delta_{12}, \gamma_{12}\rangle$, where $((q_1, q_2), \sigma, (q'_1, q'_2)) \in \delta_{12}$ iff $(q_1, \sigma, q'_1) \in \delta_1$ and $(q_2, \sigma, q'_2) \in \delta_2$; and for $((q_1, q_2), \sigma, (q'_1, q'_2) \in \delta_{12}$ we have $\gamma_{12}((q_1, q_2), \sigma, (q'_1, q'_2)) = \gamma_1((q_1, \sigma, q'_1)) - \gamma_2((q_2, \sigma, q'_2))$. The following assertion holds: if B is deterministic, then the answer to the quantitative inclusion problem is NO iff $\mathsf{Val}(A \times B) > 0$ where

$$\mathsf{Val}(A \times B) = \sup\{\mathsf{Val}(\gamma_{12}(r_{12})) \mid w \in \Sigma^\omega, r_{12} \text{ is a run of } A \times B \text{ over } w\},$$

and Val is LimAvg or Disc_λ. We present both directions of the proof.

(1) We first show that if $\mathsf{Val}(A \times B) > 0$, then the answer to the quantitative inclusion problem is "No". If $\mathsf{Val}(A \times B) > 0$, then it follows from arguments similar to Theorem 3, that there exists a lasso word $\hat{w} = w_1 \cdot (w_2)^\omega$ such that $L_{A\times B}(\hat{w}) > 0$. Consider a run r_{12}^* of $A \times B$ over $\hat{w}$ such that

$\mathsf{Val}(\gamma_{12}(r_{12}^*)) > 0$. The run r_{12}^* can be decomposed as a run r_1^* of A over $\hat{w}$ and the unique run r_2^* of B over $\hat{w}$ (the run r_2^* is unique since B is deterministic). Let the sequence of numbers in $\gamma_1(r_1^*)$ and $\gamma_2(r_2^*)$ be $v_0^1, v_1^1, v_2^1, v_3^1, \cdots$, and $v_0^2, v_1^2, v_2^2, v_3^2, \cdots$, respectively (note that $v_0^1, v_1^1, v_2^1, v_3^1, \cdots$ are weights defined by γ_1 in A and $v_0^2, v_1^2, v_2^2, v_3^2, \cdots$ are weights defined by γ_2 in B). Then the sequence of numbers in $\gamma_{12}(r_{12}^*)$ is $v_0^1 - v_0^2, v_1^1 - v_1^2, v_2^1 - v_2^2, v_3^1 - v_3^2, \cdots$.

—If $\mathsf{Val} = \mathsf{Disc}_\lambda$, then we have

$$\mathsf{Val}(\gamma_{12}(r_{12}^*)) = \sum_{i=0}^{\infty} \lambda^i \cdot (v_i^1 - v_i^2) = \sum_{i=0}^{\infty} \lambda^i \cdot v_i^1 - \sum_{i=0}^{\infty} \lambda^i \cdot v_i^2.$$

Since r_1^* is a run of A over $\hat{w}$ we have $L_A(\hat{w}) \geq \sum_{i=0}^{\infty} \lambda^i \cdot v_i^1$, and since B is deterministic we have $L_B(\hat{w}) = \sum_{i=0}^{\infty} \lambda^i \cdot v_i^2$. Since $\mathsf{Val}(\gamma_{12}(r_{12}^*)) > 0$, it follows that $L_A(\hat{w}) > L_B(\hat{w})$.

—If $\mathsf{Val} = \mathsf{LimAvg}$, then we have

$$\mathsf{Val}(\gamma_{12}(r_{12}^*)) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} (v_i^1 - v_i^2) \leq \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^1 - \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^2.$$

The last inequality follows from Lemma 3. Since r_1^* is a run of A over $\hat{w}$ we have $L_A(\hat{w}) \geq \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^1$. Moreover, since B is deterministic we have $L_B(\hat{w}) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^2$. Since $\mathsf{Val}(\gamma_{12}(r_{12}^*)) > 0$, it follows that $L_A(\hat{w}) > L_B(\hat{w})$.

—If Val is the lim sup version of LimAvg, i.e. $\mathsf{Val}(v_0 v_1 \ldots) = \limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i$, then we have

$$\mathsf{Val}(\gamma_{12}(r_{12}^*)) = \limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} (v_i^1 - v_i^2) \leq \limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^1 - \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^2.$$

The last inequality is obtained as follows: let $a_n = \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^1$ and $b_n = \frac{1}{n} \cdot \sum_{i=0}^{n-1} (-v_i^2)$, then by Lemma 3 we have that $\limsup_{n\to\infty}(a_n + b_n) \leq \limsup_{n\to\infty} a_n + \limsup_{n\to\infty} b_n$. Since $\limsup_{n\to\infty} b_n = -\liminf_{n\to\infty}(-b_n)$, the desired inequality follows. Observe that the last term is lim inf in the last inequality. Since the word $\hat{w}$ is a lasso word, and B is deterministic, the run r_2^* is a lasso run, and we have

$$\limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^2 = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^2 = \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^2 = L_B(\hat{w}).$$

Since r_1^* is a run of A over $\hat{w}$ we have $L_A(\hat{w}) \geq \limsup_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i^1$. Since $\mathsf{Val}(\gamma_{12}(r_{12}^*)) > 0$, it follows that $L_A(\hat{w}) > L_B(\hat{w})$.

(2) We now prove the other direction.

—We first consider the case when $\mathsf{Val} = \mathsf{Disc}_\lambda$. If for some word $\hat{w}$ we have $L_A(\hat{w}) > L_B(\hat{w})$, then consider a run r_1^* of A over $\hat{w}$, and the unique run r_2^* of B over $\hat{w}$ such that $\mathsf{Val}(\gamma_1(r_1^*)) > \mathsf{Val}(\gamma_2(r_2^*))$. For the run r_{12}^* composed of the runs r_1^* and r_2^* we have $\mathsf{Val}(\gamma_{12}(r_{12}^*)) = \mathsf{Val}(\gamma_1(r_1^*)) - \mathsf{Val}(\gamma_2(r_2^*))$ (this holds for Val being Disc_λ and follows from arguments similar to the case above). If follows that $\mathsf{Val}(\gamma_{12}(r^*_{12})) > 0$, and hence $\mathsf{Val}(A \times B) > 0$.

—We now consider the case when $\mathsf{Val} = \mathsf{LimAvg}$. If $\mathsf{Val}(A \times B) \leq 0$, then we show that for all words w we have $L_A(w) \leq L_B(w)$. Since $\mathsf{Val}(A \times B) \leq 0$, it follows that if we consider the graph with rewards on edges obtained from $A \times B$, then for all cycles C reachable from the state (q^1_I, q^2_I) in the graph the sum of rewards (obtained from the weights according to γ_{12}) is at most 0; i.e., in C the sum of the weights according to γ_2 is at least the sum of the weights according to γ_1. For a word w, let us consider a run r_1 in A and the unique run r_2 in B. Let the sequence of weights in r_1 and r_2 be $(v^1_i)_{i\geq 0}$ and $(v^2_i)_{i\geq 0}$, respectively. By the property of cycles in the graph obtained from $A \times B$ (i.e., the sum of weights by γ_2 in any cycle is at least the sum of the weights by γ_1), it follows that for all $j \geq 0$ we have

$$\sum_{i=0}^{j} v^1_i \leq \sum_{i=0}^{j} v^2_i + 2 \cdot |Q_1 \times Q_2| \cdot \beta$$

where $\beta = \max_{((q_1,q_2),\sigma,(q'_1,q'_2))\in\delta_{12}} |\gamma_1(q_1,\sigma,q'_1) - \gamma_2(q_2,\sigma,q'_2)|$. The above inequality is obtained as follows: for $j \geq 0$, if we consider $\sum_{i=0}^{j}(v^2_i - v^1_i)$, then for the sum any cycles the sum is positive, and there may be a initial prefix of length at most $|Q_1 \times Q_2|$ where the sum is at least $-|Q_1 \times Q_2| \cdot \beta$, and there may be a trailing prefix of length at most $|Q_1 \times Q_2|$ where the sum is at least $-|Q_1 \times Q_2| \cdot \beta$ (the rest can be decomposed as cycles where the sum is non-negative). Hence it follows that $\sum_{i=0}^{j}(v^2_i - v^1_i) \geq -2 \cdot |Q_1 \times Q_2| \cdot \beta$ and gives us the desired inequality. Thus we have

$$\liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v^1_i \leq \liminf_{n\to\infty} \left(\frac{1}{n} \cdot \sum_{i=0}^{n-1} v^2_i + \frac{2 \cdot |Q_1| \cdot |Q_2| \cdot \beta}{n} \right) = \liminf_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v^2_i .$$

The last equality follows since $|Q_1| \cdot |Q_2| \cdot \beta$ is fixed, and as $n \to \infty$ we have $\lim_{n\to\infty} \frac{2\cdot|Q_1|\cdot|Q_2|\cdot\beta}{n} = 0$. The result follows.

—We consider the case when Val is the lim sup version of LimAvg. If for some word $\hat{w}$ we have $L_A(\hat{w}) > L_B(\hat{w})$, then consider a run r^*_1 of A over $\hat{w}$, and the unique run r^*_2 of B over $\hat{w}$ such that $\mathsf{Val}(\gamma_1(r^*_1)) > \mathsf{Val}(\gamma_2(r^*_2))$. For the run r^*_{12} composed of the runs r^*_1 and r^*_2 we have

$$\mathsf{Val}(\gamma_{12}(r^*_{12})) \geq \mathsf{Val}(\gamma_1(r^*_1)) - \mathsf{Val}(\gamma_2(r^*_2)).$$

The above inequality follows from Lemma 3. If follows that $\mathsf{Val}(\gamma_{12}(r^*_{12})) > 0$, and hence $\mathsf{Val}(A \times B) > 0$.

It follows from above that the answer to the quantitative inclusion problem is "No" iff $\mathsf{Val}(A \times B) > 0$. It Theorem 3 we have shown that given a weighted automaton A, with the value functions Val as LimAvg or Disc_λ, the value $\mathsf{Val}(A)$ can be computed in polynomial time (by using algorithms on graphs with the value functions). It follows that $\mathsf{Val}(A \times B)$ is computable in polynomial time, for Val being LimAvg or Disc_λ. The desired result follows. ∎

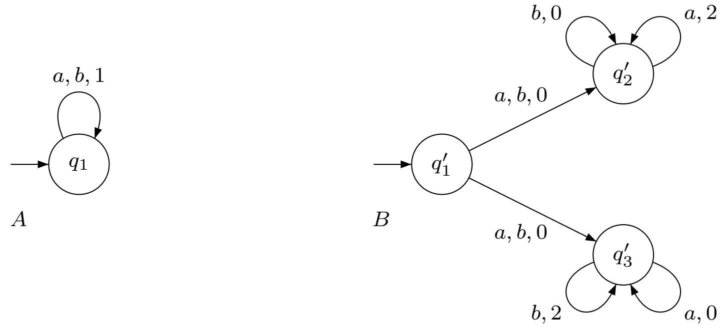


Fig. 3. Two nondeterministic limit-average automata A and B such that $L_A \not\sqsubseteq L_B$, but there is no lasso-word $w = w_1 \cdot w_2^\omega$ with $L_A(w) > L_B(w)$.

When the LimAvg automaton B is not deterministic, the decidability of quantitative language inclusion is open. We show that when language inclusion does not hold, there may not exist simple words that witness this. A simple word is a word of the form $w_1 \cdot w_2^\omega$ for finite words w_1, w_2 (w_2 nonempty), also called *lasso-word*. This observation is in contrast with the case of boolean language inclusion for, e.g., parity automata, where non-inclusion is always witnessed by a lasso-word.

Lemma 4. *There exist two* LimAvg*-automata A and B such that* (i) $L_A \not\sqsubseteq L_B$ *and* (ii) *there exists no lasso-word w such that $L_A(w) > L_B(w)$.*

Proof. Consider the two LimAvg-automata A and B shown in Figure 3, where B is nondeterministic. For all words $w \in \Sigma^\omega$, we have $L_A(w) = 1$. For a lasso-word $w = w_1 \cdot w_2^\omega$, if in w_2 there are more b's than a's, then B chooses q_3' from q_1', and else chooses q_2' from q_1'. Hence for all lasso-words $w = w_1 \cdot w_2^\omega$, we have $L_B(w) \geq 1$. However $L_A \not\sqsubseteq L_B$. Consider the word w generated inductively such that w_0 is the empty word, and w_{i+1} is generated from w_i as follows: (i) first generate a long enough sequence w_{i+1}' of a's after w_i such that the average number of b's in $w_i \cdot w_{i+1}'$ falls below $\frac{1}{3}$; (ii) then generate a long enough sequence w_{i+1}'' of b's such that the average number of a's in $w_i \cdot w_{i+1}' \cdot w_{i+1}''$ falls below $\frac{1}{3}$; and (iii) let $w_{i+1} = w_i \cdot w_{i+1}' \cdot w_{i+1}''$. The infinite word w is the limit of this sequence. For the word w, we have $L_B(w) = 2 \cdot \frac{1}{3} = \frac{2}{3} < 1$, and thus $L_A \not\sqsubseteq L_B$. ∎

For the quantitative language-inclusion problem for discounted sum automata we have the following result.

Theorem 6. *The quantitative language-inclusion problem for* Disc*-automata is co-r.e.*

Proof. For discounted-sum automata A and B, assume that there exists a finite word $w \in \Sigma^*$ such that for some run r_1 of A over w and for all runs r_2 of B over w, we have

$$v_\lambda^A(r_1) + v \cdot \frac{\lambda^{|w|}}{1-\lambda} > v_\lambda^B(r_2) + V \cdot \frac{\lambda^{|w|}}{1-\lambda} \tag{1}$$

where $v_\lambda^A(\cdot)$ and $v_\lambda^B(\cdot)$ compute the discounted sum (with discount factor λ) of runs in A and B, and v (resp. V) is the minimum (resp. maximum) weight in (the union of) A and B. Then, we immediately have $L_A \not\sqsubseteq L_B$, as $L_A(w \cdot w') > L_B(w \cdot w')$ for all words $w' \in \Sigma^\omega$. We say that w is a *finite witness* of $L_A \not\sqsubseteq L_B$. We claim that there always exists a finite witness of $L_A \not\sqsubseteq L_B$. To see this, consider an infinite word w^∞ such that $L_A(w^\infty) = \eta_1$, $L_B(w^\infty) = \eta_2$, and $\eta_1 > \eta_2$. Let r_1 be an (infinite) run of A over w^∞ whose value is η_1. For $i > 0$, consider the prefix of w^∞ of length i. Then, for all runs r_2 of B over w^∞, we have

$$v_\lambda^A(r_1^i) + V \cdot \frac{\lambda^i}{1-\lambda} \geq \eta_1 \quad \text{and} \quad v_\lambda^B(r_2^i) + v \cdot \frac{\lambda^i}{1-\lambda} \leq \eta_2$$

where r_1^i and r_2^i are the prefixes of length i of r_1 and r_2, respectively. Then, a prefix of length i of w^∞ is a finite witness of $L_A \not\sqsubseteq L_B$ if

$$\eta_1 - (V - v) \cdot \frac{\lambda^i}{1-\lambda} > \eta_2 + (V - v) \cdot \frac{\lambda^i}{1-\lambda},$$

which must hold for sufficiently large values of i.

Therefore, the following procedure terminates if $L_A \not\sqsubseteq L_B$: enumerate the finite words over Σ (and all runs of A_1 and A_2 over these words) and check the condition (1) to get a finite witness of $L_A \not\sqsubseteq L_B$. ∎

Universality and language equivalence. All of the above results about language inclusion hold for quantitative universality and language equivalence also.

THEOREM 7. *The quantitative universality problem for nondeterministic* Sup-*,* LimSup-*, and* LimInf*-automata is PSPACE-complete.*

Proof. (PSPACE upper bound). The quantitative universality problem for nondeterministic Sup-, LimSup-, and LimInf-automata can be reduced in polynomial time to respectively the boolean universality problem for finite-word languages, for infinite-word Büchi languages, and for infinite-word co-Büchi languages.

(PSPACE lower bound). The boolean universality problem for finite word languages, for infinite word Büchi languages, and for infinite word co-Büchi languages can be reduced in polynomial time to their quantitative counterparts by assigning weight 1 to the transitions entering an accepting state, and weight 0 to the other transitions, and taking the threshold $\nu = 1$. ∎

THEOREM 8. *The quantitative universality problem for deterministic* LimAvg- *and* Disc*-automata is decidable in polynomial time.*

Proof. It follows from the results of Theorem 5. ∎

THEOREM 9. *The quantitative language equivalence problem for nondeterministic* Sup-*,* LimSup-*, and* LimInf*-automata is PSPACE-complete.*

Proof. (PSPACE upper bound). The results follow from Theorem 4.

(PSPACE lower bound). The boolean language equivalence problem for finite word languages, for infinite word Büchi languages, and for infinite word co-Büchi languages can be reduced in polynomial time to their quantitative counterparts by assigning weight 1 to the transitions entering an accepting state, and weight 0 to the other transitions. ■

THEOREM 10. *The quantitative language equivalence problem for deterministic* LimAvg- *and deterministic* Disc-*automata is decidable in polynomial time.*

Proof. It follows from the results of Theorem 5. ■

4. QUANTITATIVE SIMULATION

As the decidability of the quantitative language-inclusion problems for limit-average and discounted-sum automata remain open, we introduce a notion of *quantitative simulation* as a decidable approximation of language inclusion for weighted automata. The quantitative language-inclusion problem can be viewed as a game of imperfect information, and we view the quantitative simulation problem as exactly the same game, but with perfect information. For quantitative objectives, perfect-information games can be solved much more efficiently than imperfect-information games, and in some cases the solution of imperfect-information games with quantitative objectives is not known. For example, perfect-information games with limit-average and discounted-sum objectives can be decided in NP $\cap$ coNP, whereas the solution for such imperfect-information games is not known. Second, quantitative simulation implies quantitative language inclusion, because it is easier to win a game when information is not hidden. Hence, as in the case of finite automata, simulation can be used as a conservative and efficient approximation for language inclusion.

Language-inclusion game. Let A and B be two weighted automata with weight function γ_1 and γ_2, respectively, for which we want to check if $L_A \sqsubseteq L_B$. The language-inclusion game is played by a *challenger* and a *simulator*, for infinitely many rounds. The goal of the simulator is to prove that $L_A \sqsubseteq L_B$, while the challenger has the opposite objective. The position of the game in the initial round is $\langle q_I^1, q_I^2 \rangle$ where q_I^1 and q_I^2 are the initial states of A and B, respectively. In each round, if the current position is $\langle q_1, q_2 \rangle$, first the challenger chooses a letter $\sigma \in \Sigma$ and a state q_1' such that $(q_1, \sigma, q_1') \in \delta_1$, and then the simulator chooses a state q_2' such that $(q_2, \sigma, q_2') \in \delta_2$. The position of the game in the next round is $\langle q_1', q_2' \rangle$. The outcome of the game is a pair (r_1, r_2) of runs of A and B, respectively, over the same infinite word. The simulator wins the game if $\mathsf{Val}(\gamma_2(r_2)) \geq \mathsf{Val}(\gamma_1(r_1))$. To make this game equivalent to the language-inclusion problem, we require that the challenger cannot observe the state of B in the position of the game.

Simulation game. The simulation game is the language-inclusion game without the restriction on the vision of the challenger, that is, the challenger is allowed to observe the full position of the game. Formally, given $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1 \rangle$ and

$B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2 \rangle$, a *strategy* τ for the challenger is a function from $(Q_1 \times Q_2)^+$ to $\Sigma \times Q_1$ such that for all $\pi \in (Q_1 \times Q_2)^+$, if $\tau(\pi) = (\sigma, q)$, then $(\mathsf{Last}(\pi_{|Q_1}), \sigma, q) \in \delta_1$, where $\pi_{|Q_1}$ is the projection of π on Q_1^+. A strategy τ for the challenger is *blind* if $\tau(\pi) = \tau(\pi')$ for all sequences $\pi, \pi' \in (Q_1 \times Q_2)^*$ such that $\pi_{|Q_1} = \pi'_{|Q_1}$. The set of *outcomes* of a challenger strategy τ is the set of pairs (r_1, r_2) of runs such that if $r_1 = q_0\sigma_1 q_1 \sigma_2 \dots$ and $r_2 = q'_0 \sigma_1 q'_1 \sigma_2 \dots$, then $q_0 = q_I^1$, $q'_0 = q_I^2$, and for all $i \geq 0$, we have $(\sigma_{i+1}, q_{i+1}) = \tau((q_0, q'_0) \dots (q_i, q'_i))$ and $(q'_i, \sigma_{i+1}, q'_{i+1}) \in \delta_2$. A strategy τ for the challenger is *winning* if $\mathsf{Val}(\gamma_1(r_1)) > \mathsf{Val}(\gamma_2(r_2))$ for all outcomes (r_1, r_2) of τ.

THEOREM 11. *For all value functions and weighted automata A and B, we have $L_A \sqsubseteq L_B$ if and only if there is no blind winning strategy for the challenger in the language-inclusion game for A and B.*

Proof sketch. If the quantitative language-inclusion does not hold for A, B (i.e., $L_A \not\sqsubseteq L_B$), then there exists a word $w = \sigma_1 \sigma_2 \dots$ such that $L_A(w) > L_B(w)$. Let $r = q_0 \sigma_1 q_1 \sigma_2 \dots$ be a run of A over w such that $\mathsf{Val}(\gamma_A(r)) = L_A(w)$. A blind winning strategy for the challenger is to play (σ_i, q_i) in the i^{th} round of the game. Analogously, given a blind winning strategy for the challenger, one can construct a word w and a run r of A over w such that $\mathsf{Val}(\gamma_A(r)) > \mathsf{Val}(\gamma_B(r'))$ for all runs r' of B over w. ■

Given two weighted automata A and B, there is a *quantitative simulation* of A by B if there exists no (not necessarily blind) winning strategy for the challenger in the simulation game for A and B. We note that for the special cases of Büchi and coBüchi automata, quantitative simulation coincides with *fair simulation* [Henzinger et al. 1997].

COROLLARY 1. *For all value functions and weighted automata A and B, if there is a quantitative simulation of A by B, then $L_A \sqsubseteq L_B$.*

Given two weighted automata A and B, the *quantitative simulation problem* asks if there is a quantitative simulation of A by B.

THEOREM 12. *The quantitative simulation problem for* Sup*-automata is solvable in polynomial time. The quantitative simulation problem is in NP $\cap$ coNP for* LimSup-, LimInf-, LimAvg-, *and* Disc*-automata.*

The proof of Theorem 12 is obtained as follows. The quantitative simulation problems for LimSup- and LimInf-automata is reduced to perfect-information parity games; the quantitative simulation problem for LimAvg-automata is reduced to perfect-information limit-average games; and the quantitative simulation problem for Disc-automata is reduced to perfect-information discounted-sum games. All reductions are polynomial time, and the resulting games can all be solved in NP $\cap$ coNP.

Proof of Theorem 12. First, we consider Sup-, LimSup- and LimInf-automata. Let $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1 \rangle$ and $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2 \rangle$ be two Sup-automata (or two LimSup-automata, or two LimInf-automata). Let $v_1 < v_2 < \dots < v_k$ be the weights that occur in A. We construct the graph $\mathcal{G}(A, B) = \langle Q_{challenger} \cup Q_{simulator}, q_I, E, p \rangle$ where:

—$Q_{challenger} = Q_1 \times Q_2$;
—$Q_{simulator} = Q_1 \times Q_2 \times \Sigma$;
—$q_I = (q_I^1, q_I^2)$;
—$E = \{((q_1, q_2), (q_1', q_2, \sigma)) \mid (q_1, \sigma, q_1') \in \delta_1\} \cup \{((q_1, q_2, \sigma), (q_1, q_2')) \mid (q_2, \sigma, q_2') \in \delta_2\}$;
—$p : E \to \{0, 1, \ldots, 2k\}$ assigns priorities to edges as follows:
 —$p((q_1, q_2), (q_1', q_2, \sigma)) = 2i - 1$ if $\gamma_1(q_1, \sigma, q_1') = v_i$,
 —$p((q_1, q_2, \sigma), (q_1, q_2')) = \begin{cases} 0 & \text{if} \quad \gamma_2(q_2, \sigma, q_2') < v_1 \\ 2i & \text{if } v_i \leq \gamma_2(q_2, \sigma, q_2') < v_{i+1} \\ 2k & \text{if } v_k \leq \gamma_2(q_2, \sigma, q_2') \end{cases}$

The game on $\mathcal{G}(A, B)$ is played as follows. Starting in q_I, if the current state is in $Q_{challenger}$, then the challenger chooses the successor state in the set of outgoing edges, and if the current state is in $Q_{simulator}$, then the simulator chooses the successor state in the set of outgoing edges. The game results in an infinite path through the graph.

The objective of the simulator in the game for Sup-automata is the weak-parity objective: a play satisfies a weak-parity objective if the maximal priority which occurs in the play is even (see [Thomas 1997] for details). The objective of the simulator in the game for LimSup-automata is that the maximal priority which is seen infinitely often is even, and in the game for LimInf-automata that the minimal priority which is seen infinitely often is odd, i.e., classical parity objectives.

In the three cases, a winning strategy of the simulator in $\mathcal{G}(A, B)$ is a witness that there is no winning strategy for the challenger in the simulation game. Similarly, a winning strategy for the challenger in $\mathcal{G}(A, B)$ is a witness of a winning strategy for the challenger in the simulation game. Hence it follows that the simulator wins in $\mathcal{G}(A, B)$ if and only if B simulates A.

The NP $\cap$ coNP complexity result for LimSup- and LimInf-automata then follows from the fact that parity games can be solved in NP $\cap$ coNP [Emerson and Jutla 1991]. Since weak-parity games are solvable in linear-time [Chatterjee 2008], the quantitative simulation problem for Sup-automata is solvable in polynomial time.

The simulation game for nondeterministic LimAvg- and Disc-automata (with a rational discount factor) can also be solved in NP $\cap$ coNP. We construct a game with limit-average (resp. discounted) objective. The game has the same structure (states and transitions) as $\mathcal{G}(A, B)$ above. Weights are assigned to transitions as follows: if it corresponds to a transition in A, then it has the same weight as in A, and if it corresponds to a transition in B with weight v, then it has weight $-v$ for the limit-average game and $\frac{v}{\sqrt{\lambda}}$ for the discounted game (where λ is the discount factor of A and B). Moreover, the discount factor of the discounted game is $\sqrt{\lambda}$.

Now, we consider the case of LimAvg-automata. Let $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1 \rangle$ and $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2 \rangle$ be two LimAvg-automata. We construct the quantitative perfect-information limit-average game $\mathcal{G}(A, B) = \langle Q_{\min}, Q_{\max}, q_I, E, \gamma \rangle$ where:

—$Q_{\min} = Q_1 \times Q_2$;
—$Q_{\max} = Q_1 \times Q_2 \times \Sigma$;
—$q_I = (q_I^1, q_I^2)$;

—$E = \{((q_1, q_2), (q_1', q_2, \sigma)) \mid (q_1, \sigma, q_1') \in \delta_1\} \cup \{((q_1, q_2, \sigma), (q_1, q_2')) \mid (q_2, \sigma, q_2') \in \delta_2\}$.

—γ assigns $-\gamma_1(q_1, \sigma, q_1')$ to each $((q_1, q_2), (q_1', q_2, \sigma)) \in E$, and $\gamma_2(q_2, \sigma, q_2')$ to each $((q_1, q_2, \sigma), (q_1, q_2')) \in E$.

It is easy to establish a correspondence between strategies of the challenger in the simulation game and the min-player in game $\mathcal{G}(A, B)$, and similarly, for the simulator and the max-player. The following two case analysis relates the maximal value of the perfect-information limit-average game $\mathcal{G}(A, B)$ and the simulation game. In the following analysis we use pure memoryless determinacy of perfect-information limit-average games [Ehrenfeucht and Mycielski 1979] (i.e., existence of pure memoryless optimal strategies in such games).

(1) We first show that if the maximal value that the max-player can ensure is at least 0, then B simulates A. We fix an optimal strategy τ_1^* for the max-player in $\mathcal{G}(A, B)$. Consider an arbitrary strategy for the min-player, and the resulting play π starting from q_I. Let the sequence of weights in the play be $-u_0, v_0, -u_1, v_1, -u_2, v_2, \cdots$. Since the maximal value is at least 0 and τ_1^* is an optimal strategy, it follows that

$$\liminf_{n \to \infty} \frac{1}{2n} \cdot \sum_{i=0}^{n-1} (v_i - u_i) \geq 0.$$

By Lemma 3 we obtain that

$$\frac{1}{2}\Big(\liminf_{n \to \infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} v_i - \liminf_{n \to \infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} u_i)\Big) \geq \liminf_{n \to \infty} \frac{1}{2n} \cdot \sum_{i=0}^{n-1} (v_i - u_i) \geq 0.$$

Hence the strategy that corresponds to τ_1^* in the simulation game is a witness that there is no winning strategy for the challenger, i.e., B simulates A.

(2) We now show that if the maximal value that the max-player can ensure is negative, then B does not simulate A. In this case, we fix a pure memoryless optimal strategy τ_2^* for the min-player, and let us refer to the graph after fixing τ_2^* as $\mathcal{G}(A, B)_{\tau_2^*}$. Since the maximal value that the max-player can ensure is negative and τ_2^* is an optimal strategy, it follows that the sum of weights of all cycles C reachable from q_I in $\mathcal{G}(A, B)_{\tau_2^*}$ is negative (i.e., the sum of weights by γ_1 exceeds the sum weights by γ_2 in C). By arguments similar to Theorem 5, it follows that given the strategy τ_2^*, for all strategies τ_1 of the max-player, the value of the resulting play in A exceeds the value of the resulting play in B. That is, the strategy that corresponds to τ_2^* in the simulation game is a (not necessarily blind) winning strategy for the challenger. It follows that B does not simulate A.

It follows from above that the maximal value that the max-player can enforce in $\mathcal{G}(A, B)$ is nonnegative if and only if B simulates A. The result then follows from the fact that the maximal value of perfect-information limit-average games can be decided in NP $\cap$ coNP.

Finally, we consider the case of Disc-automata. Let $A = \langle Q_1, q_I^1, \Sigma, \delta_1, \gamma_1 \rangle$ and $B = \langle Q_2, q_I^2, \Sigma, \delta_2, \gamma_2 \rangle$ be two Disc-automata (with rational discount factor λ).

We construct the discounted game $\mathcal{G}(A, B) = \langle Q_{\min}, Q_{\max}, q_I, E, \gamma \rangle$ with discount factor $\lambda' = \sqrt{\lambda}$ where:

—$Q_{\min} = Q_1 \times Q_2$;
—$Q_{\max} = Q_1 \times Q_2 \times \Sigma$;
—$q_I = (q_I^1, q_I^2)$;
—$E = \{((q_1, q_2), (q_1', q_2, \sigma)) \mid (q_1, \sigma, q_1') \in \delta_1\} \cup \{((q_1, q_2, \sigma), (q_1, q_2')) \mid (q_2, \sigma, q_2') \in \delta_2\}$.
—γ assigns $-\gamma_1(q_1, \sigma, q_1')$ to each $((q_1, q_2), (q_1', q_2, \sigma)) \in E$, and $\frac{1}{\lambda'} \cdot \gamma_2(q_2, \sigma, q_2')$ to each $((q_1, q_2, \sigma), (q_1, q_2')) \in E$.

It is easy to see that the maximal value that the max-player can enforce in this game is nonnegative if and only if B simulates A. Essentially, this is because if the sequence of weights in the play of the game $\mathcal{G}(A, B)$ is $-u_0, \frac{v_0}{\lambda'}, -u_1, \frac{v_1}{\lambda'}, \dots,$ then its λ'-discounted sum is $(v_0 - u_0) + \lambda'^2 \cdot (v_1 - u_1) + \lambda'^4 \cdot (v_2 - u_2) + \dots,$ that is $(v_0 - u_0) + \lambda \cdot (v_1 - u_1) + \lambda^2 \cdot (v_2 - u_2) + \dots$ which is nonnegative iff $\mathsf{Disc}_\lambda(v_i) \geq \mathsf{Disc}_\lambda(u_i)$.

Now, we show that deciding whether the value of the game $\mathcal{G}(A, B)$ is nonnegative can be done in NP $\cap$ coNP. This would be straightforward if λ' was rational, but $\lambda' = \sqrt{\lambda}$ can be irrational even if λ is rational. It is known that perfect-information discounted games admit pure memoryless optimal strategies, and the pure memoryless optimal strategies serve as polynomial witnesses for the NP $\cap$ coNP procedure. To complete the NP $\cap$ coNP result we need to present polynomial-time verification procedure for graphs (i.e., game graphs after a pure memoryless strategy for a player is fixed). Suppose a pure memoryless strategy for one of the player (say the min player) is fixed, and then we do the following polynomial-time check: we construct a weighted graph from $\mathcal{G}(A, B)$ and the fixed strategy for player min by first removing the edges that are not played by the strategy, and then removing the min-states and replacing every path of length 2 between max-states by a direct edge, weighted $v - u$ if the corresponding two edges in the game were labeled $-u$ and $\frac{v}{\lambda'}$, and then solving the resulting graph as a λ-discounted graph (in polynomial-time [Andersson 2006]). The dual construction can be done in a similar fashion when a strategy for player max is fixed. Since checking if the minimal (or maximal) value of a discounted graph (with rational discount) is nonnegative can be done in polynomial time, we get an NP (or coNP) procedure. ■

5. THE EXPRESSIVE POWER OF WEIGHTED AUTOMATA

We study the expressiveness of the different classes of weighted automata over infinite words by comparing the quantitative languages they can define. For this purpose, we show the existence and non-existence of translations between classes of finite and weighted automata. All reducibility relationships are summarized in Table III and Figure 9.

5.1 Positive Reducibility Results

We start with the positive results about the existence of reductions between various classes of weighted automata, most of which can be obtained by generalizing corresponding results for finite automata. Our results also hold if we allow transition weights to be irrational numbers.

First, it is clear that Büchi and coBüchi automata can be reduced to LimSup- and LimInf-automata, respectively. In addition, we have the following results.

THEOREM 13. *(i)* Sup*-automata can be determinized in* $O(2^n)$ *time; (ii)* LimInf*-automata can be determinized in* $O(m^n)$ *time; (iii) Deterministic* Sup*-automata can be reduced to deterministic* LimInf*-, to deterministic* LimSup*-, and to deterministic* LimAvg*-automata, all in* $O(n \cdot m)$ *time; (iv)* LimInf*-automata can be reduced to* LimSup*- and to* LimAvg*-automata, both in* $O(n \cdot m)$ *time.*

Proof sketch. (i) Given a Sup-automaton $A = \langle Q, q_I, \Sigma, \delta, \gamma \rangle$, we construct a deterministic Sup-automaton $A_D = \langle Q_D, q_I^D, \Sigma, \delta_D, \gamma_D \rangle$ such that $L_{A_D} = L_A$, using a subset construction:

—$Q_D = 2^Q$;

—$q_I^D = \{q_I\}$;

—δ_D contains all transitions (s, σ, s') such that $\sigma \in \Sigma$ and $s' = \{q' \in Q \mid \exists q \in s : (q, \sigma, q') \in \delta\}$;

—γ_D assigns to $(s, \sigma, s') \in \delta_D$ the weight $v = \max\{\gamma(q, \sigma, q') \mid q \in s, q' \in s' \text{ and } (q, \sigma, q') \in \delta\}$.

(ii) Given a LimInf-automaton $A = \langle Q, q_I, \Sigma, \delta, \gamma \rangle$, we construct a deterministic LimInf-automaton $A_D = \langle Q_D, q_I^D, \Sigma, \delta_D, \gamma_D \rangle$ such that $L_{A_D} = L_A$. Let the weights that appear on transitions of A be (in increasing order) $v_1 < v_2 < \cdots < v_k$. Define:

—$Q_D = \{(t_1, \dots, t_k) \mid t_1 \subseteq Q \text{ and } t_i \subseteq t_{i-1} \text{ for } i = 2, \dots, k\}$. Intuitively, A_D keeps k copies of the classical subset construction for finite automata, one for each weight in A. However, the transitions from a set t_i are limited to those with a weight at least v_i. Therefore, t_1 corresponds exactly to the subset construction, and never gets empty (since A is total). If t_i gets empty (for $i \geq 2$), it means that all runs over the finite prefix of the input word that we have read contain a weight less than v_i;

—$q_I^D = (\{q_I\}, \dots, \{q_I\})$;

—δ_D contains all transitions $((t_1, \dots, t_k), \sigma, (t'_1, \dots, t'_k))$ such that $\sigma \in \Sigma$ and for all $1 \leq i \leq k$,
  —if $t_i \neq \varnothing$, then $t'_i = \{q' \in Q \mid \exists q \in t_i : (q, \sigma, q') \in \delta \wedge \gamma(q, \sigma, q') \geq v_i\}$;
  —if $t_i = \varnothing$, then let j such that $t_j \neq \varnothing$ and $t_{j+1} = \varnothing$ (note that such j exists and is unique) and $t'_i = \{q' \in Q \mid \exists q \in t_j : (q, \sigma, q') \in \delta \wedge \gamma(q, \sigma, q') \geq v_i\}$;

  When a set t_i gets empty (then all t_j gets empty for $j > i$), it is initiated with the least nonempty set of states;

—γ_D assigns to $((t_1, \dots, t_k), \sigma, (t'_1, \dots, t'_k)) \in \delta_D$ the weight v_m where $m = \max\{j \mid t'_j \neq \varnothing\}$. Intuitively, the value of the input word is at least v_i if and only if the set t_i never gets empty from some point on. This construction generalizes the Miyano-Hayashi construction [Miyano and Hayashi 1984] for determinizing coBüchi automata.

(iii) Given a deterministic Sup-automaton $A = \langle Q, q_I, \Sigma, \delta, \gamma \rangle$, we construct a deterministic LimInf-automaton $A_D = \langle Q_D, q_I^D, \Sigma, \delta_D, \gamma_D \rangle$ such that $L_{A_D} = L_A$. Let $v_1 < v_2 < \cdots < v_k$ be the weights that appear on transitions of A. Define:

—$Q_D = Q \times \{1, \ldots, k\}$;

—$q_I^D = (q_I, 1)$;

—δ_D contains all transitions $((q, i), \sigma, (q', i'))$ such that $(q, \sigma, q') \in \delta$ and $i' = \max\{i, k\}$ where k is such that $v_k = \gamma(q, \sigma, q')$;

—$\gamma_D((q, i), \sigma, (q', i')) = v_i$ for all $((q, i), \sigma, (q', i')) \in \delta_D$.

To show that deterministic Sup-automata are reducible to deterministic LimSup-automata (resp. to deterministic LimAvg-automata), we use the same automaton A_D interpreted as a LimSup- (resp. LimAvg-) automaton.

(iv) The reduction from LimInf- to LimSup-automata (and to LimAvg-automata) essentially consists of guessing a position i and a transition weight v such that only weights greater than v are seen after position i. Once the guess is made, all transitions have weight v. Given a LimInf-automaton A we present an equivalent LimAvg-automaton B. Let $v_1 < v_2 < v_3 < \ldots < v_k$ be the set of weights of A. The automaton B is obtained as follows: we make k copies $A_1, A_2, \ldots, A_k$ of the automaton A; in automaton A_i we only allow transitions of A of weights at least v_i and assign each of them weight v_i. We start in automaton A_1 and at any point can choose to stay in A_i or choose to move to any of the copies $A_{i+1}, \ldots, A_k$. In each copy A_i, the transition of A with weights smaller than v_i are replaced by a transition with weight v_1 over the same letter and leading to a sink state. The sink state has a self-loop with weight v_1 over every letter in the alphabet. ■

5.2 Negative Reducibility Results

We show that all other reducibility relationships do not hold. The most important results in this section show that (i) deterministic coBüchi automata cannot be reduced to deterministic LimAvg-automata, deterministic Büchi automata cannot be reduced to LimAvg-automata, and (ii) neither LimAvg- nor Disc-automata can be determinized. Over the alphabet $\hat{\Sigma} = \{a, b\}$, we use in the sequel the boolean languages L_F, which contains all infinite words with finitely many a's, and L_I, which contains all infinite words with infinitely many a's. We also use the following definition. A class $\mathcal{C}$ of finite automata *can be weakly reduced* to a class $\mathcal{C}'$ of weighted automata if for every $A \in \mathcal{C}$ there exists an $A' \in \mathcal{C}'$ such that $\inf_{w \in L_A} L_{A'}(w) > \sup_{w \notin L_A} L_{A'}(w)$. Intuitively, weak reductions may not preserve the values of the words, but preserve the order on values: for two words $w \in L_A$ (i.e., $L_A(w) = 1$) and $w' \notin L_A$ (i.e., $L_A(w') = 0$), we have both $L_A(w) > L_A(w')$ and $L_{A'}(w) > L_{A'}(w')$.

The classical proof that deterministic coBüchi automata cannot be reduced to deterministic Büchi automata can be adapted to show the following theorem.

THEOREM 14. *Deterministic coBüchi automata cannot be reduced to deterministic* LimSup-*automata.*

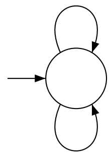


Fig. 4. A deterministic weighted automaton.

Since deterministic LimAvg- and deterministic Disc-automata can define quantitative languages whose range is infinite, while LimSup-automata cannot, we obtain the following result.

THEOREM 15. *Deterministic* LimAvg-*automata and deterministic* Disc-*automata cannot be reduced to* LimSup-*automata.*

Proof. Consider the deterministic automaton A (shown in Figure 4) that consists of a single self-loop state with weight 1 for a and 0 for b. For $j \geq 0$, consider the words $w_j = (b^j a)^\omega$ and $w'_j = a^j b^\omega$. Then we have $L_A(w_j) = \frac{1}{j+1}$ if A is interpreted as a deterministic LimAvg-automaton, and $L_A(w'_j) = \frac{1-\lambda^i}{1-\lambda}$ if A is interpreted as a deterministic Disc-automaton, i.e., the automaton A has infinitely many output values. The possible output value set for Büchi, coBüchi, $LimInf$-, and $LimSup$-automata is finite. Hence the result follows.

■

Remark. For the automaton A of Theorem 15, if we consider the language $L = \{w \in \Sigma^\omega \mid L_A(w) = 1\}$, then there is no nondeterministic Büchi or coBüchi automaton that accepts the language L. This is because it is known from [Chatterjee 2007b] that the set L is complete for the third level of the Borel hierarchy. Nondeterministic Büchi or coBüchi automata can express only ω-regular languages that lie in the boolean closure of the second level of the Borel hierarchy, and cannot express languages that are complete for the third level of the Borel hierarchy. Hence it follows that L cannot be expressed by nondeterministic Büchi or coBüchi automata.

The next theorem shows that nondeterministic LimAvg-automata are strictly more expressive than their deterministic counterpart. Theorem 17 will show that the expressive powers of LimAvg- and LimSup-automata are incomparable.

THEOREM 16. *Deterministic coBüchi automata cannot be weakly reduced to deterministic* LimAvg-*automata, and therefore they cannot be reduced to deterministic* LimAvg-*automata.* LimAvg-*automata cannot be determinized.*

Proof. Consider the language L_F of finitely many a's, which is the language defined by the deterministic coBüchi automaton shown in Figure 5. It is also easy to see that the nondeterministic LimAvg-automaton shown in Figure 6 defines L_F. We

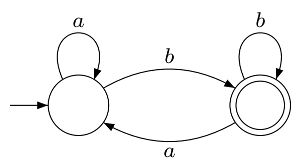


Fig. 5. A deterministic coBüchi automaton.

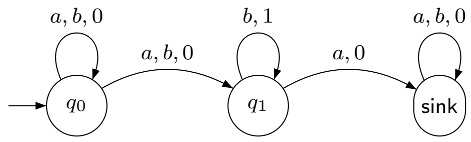


Fig. 6. A nondeterministic limit-average automaton.

show that L_F cannot be defined by any deterministic LimAvg-automaton to prove the desired claims. By contradiction, assume that A is a deterministic LimAvg-automaton with set of states Q and the initial state q_I that defines L_F. We assume without loss of generality that every state $q \in Q$ is reachable from q_I by a finite word w_q.

Let $\alpha = \inf_{w \in L_F} L_A(w)$. We claim that all b-cycles (a b-cycle is a cycle in A that can be executed with only b's) must be such that the average of the weights on the cycle is at least α. Indeed, if there is a b-cycle C in A with average weights less than α, then consider a state $q \in C$ and the word $w = w_q \cdot b^\omega$. We have $L_A(w) < \alpha$. Since $w = w_q \cdot b^\omega \in L_F$, this contradicts $\alpha = \inf_{w \in L_F} L_A(w)$.

We now show that for all $\epsilon > 0$, there exists $w' \notin L_F$ such that $L_A(w') \geq \alpha - \epsilon$. Fix $\epsilon > 0$. Let $\beta = \max_{q,q' \in Q, \sigma \in \{a,b\}} |\gamma(q, \sigma, q')|$. Let $j = \lceil \frac{6 \cdot |Q| \cdot \beta}{\epsilon} \rceil$, and consider the word $w_\epsilon = (b^j \cdot a)^\omega$. A lower bound on the average of the weights in the unique run of A over $(b^j \cdot a)$ is as follows: it can have a prefix of length at most $|Q|$ whose sum of weights is at least $-|Q| \cdot \beta$, then it goes through b-cycles for at least $j - 2 \cdot |Q|$ steps with sum of weights at least $(j - 2 \cdot |Q|) \cdot \alpha$ (since all b-cycles have average weights at least α), then again a prefix of length at most $|Q|$ without completing the cycle (with sum of weights at least $-|Q| \cdot \beta$), and then weight for a is at least $-\beta$. Hence the average is at least

$$\frac{(j - 2 \cdot |Q|) \cdot \alpha - 2 \cdot |Q| \cdot \beta - \beta}{j + 1} \geq \alpha - \frac{6 \cdot |Q| \cdot \beta}{j} \geq \alpha - \epsilon;$$

we used above that $|\alpha| \leq \beta$, and by choice of j we have $\frac{6 \cdot |Q| \cdot \beta}{j} \leq \epsilon$. Hence we have $L_A(w_\epsilon) \geq \alpha - \epsilon$. Since $\epsilon > 0$ is arbitrary, and $w_\epsilon \notin L_F$, we have $\sup_{w \notin L_F} L_A(w) \geq \alpha = \inf_{w \in L_F} L_A(w)$. This establishes a contradiction, and thus A cannot exist. The desired result follows. ■

THEOREM 17. *Deterministic Büchi automata cannot be weakly reduced to* LimAvg-*automata, and therefore they cannot be reduced to* LimAvg-*automata.*

Proof. We consider the language L_I of infinitely many a's which is accepted by the deterministic Büchi automaton shown in Figure 7.

By contradiction, assume that A is a nondeterministic LimAvg-automaton with set of states Q and initial state q_I that defines L_I. We assume without loss of generality that every state $q \in Q$ is reachable from q_I by a finite word w_q.

Let $\alpha = \sup_{w \notin L_I} L_A(w)$, and $\beta = \max_{q,q' \in Q, \sigma \in \{a,b\}} |\gamma(q, \sigma, q')|$. We claim that all b-cycles C in A must have average weights at most α; otherwise, consider a

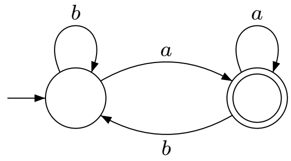


Fig. 7. A deterministic Büchi automaton.

state $q \in C$ and the word $w = w_q \cdot b^\omega$, we have $L_A(w) > \alpha$ which contradicts that $\alpha = \sup_{w \notin L_I} L_A(w)$.

We now show that for all $\epsilon > 0$, there exists $w \in L_I$ such that $L_A(w) \leq \alpha + \epsilon$. Fix $\epsilon > 0$. Let $j = \lceil \frac{3 \cdot |Q| \cdot \beta}{\epsilon} \rceil$, and consider the word $w_\epsilon = (b^j \cdot a)^\omega$. An upper bound on the average of the weights in any run of A over $(b^j \cdot a)$ is as follows: it can have a prefix of length at most $|Q|$ with the sum of weights at most $|Q| \cdot \beta$, then it follows (possibly nested) b-cycles[1] for at most j steps with sum of weights at most $j \cdot \alpha$ (since all b-cycles have average weights at most α), then again a prefix of length at most $|Q|$ without completing a cycle (with sum of weights at most $|Q| \cdot \beta$), and then weight for a is at most β. So, for any run of A over $w_\epsilon = (b^j \cdot a)^\omega$, the average weight is at most

$$\frac{j \cdot \alpha + 2 \cdot |Q| \cdot \beta + \beta}{j+1} \leq \alpha + \frac{3 \cdot |Q| \cdot \beta}{j} \leq \alpha + \epsilon$$

Hence we have $L_A(w_\epsilon) \leq \alpha + \epsilon$. Since $\epsilon > 0$ is arbitrary, and $w_\epsilon \in L_I$, we have $\inf_{w \in L_I} L_A(w) \leq \alpha = \sup_{w \notin L_I} L_A(w)$. The desired result follows. ∎

None of the weighted automata we consider can be reduced to Disc-automata (Theorem 18), and Disc-automata cannot be reduced to any of the other classes of weighted automata (Theorem 19, and also Theorem 15).

Theorem 18. *Deterministic coBüchi automata and deterministic Büchi automata cannot be weakly reduced to* Disc*-automata, and therefore they cannot be reduced to* Disc*-automata. Also deterministic* Sup*-automata cannot be reduced to* Disc*-automata.*

The proofs of Theorem 18 and 19 are based on the property that the value assigned by a Disc-automaton to an infinite word depends essentially on a finite prefix, in the sense that the values of two words become arbitrarily close when they have sufficiently long common prefixes. In other words, the quantitative language defined by a discounted-sum automaton is a continuous function in the Cantor topology. In contrast, for the other classes of weighted automata, the value of an infinite word depends essentially on its tail.

Proof of Theorem 18. First, we show that deterministic coBüchi automata cannot be weakly reduced to Disc-automata. Consider the language L_F of finitely

[1]Since A is nondeterministic, a run over b^j may have nested cycles. We can decompose the run by repeatedly eliminating the innermost cycles.

many a's. The language L_F is accepted by the deterministic coBüchi automaton shown in Figure 5.

We show that L_F is not weakly reducible to any nondeterministic Disc-automaton. By contradiction, assume that there exists a nondeterministic Disc-automaton A such that $\alpha = \inf_{w \in L_F} L_A(w) > \sup_{w \notin L_F} L_A(w) = \beta$. Then, $L_A(a^i b^\omega) \geq \alpha$ for all $i \geq 0$. So, for all $\epsilon > 0$, there exists $i \geq 0$ such that A has a run over a^i with value at least $\alpha - \epsilon$. Therefore $L_A(a^\omega) \geq \alpha - 2\epsilon$. Since this holds for all $\epsilon > 0$, we have $L_A(a^\omega) \geq \alpha$. Similarly, $L_A(b^i a^\omega) \leq \beta$ for all $i \geq 0$, and for all $\epsilon > 0$, there exists $i \geq 0$ such that A has all its runs over b^i with value at most $\beta + \epsilon$. Therefore $L_A(b^\omega) \leq \beta + 2\epsilon$. Since this holds for all $\epsilon > 0$, we have $L_A(b^\omega) \leq \beta$. Since $a^\omega \notin L_F$ and $b^\omega \in L_F$, this contradicts that $\alpha > \beta$.

Second, we show that deterministic Büchi automata cannot be weakly reduced to Disc-automata. We consider the language L_I of infinitely many a's. The deterministic Büchi automaton shown in Figure 7 accepts L_I. We now show that L_I is not weakly reducible to any nondeterministic Disc-automaton.

By contradiction, assume that there exists a nondeterministic Disc-automaton A such that $\alpha = \inf_{w \in L_I} L_A(w) > \sup_{w \notin L_I} L_A(w) = \beta$. Then, $L_A(b^i a^\omega) \geq \alpha$ for all $i \geq 0$. So, for all $\epsilon > 0$, there exists $i \geq 0$ such that A has a run over b^i with value at least $\alpha - \epsilon$. Therefore $L_A(b^\omega) \geq \alpha - 2\epsilon$. Since this holds for all $\epsilon > 0$, we have $L_A(b^\omega) \geq \alpha$. Similarly, $L_A(a^i b^\omega) \leq \beta$ for all $i \geq 0$, and for all $\epsilon > 0$, there exists $i \geq 0$ such that A has all its runs over a^i with value at most $\beta + \epsilon$. Therefore $L_A(a^\omega) \leq \beta + 2\epsilon$. Since this holds for all $\epsilon > 0$, we have $L_A(a^\omega) \leq \beta$. Since $a^\omega \in L_I$ and $b^\omega \notin L_I$, this contradicts that $\alpha > \beta$.

Third, we show that Sup-automata cannot be reduced to Disc-automata. Consider the deterministic Sup-automaton A (shown in Figure 4) that consists of a single self-loop state with weight 1 for a and 0 for b.

Assume that there exists a nondeterministic Disc-automaton B such that for all $w \in \Sigma^\omega$ we have $L_A(w) = L_B(w)$. For each $i \geq 0$, consider the word $w_i = b^i a^\omega$. We have $L_B(w_i) = L_A(w_i) = 1$, and thus for all $\epsilon > 0$, there exists $i \geq 0$ such that B has a run over b^i with value at least $1 - \epsilon$. Therefore $L_B(b^\omega) \geq 1 - 2\epsilon$. Since this holds for all $\epsilon > 0$, we have $L_B(b^\omega) \geq 1$. However, $L_A(b^\omega) = 0$ which contradicts that $L_A(w) = L_B(w)$ for all $w \in \Sigma^\omega$. ∎

THEOREM 19. *Deterministic* Disc-*automata cannot be reduced to* LimAvg-*automata.*

Proof. Consider the deterministic Disc-automaton A (shown in Figure 4) that consists of a single self-loop state with weight 1 for a and 0 for b.

Assume that there exists a nondeterministic LimAvg-automaton B such that for all $w \in \Sigma^\omega$ we have $L_A(w) = L_B(w)$. For each $i \geq 0$, consider the finite word $w_i = a^i$ and let s_i be the set of states of B in which can be the last state of a run of B over w_i. Since B is finite, there exist $j \neq k$ such that $s_j = s_k$. Therefore, $L_B(w_j b^\omega) = L_B(w_k b^\omega)$. However, $L_A(w_j b^\omega) = \frac{1-\lambda^j}{1-\lambda}$ and $L_A(w_k b^\omega) = \frac{1-\lambda^k}{1-\lambda}$ and thus $L_A(w_j b^\omega) \neq L_A(w_k b^\omega)$ which establishes a contradiction. ∎

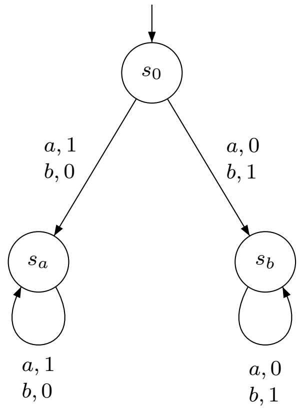


Fig. 8. The automaton N.

The next result shows that discounted-sum automata cannot be determinized. Consider the nondeterministic discounted-sum automaton N over the alphabet $\hat{\Sigma} = \{a, b\}$ shown in Figure 8. The automaton N computes the maximum of the discounted sum of a's and b's. Formally, given a (finite or infinite) word $w = w_0 w_1 \ldots \in \hat{\Sigma}^* \cup \hat{\Sigma}^\omega$, let

$$v_a(w) = \sum_{i|w_i=a}^{|w|} \lambda^i \quad \text{and} \quad v_b(w) = \sum_{i|w_i=b}^{|w|} \lambda^i$$

be the λ-discounted sum of all a's (resp. b's) in w. Then $L_N(w) = \max\{v_a(w), v_b(w)\}$ for all infinite words $w \in \hat{\Sigma}^\omega$. We show that N cannot be determinized for rational discount factors λ greater than $\frac{1}{2}$. The proof uses a sequence of intermediate lemmas.

For $\sigma \in \hat{\Sigma}$, let $\overline{\sigma} = a$ if $\sigma = b$, and $\overline{\sigma} = b$ if $\sigma = a$. We say that an infinite word $w \in \hat{\Sigma}^\omega$ *prefers* $\sigma \in \hat{\Sigma}$ if $v_\sigma(w) > v_{\overline{\sigma}}(w)$.

Lemma 5. *For all $0 < \lambda < 1$, all $w \in \hat{\Sigma}^*$, and all $\sigma \in \hat{\Sigma}$, there exists $w' \in \hat{\Sigma}^\omega$ such that $w \cdot w'$ prefers σ if and only if $v_\sigma(w \cdot \sigma^\omega) > v_{\overline{\sigma}}(w \cdot \sigma^\omega)$.*

Proof. Assume that $w.w'$ strictly prefers σ. Then $v_\sigma(w \cdot \sigma^\omega) \geq v_\sigma(w \cdot w') > v_{\bar{\sigma}}(w \cdot w') \geq v_{\bar{\sigma}}(w \cdot \sigma^\omega)$. The reverse direction is trivial. ■

We say that a finite word $w \in \hat{\Sigma}^*$ is *ambiguous* if there exist two infinite words $w'_a, w'_b \in \hat{\Sigma}^\omega$ such that $w \cdot w'_a$ prefers a and $w \cdot w'_b$ prefers b.

Lemma 6. *For all $0 < \lambda < 1$ and $w \in \hat{\Sigma}^*$, the word w is ambiguous if and only if $|v_a(w) - v_b(w)| < \frac{\lambda^{|w|}}{1-\lambda}$.*

Proof. By Lemma 5, w is ambiguous if and only if $v_a(w \cdot a^\omega) > v_b(w \cdot a^\omega)$ and $v_b(w \cdot b^\omega) > v_a(w \cdot b^\omega)$, that is

$$v_a(w) + \frac{\lambda^{|w|}}{1-\lambda} > v_b(w) \quad \text{and} \quad v_b(w) + \frac{\lambda^{|w|}}{1-\lambda} > v_a(w).$$

■

Intuitively, ambiguous words are problematic for a deterministic automaton because it cannot decide which of the two functions v_a or v_b to choose.

LEMMA 7. *For all $\frac{1}{2} < \lambda < 1$, there exists an infinite word $\hat{w} \in \hat{\Sigma}^\omega$ such that every finite prefix of $\hat{w}$ is ambiguous.*

Proof. We construct $\hat{w} = w_1 w_2 \dots$ inductively as follows. First, let $w_1 = a$ which is an ambiguous word for all $\lambda > \frac{1}{2}$ (Lemma 6). Assume that $w_1 \dots w_i$ is ambiguous for all $1 \leq i \leq k$, that is $|x_i| < \frac{\lambda^i}{1-\lambda}$ where $x_i = v_a(w_1 \dots w_i) - v_b(w_1 \dots w_i)$ (Lemma 6). We take $w_{k+1} = a$ if $x_k < 0$, and $w_{k+1} = b$ otherwise. Let us show that $|x_{k+1}| < \frac{\lambda^{k+1}}{1-\lambda}$. We have $|x_{k+1}| = \big||x_k| - \lambda^k\big|$, and thus we need to show that $|x_k| - \lambda^k < \frac{\lambda^{k+1}}{1-\lambda}$ and $-|x_k| + \lambda^k < \frac{\lambda^{k+1}}{1-\lambda}$ knowing that $|x_k| < \frac{\lambda^k}{1-\lambda}$. It suffices to show that

$$\frac{\lambda^k}{1-\lambda} \leq \lambda^k + \frac{\lambda^{k+1}}{1-\lambda} \quad \text{and} \quad \lambda^k - \frac{\lambda^{k+1}}{1-\lambda} < 0.$$

In other words, it suffices that $1 \leq 1 - \lambda + \lambda$ and $1 - \lambda - \lambda < 0$, which is true for all $\lambda > \frac{1}{2}$. ■

The word $\hat{w}$ constructed in Lemma 7 could be harmless for a deterministic automaton if some kind of periodicity or reguylarity was appearinf in reading $\hat{w}$. We make this notion formal by defining $\mathit{diff}(w) = \frac{v_a(w) - v_b(w)}{\lambda^{|w|}}$ for all finite words $w \in \hat{\Sigma}^*$. It can be shown that if the set $R_\lambda = \{\mathit{diff}(w) \mid w \in \Sigma^*\} \cap (\frac{-1}{1-\lambda}, \frac{1}{1-\lambda})$ is finite, then the automaton N can be determinized (see Appendix A), where (x, y) denotes the open interval between two reals x and y with $x < y$. Lemma 8 shows that this is also a necessary condition.

LEMMA 8. *For all $0 < \lambda < 1$, if the set R_λ is infinite, then there exists no deterministic* Disc-*automaton D such that $L_D = L_N$.*

Proof. By contradiction, assume that R_λ is infinite and there exists a deterministic Disc-automaton D such that $L_D = L_N$. For all $w \in \hat{\Sigma}^*$, let $\mathsf{Post}(w)$ be the (unique) state reached in D after reading w. We show that for all words $w_1, w_2 \in \hat{\Sigma}^*$ such that $\mathit{diff}(w_1), \mathit{diff}(w_2) \in R_\lambda$, if $\mathit{diff}(w_1) \neq \mathit{diff}(w_2)$, then $\mathsf{Post}(w_1) \neq \mathsf{Post}(w_2)$. Therefore D cannot have finitely many states.

We show this by contradiction. Assume that $\mathsf{Post}(w_1) = \mathsf{Post}(w_2)$. Then w_1 and w_2 are ambiguous by Lemma 6 since $\mathit{diff}(w_1), \mathit{diff}(w_2) \in R_\lambda$. By Lemma 5, we thus have for $i = 1, 2$,

$$L_N(w_i \cdot a^\omega) = v_a(w_i) + \frac{\lambda^{|w_i|}}{1-\lambda} \quad \text{and} \quad L_N(w_i \cdot b^\omega) = v_b(w_i) + \frac{\lambda^{|w_i|}}{1-\lambda}.$$

On the other hand, since $\mathsf{Post}(w_1) = \mathsf{Post}(w_2)$, there exist $v_1, v_2, K_a, K_b \in \mathbb{R}$ such that for $i = 1, 2$,

$$L_D(w_i \cdot a^\omega) = v_i + \lambda^{|w_i|} \cdot K_a \quad \text{and} \quad L_D(w_i \cdot b^\omega) = v_i + \lambda^{|w_i|} \cdot K_b.$$

Since $L_D = L_N$, this entails that $L_D(w_i \cdot a^\omega) - L_D(w_i \cdot b^\omega) = L_N(w_i \cdot a^\omega) - L_N(w_i \cdot b^\omega)$, and therefore

Reducibility		boolean			quantitative							
		N/DCW	DBW	NBW	N/DSup	N/DLinf	DLsup	NLsup	DLavg	NLavg	DDisc	NDisc
boolean	N/DCW	·	×	✓	×	✓	×	✓	×	✓	×	×
	DBW	×	·	✓	×	×	×	✓	×	×	×	×
	NBW	×	×	·	×	×	×	✓	×	×	×	×
quantitative	N/DSup				·	✓	✓	✓	✓	✓	×	×
	N/DLinf				×	·	×	✓	×	✓	×	×
	DLsup				×	×	·	✓	×	×	×	×
	NLsup		×		×	×	×	·	×	×	×	×
	DLavg				×	×	×	×	·	✓	×	×
	NLavg				×	×	×	×	×	·	×	×
	DDisc				×	×	×	×	×	×	·	✓
	NDisc				×	×	×	×	×	×	×	·

Table III. Reducibility relation. $\mathcal{C}$ is reducible to $\mathcal{C}'$ if the entry $R(\mathcal{C},\mathcal{C}')$ is ✓.

$$\frac{v_a(w_1) - v_b(w_1)}{\lambda^{|w_1|}} = K_a - K_b = \frac{v_a(w_2) - v_b(w_2)}{\lambda^{|w_2|}}$$

which yields a contradiction (to the fact that $\mathit{diff}(w_1) \neq \mathit{diff}(w_2)$). ■

We are now ready to prove the following theorem.

THEOREM 20. Disc-*automata cannot be determinized.*

Proof. Let λ^* be a non-algebraic number in the open interval $(\frac{1}{2}, 1)$. Then, we show that the set R_{λ^*} is infinite, which establishes the theorem by Lemma 8.

By Lemma 6 and Lemma 7, there exist infinitely many finite words $w \in \hat{\Sigma}^*$ such that $\mathit{diff}(w) \in R_{\lambda^*}$. Since λ^* is not algebraic, the polynomial equation $\mathit{diff}(w_1) = \mathit{diff}(w_2)$ cannot hold for $w_1 \neq w_2$. Therefore, R_{λ^*} is infinite. ■

By a careful analysis of the shape of the family of polynomial equations in the above proof (see Lemma 9), we can show that the automaton N cannot be determinized for any rational value of λ greater than $\frac{1}{2}$.

LEMMA 9. *For all finite words $w_1, w_2 \in \Sigma^*$ with $w_1 \neq w_2$, the polynomial equation $\mathit{diff}(w_1) = \mathit{diff}(w_2)$ in variable λ has no rational solution in $]\frac{1}{2}, 1[$.*

Proof. First, consider a polynomial $f(x) = c_0 + c_1 \cdot x + \cdots + c_n \cdot x^n$ with integer coefficients, $c_0 \neq 0$ and $c_n \neq 0$. If $f(\frac{p}{q}) = 0$ for some mutually prime integers p and q, then we have

$$c_0 \cdot q^n + c_1 \cdot p \cdot q^{n-1} + \cdots + c_{n-1} \cdot p^{n-1} \cdot q + c_n \cdot p^n = 0.$$

The first term in this sum must be a multiple of p since the rest of the sum is divisible by p, and analogously the last term must be a multiple of q. Since $\frac{p}{q}$ is irreducible, it must be that p divides c_0 and q divides c_n.

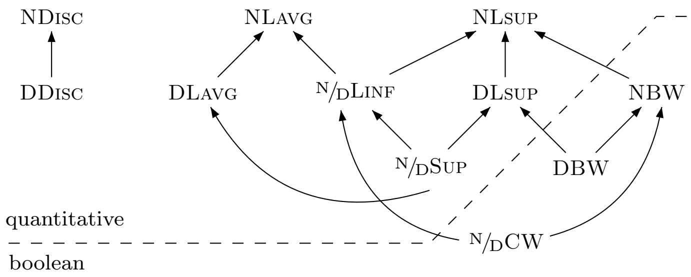


Fig. 9. Reducibility relations: a class $\mathcal{C}$ of automata can be reduced to $\mathcal{C}'$ iff there is a path from $\mathcal{C}$ to $\mathcal{C}'$.

Now, consider the equation $\mathit{diff}(w_1) = \mathit{diff}(w_2)$:

$$\frac{v_a(w_1) - v_b(w_1)}{\lambda^{|w_1|}} = \frac{v_a(w_2) - v_b(w_2)}{\lambda^{|w_2|}}.$$

Assume without loss of generality that $n = |w_2| - |w_1| \geq 0$. We get the equation

$$\lambda^n(v_a(w_1) - v_b(w_1)) = v_a(w_2) - v_b(w_2).$$

If $n > 0$, then the coefficient of the term of degree zero in the above equation is either 1 or -1, and by the above argument, the only rational solutions it can have are 1 or -1. If $n = 0$, then it is easy to see that at least one coefficient is not zero (since $w_1 \neq w_2$) and every non-zero coefficient is either 2 or -2. Therefore, as above the only possible rational solutions are 1 or -1. The result follows. ∎

REFERENCES

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

Chakrabarti, A., Chatterjee, K., Henzinger, T. A., Kupferman, O., and Majumdar, R. 2005. Verifying quantitative properties using bound functions. In *Proc. of CHARME: Correct Hardware Design and Verification Methods*. LNCS 3725. Springer, 50–64.

Chakrabarti, A., de Alfaro, L., Henzinger, T. A., and Stoelinga, M. 2003. Resource interfaces. In *Proc. of EMSOFT: Embedded Software*. LNCS 2855. Springer, 117–133.

Chatterjee, K. 2007a. Concurrent games with tail objectives. *Theoretical Computer Science 388*, 181–198.

Chatterjee, K. 2007b. Stochastic ω-regular games. Ph.D. thesis, University of California, Berkeley.

Chatterjee, K. 2008. Linear time algorithm for weak parity games. *CoRR abs/0805.1391*.

Chatterjee, K., Doyen, L., and Henzinger, T. A. 2009. Alternating weighted automata. In *Proc. of FCT: Fundamentals of Computation Theory*. LNCS 5699. Springer, 3–13.

Chatterjee, K., Ghosal, A., Henzinger, T. A., Iercan, D., Kirsch, C., Pinello, C., and Sangiovanni-Vincentelli, A. 2008. Logical reliability of interacting real-time tasks. In *Proc. of DATE: Design, Automation and Test*. ACM.

Condon, A. 1992. The complexity of stochastic games. *Information and Computation 96*, 203–224.

Culik, K. and Karhumäki, J. 1994. Finite automata computing real functions. *SIAM J. Computing 23*, 789–814.

DAL-ZILIO, S. AND LUGIEZ, D. 2003. Xml schema, tree logic and sheaves automata. In *Proc. of RTA: Rewriting Techniques and Applications*. 246–263.

DE ALFARO, L., HENZINGER, T. A., AND MAJUMDAR, R. 2003. Discounting the future in systems theory. In *Proc. of ICALP: International Colloquium on Automata, Languages and Programming*. LNCS 2719. Springer, 1022–1037.

DROSTE, M. AND GASTIN, P. 2007. Weighted automata and weighted logics. *Theoretical Computer Science 380*, 69–86.

DROSTE, M., KUICH, W., AND RAHONIS, G. 2008. Multi-valued MSO logics over words and trees. *Fundamenta Informaticae 84*, 305–327.

DROSTE, M. AND KUSKE, D. 2003. Skew and infinitary formal power series. In *Proc. of ICALP: International Colloquium on Automata, Languages and Programming*. LNCS 2719. Springer, 426–438.

EHRENFEUCHT, A. AND MYCIELSKI, J. 1979. Positional strategies for mean payoff games. *International Journal of Game Theory 8*, 109–113.

EMERSON, E. A. AND JUTLA, C. 1991. Tree automata, mu-calculus and determinacy. In *Proc. of FOCS: Foundations of Computer Science*. IEEE, 368–377.

EVERETT, H. 1957. Recursive games. In *Contributions to the Theory of Games III*. Annals of Mathematical Studies, vol. 39. 47–78.

FILAR, J. AND VRIEZE, K. 1997. *Competitive Markov Decision Processes*. Springer.

GIMBERT, H. 2006. Jeux positionnels. Ph.D. thesis, Université Paris 7.

GURFINKEL, A. AND CHECHIK, M. 2003. Multi-valued model checking via classical model checking. In *Proc. of CONCUR: Concurrency Theory*. LNCS 2761. Springer, 263–277.

HENZINGER, T., KUPFERMAN, O., AND RAJAMANI, S. 1997. Fair simulation. In *Proc. of CONCUR: Concurrency Theory*. LNCS 1243. Springer, 273–287.

HOFFMAN, A. AND KARP, R. 1966. On nonterminating stochastic games. *Management Sciences 12,* 5, 359–370.

KARIANTO, W. 2005. Adding monotonic counters to automata and transition graphs. In *Proc. of DLT: Developments in Language Theory*. 308–319.

KARP, R. M. 1978. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics 23,* 3, 309–311.

KIRSTEN, D. AND MÄURER, I. 2005. On the determinization of weighted automata. *Journal of Automata, Languages and Combinatorics 10*, 287–312.

KLAEDTKE, F. AND RUESS, H. 2003. Monadic second-order logics with cardinalities. In *Proc. of ICALP: International Colloquium on Automata, Languages and Programming*. 681–696.

KROB, D. 1992. The equality problem for rational series with multiplicities in the tropical semiring is undecidable. In *Proc. of ICALP: International Colloquium on Automata, Languages and Programming*. LNCS 623. Springer, 101–112.

KUICH, W. AND SALOMAA, A. 1986. *Semirings, Automata, Languages*. Monographs in Theoretical Computer Science. An EATCS Series, vol. 5. Springer.

KUPFERMAN, O. AND LUSTIG, Y. 2007. Lattice automata. In *Proc. of VMCAI: Verification, Model Checking, and Abstract Interpretation*. LNCS 4349. Springer, 199–213.

KUPFERMAN, O. AND VARDI, M. Y. 2001. Weak alternating automata are not that weak. *ACM Trans. Comput. Log. 2,* 3, 408–429.

LIGGETT, T. A. AND LIPPMAN, S. A. 1969. Stochastic games with perfect information and time average payoff. *Siam Review 11*, 604–607.

MERTENS, J. AND NEYMAN, A. 1981. Stochastic games. *International Journal of Game Theory 10*, 53–66.

MEYER, A. R. AND STOCKMEYER, L. J. 1972. The equivalence problem for regular expressions with squaring requires exponential space. In *Proc. of FOCS: Foundations of Computer Science*. IEEE, 125–129.

MIYANO, S. AND HAYASHI, T. 1984. Alternating finite automata on omega-words. In *Proc. of CAAP: Int. Colloquium on Trees in Algebra and Programming*. 195–210.

Mohri, M. 1997. Finite-state transducers in language and speech processing. *Comp. Linguistics 23,* 2, 269–311.

Paz, A. 1971. *Introduction to probabilistic automata*. Computer Science and Applied Mathematics. Academic Press, New York.

Puterman, M. 1994. *Markov Decision Processes*. John Wiley and Sons.

Schützenberger, M. P. 1961. On the definition of a family of automata. *Information and Control 4*, 245–270.

Seidl, H., Schwentick, T., and Muscholl, A. 2003. Numerical document queries. In *PODS*. 155–166.

Seidl, H., Schwentick, T., Muscholl, A., and Habermehl, P. 2004. Counting in trees for free. In *Proc. of ICALP: International Colloquium on Automata, Languages and Programming*. 1136–1149.

Shapley, L. S. 1953. Stochastic games. In *Proc. of the National Acadamy of Science USA*. Vol. 39. 1095–1100.

Sistla, A. P., Vardi, M. Y., and Wolper, P. 1987. The complementation problem for Büchi automata with applications to temporal logic. *Theoretical Computer Science 49*, 217–237.

Thomas, W. 1997. Languages, automata, and logic. In *Handbook of Formal Languages*. Vol. 3, Beyond Words. Springer, Chapter 7, 389–455.

Wadge, W. 1984. Reducibility and determinateness of Baire spaces. Ph.D. thesis, University of California, Berkeley.

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

A. DETERMINIZATION OF N WHEN R_λ IS FINITE

LEMMA 10. *For $\frac{1}{2} < \lambda < 1$ and $R_\lambda = \{\frac{v_a(w)-v_b(w)}{\lambda^{|w|}} \mid w \in \Sigma^*\} \cap [-\frac{1}{1-\lambda}, \frac{1}{1-\lambda}]$, the* Disc*-automaton N of Figure 8 can be determinized if the set R_λ is finite.*

Proof sketch. We construct the DDISC $A_D = \langle Q, I, \Sigma, \delta, \gamma\rangle$ such that $L_{A_D} = L_N$ as follows:

- $Q = R_\lambda \cup \{s_a, s_b\}$;
- $I = \{0\}$ (notice that $0 \in R_\lambda$ since $v_a(\epsilon) - v_b(\epsilon) = 0$);
- $\Sigma = \{a, b\}$;
- δ contains the following transitions:
  - $(s_a, a, s_a), (s_a, b, s_a), (s_b, a, s_b), (s_b, b, s_b)$
  - all transitions (s, a, s') such that $s' = \begin{cases} s_a & \text{if } \frac{s+1}{\lambda} \geq \frac{1}{1-\lambda} \\ s_b & \text{if } \frac{s+1}{\lambda} \leq \frac{-1}{1-\lambda} \\ \frac{s+1}{\lambda} & \text{otherwise} \end{cases}$
  - all transitions (s, b, s') such that $s' = \begin{cases} s_a & \text{if } \frac{s-1}{\lambda} \geq \frac{1}{1-\lambda} \\ s_b & \text{if } \frac{s-1}{\lambda} \leq \frac{-1}{1-\lambda} \\ \frac{s-1}{\lambda} & \text{otherwise} \end{cases}$
- $\gamma : \delta \to \mathbb{Q}$ is defined as follows:
  - $\gamma(s_a, a, s_a) = \gamma(s_b, b, s_b) = 1$;
  - $\gamma(s_a, b, s_a) = \gamma(s_b, a, s_b) = 0$;
  - $\gamma(s, a, s') = 1$ and $\gamma(s, b, s') = 0$ for all $s, s' \in R_\lambda$
  - $\gamma(s, a, s_a) = 1$ and $\gamma(s, b, s_b) = 1 - s$ for all $s \in R_\lambda$

The correctness of this construction can be justified by the following observations:

(1) If $s = \frac{v_a(w)-v_b(w)}{\lambda^{|w|}}$, then $\frac{v_a(w.a)-v_b(w.a)}{\lambda^{|w.a|}} = \frac{v_a(w)+\lambda^{|w|}-v_b(w)}{\lambda^{|w|+1}} = \frac{s+1}{\lambda}$ and similarly $\frac{v_a(w.b)-v_b(w.b)}{\lambda^{|w.b|}} = \frac{s-1}{\lambda}$

(2) The automaton A_D gives weight $v_a(w)$ to the finite words w such that $\mathsf{Post}(w) \in R_\lambda$. Assume that A_D is in state $s \in R_\lambda$ after reading a finite word w. Then, if $(s, a, s_a) \in \delta$, the correct value is given to every continuation of $w.a$, since they all prefer a (cf. Lemma 6). If $(s, b, s_b) \in \delta$, then all continuations of $w.b$ prefer b and the value given by A_D to $w.b$ is $v_a(w) + \lambda^{|w|} \cdot (1 - s) = v_a(w) + \lambda^{|w|} - (v_a(w) - v_b(w)) = v_b(w.b)$.

(3) By Lemma 6, if an infinite word w has all its prefixes ambiguous, then $v_a(w) = v_b(w)$ and thus the value given by A_D to w (which is $v_a(w)$) is correct in this case as well.

∎

Figure 10 shows a deterministic Disc-automaton defining the language L_N for $\lambda = \frac{\sqrt{5}-1}{2}$ (which gives a finite set R_λ).

Received August 2008; revised September 2009; accepted December 2009

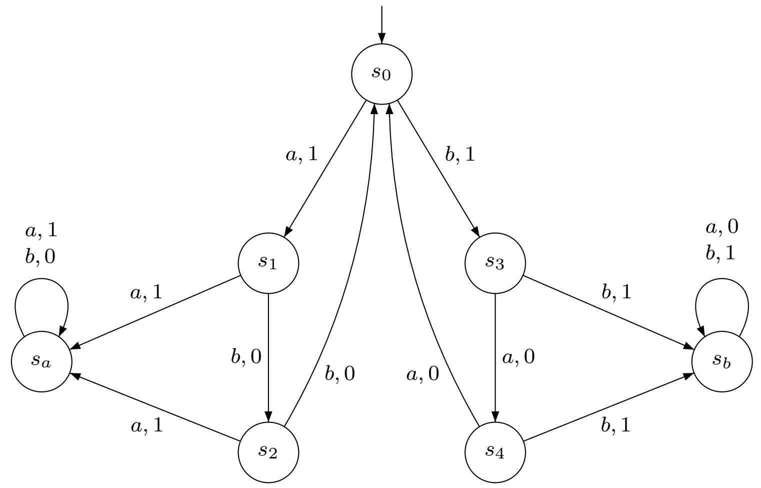


Fig. 10. A_D for $\lambda = \frac{\sqrt{5}-1}{2}$ (i.e., $1 = \lambda + \lambda^2$).