International Journal of Foundations of Computer Science

Equivalence of Labeled Markov Chains*

Laurent Doyen
I&C, École Polytechnique Fédérale de Lausanne (EPFL)
1015 Lausanne, Switzerland

Thomas A. Henzinger
I&C, École Polytechnique Fédérale de Lausanne (EPFL)
EECS, University of California at Berkeley

Jean-François Raskin
DI, Université Libre de Bruxelles (ULB)
Boulevard du Triomphe CP 212, 1050 Bruxelles, Belgium

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

We consider the equivalence problem for labeled Markov chains (LMCs), where each state is labeled with an observation. Two LMCs are equivalent if every finite sequence of observations has the same probability of occurrence in the two LMCs. We show that equivalence can be decided in polynomial time, using a reduction to the equivalence problem for probabilistic automata, which is known to be solvable in polynomial time. We provide an alternative algorithm to solve the equivalence problem, which is based on a new definition of bisimulation for probabilistic automata. We also extend the technique to decide the equivalence of weighted probabilistic automata.
Then, we consider the equivalence problem for labeled Markov decision processes (LMDPs), which asks given two LMDPs whether for every scheduler (*i.e.* way of resolving the nondeterministic decisions) for each of the processes, there exists a scheduler for the other process such that the resulting LMCs are equivalent. The decidability of this problem remains open. We show that the schedulers can be restricted to be observation-based, but may require infinite memory.

Keywords: Labeled Markov chain, Markov decision process, probabilistic automaton, equivalence, bisimulation.

*This research was supported by the Belgian FNRS grant 2.4530.02 of the FRFC project "Centre Fédéré en Vérification", by the project "MoVES", an Interuniversity Attraction Poles Programme of the Belgian Federal Government, by the Swiss National Science Foundation, and by the European Network of Excellence on Embedded Systems Design (ARTIST 2).

1. Introduction

A labeled Markov chain over an alphabet Σ is a state-transition graph whose states are labeled by observations in Σ, and whose transitions are probabilistic. The probability of a transitions $q \to q'$ is fixed independently of the past sequence of statefs visited by the machine. It generates a trace distribution $P : \Sigma^+ \to [0, 1]$, where $P(w)$ is the probability to observe the sequence $w \in \Sigma^+$ when the Markov chain is executed for $|w|$ steps.

In a labeled Markov decision process, the probabilities of the transitions can be chosen among a finite set Γ of probabilistic transitions. This choice is made by a *scheduler* which can in general choose randomly an element in Γ, and can change the choice made in a given state, depending on the history of the process. A Markov decision process can be seen as a set of Markov chains, each associated with a particular scheduler.

Given two labeled Markov chains, the *equivalence problem* asks whether they generate the same trace distribution. To show that this question can be decided in polynomial time, we consider the same problem for probabilistic automata [4, 3]. A probabilistic automaton is a finite automaton whose transitions are probabilistic and that accepts or rejects finite words probabilistically, defining a trace distribution. A polynomial-time $O(n^4)$ algorithm is known to solve the equivalence problem for probabilistic automata [6].

In this paper, we present a new algorithm with the same complexity, which is based on a fixed-point computation. The correctness of this algorithm is established using a relation over state distributions for probabilistic automata, which is reminiscent of the well-known bisimulation relation for finite automata. In contrast to [6], we compute all initial distributions that make the two automata equivalent, which reduces the execution time to $O(n^2)$ for further equivalence checks of the same automata with different initial distributions. We also give a simple linear-time equivalence-preserving transformation of labeled Markov chains to probabilistic automata, which implies that all the above results also apply to labeled Markov chains. Finally, we extend our technique to solve the equivalence problem for weighted probabilistic automata. A weighted probabilistic automaton has weights on transitions, and the value of a word w is defined as the expected value of the maximal weight of all paths over w in the automaton. Two weighted probabilistic automata are equivalent if every word has the same value in the two automata.

Then, we consider the refinement and equivalence problems for labeled Markov decision processes. Given two labeled Markov decision processes $\mathcal{P}_1$ and $\mathcal{P}_2$, the *refinement problem* asks whether for every scheduler for $\mathcal{P}_1$, there exists a scheduler for $\mathcal{P}_2$ such that the resulting labeled Markov chains are equivalent. If yes, we say that $\mathcal{P}_1$ refines $\mathcal{P}_2$. The *equivalence problem* asks whether both $\mathcal{P}_1$ refines $\mathcal{P}_2$ and $\mathcal{P}_2$ refines $\mathcal{P}_1$. The decidability of these problems remains open, even in the particular case where $\mathcal{P}_1$ or $\mathcal{P}_2$ is a labeled Markov chain. We show that the schedulers used to resolve the decisions can be restricted to observation-based, but may require infinite memory.

2. Definitions

Given a set A, we denote by $\mathcal{D}(A)$ the set of all *probabilistic distributions* over A, *i.e.* functions $X : A \to [0,1]$ with $\sum_A X(a) = 1$, and given $a \in A$ we denote by X_a the *Dirac-distribution* over A such that $X_a(a) = 1$. Let A^+ be the set of nonempty finite sequences of elements in A, and $A^* = A^+ \cup \{\epsilon\}$ where ϵ is the empty word (of length 0). For each $n \geq 0$, let A^n be the set of all sequences $w \in A^*$ of length n.

Labeled Markov chains A *labeled Markov chain* (LMC) over Σ is a tuple $\mathcal{M} = \langle Q, \pi_0, \Sigma, \mathcal{L}, \delta \rangle$ where:

- Q is a finite set of states;
- $\pi_0 \in \mathcal{D}(Q)$ is the initial distribution of states;
- Σ is a finite set of observations;
- $\mathcal{L} : Q \to \Sigma$ is a labelling function. We extend $\mathcal{L}$ to finite sequences of states in the natural way, *i.e.* $\mathcal{L}(q_0 \dots q_k) = \mathcal{L}(q_0) \dots \mathcal{L}(q_k)$;
- $\delta : Q \to \mathcal{D}(Q)$ is a probabilistic transition function. In state q, the successor state is q' with probability $\delta(q)(q')$. We extend δ to distributions over Q as follows: for all $X \in \mathcal{D}(Q)$, let $\delta(X)$ be the distribution defined by $\delta(X)(q') = \sum_{q \in Q} X(q) \cdot \delta(q)(q')$ for all $q' \in Q$.

Define $\Delta : \mathcal{D}(Q) \times Q^+ \to [0,1]$, the function that gives the probability $\Delta(X, \bar{q})$ of observing the sequence of states $\bar{q} = q_0 \dots q_k$ when $\mathcal{M}$ is started with the initial distribution X as follows:

$$\Delta(X, \bar{q}) = X(q_0) \cdot \prod_{i=1}^{k} \delta(q_{i-1})(q_i)$$

We extend Δ to nonempty sequences of observations as follows: for all $w \in \Sigma^+$, let $\Delta(X, w) = \sum_{\{\bar{q} \mid \mathcal{L}(\bar{q}) = w\}} \Delta(X, \bar{q})$ be the probability of observing w when $\mathcal{M}$ is started with the initial distribution X.

The labeled Markov chain $\mathcal{M}$ defines the *trace distribution* $P_{\mathcal{M}} : \Sigma^+ \to [0,1]$ defined by $P_{\mathcal{M}}(w) = \Delta(\pi_0, w)$ for all $w \in \Sigma^+$. Notice that for each length $n \geq 1$, the restriction of $P_{\mathcal{M}}$ to Σ^n is a probabilistic distribution over Σ^n, and thus $P_{\mathcal{M}}$ is not a probabilistic distribution over Σ^+.

Remark 1 We could have defined the labelling of an LMC as a probabilistic function $\mathcal{L} : Q \to \mathcal{D}(\Sigma)$ and then for $w = w_1 \dots w_k$, define $\Delta(X, w) = \sum_{\bar{q} = q_1 \dots q_k \in Q^k} \Delta(X, \bar{q}) \cdot \prod_{i=1}^{k} \mathcal{L}(q_i)(w_i)$. This is no more general than our definition. Given an LMC $\mathcal{M} = \langle Q, \pi_0, \Sigma, \mathcal{L}, \delta \rangle$ with $\mathcal{L} : Q \to \mathcal{D}(\Sigma)$, we can construct an LMC $\mathcal{M}' = \langle Q', \pi'_0, \Sigma, \mathcal{L}', \delta' \rangle$ that defines the same trace distribution as $\mathcal{M}$. Take $Q' = Q \times \Sigma$ and for all $\langle q, \sigma \rangle \in Q'$ define $\pi'_0(\langle q, \sigma \rangle) = \pi_0(q) \cdot \mathcal{L}(q)(\sigma)$, $\mathcal{L}'(\langle q, \sigma \rangle) = \sigma$ and $\delta'(\langle q, \sigma \rangle)(\langle q', \sigma' \rangle) = \delta(q)(q') \cdot \mathcal{L}(q')(\sigma')$ for all $\langle q, \sigma \rangle \in Q'$.

Probabilistic automata A *probabilistic automaton* is a tuple $\mathcal{A} = \langle S, \rho_0, \Sigma, M, F \rangle$ where

- $S = \{s_1, \ldots, s_n\}$ is a finite set of states;
- $\rho_0 \in \mathcal{D}(S)$ is the initial distribution of states;
- Σ is a finite alphabet of symbols;
- $M : \Sigma \to (S \times S \to [0, 1])$ assigns to each symbol $\sigma \in \Sigma$ of the alphabet a *stochastic matrix* M_σ (the entries of each row sum up to 1). In state s_i, when reading symbol σ, the successor state is s_j with probability $M_\sigma(s_i, s_j)$. We say that (s_i, σ, s_j) is a transition of $\mathcal{A}$ if $M_\sigma(s_i, s_j) \neq 0$.

  We extend $\mathcal{M}$ to finite words $w \in \Sigma^*$ as follows: M_ϵ is the $(n \times n)$ identity matrix, and inductively, $M_{\sigma w} = M_\sigma \cdot M_w$ for all $\sigma \in \Sigma$ and $w \in \Sigma^*$;
- $F \subseteq S$ is a set of accepting states.

Let η_F be the n-dimensional column vector such that $\eta[i] = 1$ if $s_i \in F$ and $\eta[i] = 0$ if $s_i \notin F$, for all $1 \leq i \leq n$. The probabilistic automaton $\mathcal{A}$ defines the *accepting trace distribution* $P_\mathcal{A} : \Sigma^* \to [0, 1]$ defined by $P_\mathcal{A}(w) = \rho_0 \cdot M_w \cdot \eta_F$, the probability that the word $w \in \Sigma^*$ is accepted by $\mathcal{A}$.

For computability issues, we assume that all the numbers appearing in LMC and probabilistic automata are rational, encoded as pairs of integers in binary. We assume that largest integer in Markov chains is bounded by a constant p. Without this assumption, the time complexity of our algorithm would involve a factor p^2 as the operations of addition, multiplication and comparison are quadratic over the rational numbers.

3. Bisimulation and equivalence

Definition 1 *Two probabilistic automata $\mathcal{A}_1, \mathcal{A}_2$ with alphabet Σ are* equivalent *if $P_{\mathcal{A}_1}(w) = P_{\mathcal{A}_2}(w)$ for all $w \in \Sigma^*$. Two LMC $\mathcal{M}_1, \mathcal{M}_2$ with alphabet Σ are* equivalent *if $P_{\mathcal{M}_1}(w) = P_{\mathcal{M}_2}(w)$ for all $w \in \Sigma^+$.*

A polynomial-time algorithm is known to decide equivalence of probabilistic automata [6]. We present an alternative algorithm to decide equivalence which is based on a new definition of bisimulation for probabilistic automata. Our algorithm has the same worst-case time complexity as in [6], namely $O((n_1 + n_2)^4)$ where n_i is the number of states of $\mathcal{A}_i$.

Then we present a linear-time transformation of labeled Markov chains to probabilistic automata that preserves the trace distribution. This shows that the algorithms for equivalence of probabilistic automata can also be used for checking equivalence of labeled Markov chains with the same complexity.

3.1. Equivalence is decidable for probabilistic automata

We define a natural generalization of classical bisimulation for probabilistic automata, as an equivalence relation over state distributions. It can be seen as the

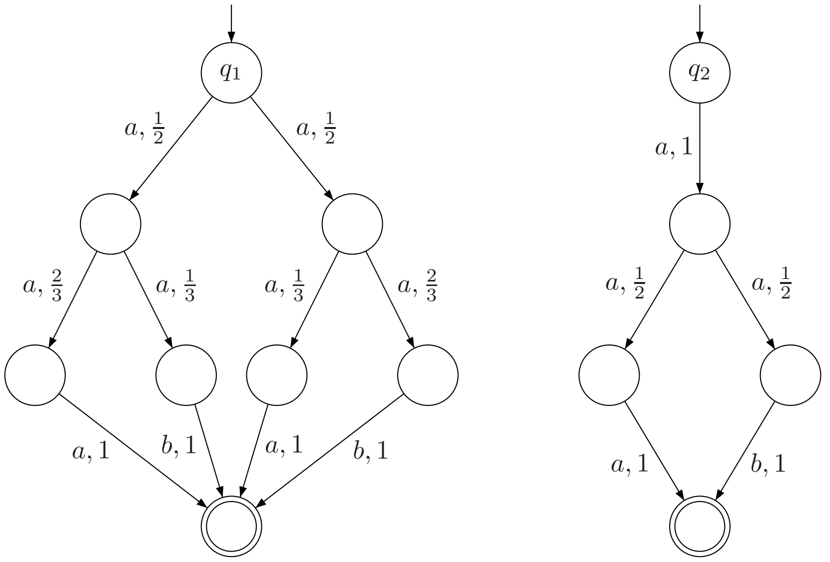

Figure 1: The states q_1, q_2 are not bisimilar according to [1, 2], but the Dirac-distributions X_{q_1} and X_{q_2} are bisimilar according to Definition 2.

classical bisimulation for a (non-probabilistic) infinite-state transition graph whose vertices are state distributions. The graph is deterministic and the successor by $\sigma \in \Sigma$ of a distribution $X \in \mathcal{D}(S)$ is the distribution $X \cdot M_\sigma$.

Definition 2 *A* bisimulation *for two probabilistic automata $\mathcal{A}_i = \langle S^i, \rho_0^i, \Sigma, M^i, F^i \rangle$ ($i = 1, 2$) is a relation $\approx \subseteq \mathcal{D}(S^1) \times \mathcal{D}(S^2)$ such that for all $X \in \mathcal{D}(S^1), Y \in \mathcal{D}(S^2)$ with $X \approx Y$, we have (i) $X \cdot \eta_{F^1} = Y \cdot \eta_{F^2}$ and (ii) $X \cdot M_\sigma^1 \approx Y \cdot M_\sigma^2$ for all $\sigma \in \Sigma$. We say that $\mathcal{A}_1$ and $\mathcal{A}_2$ are* bisimilar *if there exists a bisimulation for them containing (ρ_0^1, ρ_0^2).*

The classical definition of bisimulation for probabilistic processes relates single states rather than state distributions [1, 2]. There, a bisimulation for $\mathcal{A} = \langle S, \rho_0, \Sigma, M, F \rangle$ is the largest equivalence relation $\approx \subseteq S \times S$ such that $s \approx s'$ implies (i) $s \in F$ iff $s' \in F$ and (ii) for each symbol $\sigma \in \Sigma$ and each equivalence class C of $\approx$, the probability that the successor of s belongs to C is equal to the probability that the successor of s' belongs to C, that is $X_s \cdot M_\sigma \cdot \eta_C = X_{s'} \cdot M_\sigma \cdot \eta_C$. Notice that the automaton $\mathcal{A}$ can be seen as the disjoint union of automata $\mathcal{A}_1$ and $\mathcal{A}_2$ of Definition 2.

Our definition of bisimulation is weaker in the sense that whenever two states q_1, q_2 are bisimilar according to [1, 2], then the Dirac-distributions X_{q_1} and X_{q_2} are bisimilar according to Definition 2. Figure 1 shows an example over the alphabet $\Sigma = \{a, b\}$ where the reverse implication does not hold (the transitions that are not depicted lead to a rejecting sink state). The two automata define the same trace distribution, namely $P(aaa) = \frac{1}{2}$, $P(aab) = \frac{1}{2}$ and $P(w) = 0$ for all $w \notin \{aaa, aab\}$. We will see in Theorem 1 that this implies that they are bisimilar. However, it is easy to see that individual states q_1, q_2 are not bisimilar according to [1, 2].

Notice that the union of two bisimulations is again a bisimulation, and thus there always exists a unique maximal bisimulation for two probabilistic automata. We show that equivalence and bisimilarity coincide.

Theorem 1 *Two probabilistic automata are equivalent if and only if they are bisimilar.*

Proof. Let $\mathcal{A}_1 = \langle S^1, \rho_0^1, \Sigma, M^1, F^1 \rangle$ and $\mathcal{A}_2 = \langle S^2, \rho_0^2, \Sigma, M^2, F^2 \rangle$ be two probabilistic automata. First, assume that $\mathcal{A}_1$ and $\mathcal{A}_2$ are bisimilar, witnessed by the relation $\approx$. By definition of bisimilarity, since $\rho_0^1 \approx \rho_0^2$ we have $\rho_0^1 \cdot M_w^1 \approx \rho_0^2 \cdot M_w^2$ for all words $w \in \Sigma^*$, and therefore $\rho_0^1 \cdot M_w^1 \cdot \eta_{F^1} = \rho_0^2 \cdot M_w^2 \cdot \eta_{F^2}$, that is $P_{\mathcal{A}_1}(w) = P_{\mathcal{A}_2}(w)$.

Second, assume that $\mathcal{A}_1$ and $\mathcal{A}_2$ are equivalent. We show that the relation $\approx = \{(X, Y) \in \mathcal{D}(S^1) \times \mathcal{D}(S^2) \mid \forall w \in \Sigma^* : X \cdot M_w^1 \cdot \eta_{F^1} = Y \cdot M_w^2 \cdot \eta_{F^2}\}$ witnesses the bisimilarity of $\mathcal{A}_1$ and $\mathcal{A}_2$. Clearly $\rho_0^1 \approx \rho_0^2$, and for all $X \approx Y$, we have (i) $X \cdot \eta_{F^1} = Y \cdot \eta_{F^2}$ (take $w = \epsilon$ in the definition of $\approx$) and (ii) $X \cdot M_{\sigma w}^1 \cdot \eta_{F^1} = Y \cdot M_{\sigma w}^2 \cdot \eta_{F^2}$ for all $w \in \Sigma^*$ and $\sigma \in \Sigma$, so that $X \cdot M_\sigma^1 \cdot M_w^1 \cdot \eta_{F^1} = Y \cdot M_\sigma^2 \cdot M_w^2 \cdot \eta_{F^2}$, that is $X \cdot M_\sigma^1 \approx Y \cdot M_\sigma^2$. ■

By Theorem 1, we can solve the equivalence problem by computing the largest bisimulation. In the proof of Theorem 2, we give a fix-point algorithm to construct all the state distributions that are bisimilar.

Theorem 2 *The equivalence of two probabilistic automata can be decided in time $O((n_1+n_2)^4)$, where n_1 and n_2 are the respective number of states of the automata.*

Proof. Let $\mathcal{A}_1 = \langle S^1, \rho_0^1, \Sigma, M^1, F^1 \rangle$ and $\mathcal{A}_2 = \langle S^2, \rho_0^2, \Sigma, M^2, F^2 \rangle$ be two probabilistic automata with $n_i = |S^i|$ for $i = 1, 2$. We construct a formula $\phi^*(X, Y)$ over variables $X = (x_1, \dots, x_{n_1})$ and $Y = (y_1, \dots, y_{n_2})$ that characterizes all the bisimilar distributions for $\mathcal{A}_1$ and $\mathcal{A}_2$, that is such that $\{(\rho, \rho') \mid \phi(\rho, \rho') \text{ holds}\}$ is the largest bisimulation for $\mathcal{A}_1$ and $\mathcal{A}_2$. Let

$$\begin{aligned} \phi_0 &\equiv X \cdot \eta_{F^1} = Y \cdot \eta_{F^2} \\ \phi_{i+1} &\equiv \phi_i \wedge \bigwedge_{\sigma \in \Sigma} \phi_i(X \leftarrow X \cdot M_\sigma^1, Y \leftarrow Y \cdot M_\sigma^2) \text{ for all } i \geq 0. \end{aligned}$$

where $\phi_i(X \leftarrow X \cdot M_\sigma^1, Y \leftarrow Y \cdot M_\sigma^2)$ is the formula obtained from ϕ_i by substituting each variable x_j (resp. y_j) by the expression $\sum_i x_i \cdot M_\sigma^1(i,j)$ (resp. $\sum_i \cdot y_i M_\sigma^2(i,j)$).

Then, for all $i \geq 0$, either ϕ_{i+1} is equivalent to ϕ_i, or ϕ_{i+1} is the conjunction of ϕ_i and at least one equality constraint which is independent of the constraints in ϕ_i. Since there can be at most $n_1 + n_2$ independent linear equations over $n_1 + n_2$ variables, there must exists $j^* \leq n_1 + n_2$ such that ϕ_{j+1} is equivalent to ϕ_j for all $j \geq j^*$. We take

$$\phi^* = \phi_{j^*} \wedge \sum_i x_i = 1 \wedge \bigwedge_i x_i \geq 0 \wedge \sum_i y_i = 1 \wedge \bigwedge_i y_i \geq 0$$

Let us show that $\phi^*(\rho_0^1, \rho_0^2)$ holds if and only if $\mathcal{A}_1$ and $\mathcal{A}_2$ are bisimilar.

First, assume that $\phi^*(\rho_0^1, \rho_0^2)$ holds. Let $\rho \approx \rho'$ if and only if $\phi^*(\rho, \rho')$ holds, and let us show that $\approx$ is a bisimulation for $\mathcal{A}_1$ and $\mathcal{A}_2$. We have to show that $\rho \approx \rho'$ implies (i) $\rho \cdot \eta_{F^1} = \rho' \cdot \eta_{F^2}$ and (ii) $\rho \cdot M_\sigma^1 \approx \rho' \cdot M_\sigma^2$ holds for all $\sigma \in \Sigma$. We have seen above that $\phi^* \equiv \phi_{j^*} \equiv \phi_{j^*+1}$ with $j^* \leq n_1 + n_2$, and clearly ϕ_{j+1} implies ϕ_j for all $j \geq 0$. So, we have (i) $\rho \approx \rho'$ implies $\phi_0(\rho, \rho')$, *i.e.* $\rho \cdot \eta_{F^1} = \rho' \cdot \eta_{F^2}$ and (ii) $\rho \approx \rho'$ implies $\phi_{j^*}(X \leftarrow \rho \cdot M_\sigma^1, Y \leftarrow \rho' \cdot M_\sigma^2)$ for all $\sigma \in \Sigma$, which is $\phi^*(X \leftarrow \rho \cdot M_\sigma^1, Y \leftarrow \rho' \cdot M_\sigma^2)$ and thus $\rho \cdot M_\sigma^1 \approx \rho' \cdot M_\sigma^2$.

Second, to show that $\{(\rho, \rho') \mid \phi(\rho, \rho') \text{ holds}\}$ is the largest bisimulation, assume that $\approx$ is a bisimulation for $\mathcal{A}_1$ and $\mathcal{A}_2$ and let us show by induction on i that $\rho \approx \rho'$ implies $\phi_i(\rho, \rho')$ for all $i \geq 0$. By definition of bisimulation, $\rho \approx \rho'$ implies $\rho \cdot \eta_{F^1} = \rho' \cdot \eta_{F^2}$, that is $\phi_0(\rho, \rho')$. Assume by induction that $\rho \approx \rho'$ implies $\phi_i(\rho, \rho')$ for all $i \leq k$ and show that $\rho \approx \rho'$ implies $\phi_{k+1}(\rho, \rho')$. Since $\phi_{k+1}(\rho, \rho') \equiv \phi_k(\rho, \rho') \wedge \bigwedge_{\sigma \in \Sigma} \phi_i(\rho \cdot M_\sigma^1, \rho' \cdot M_\sigma^2)$, we have to show that (i) $\rho \approx \rho'$ implies $\phi_k(\rho, \rho')$ which is entailed by the induction hypothesis, and (ii) for all $\sigma \in \Sigma$, $\rho \approx \rho'$ implies $\phi_k(\rho \cdot M_\sigma^1, \rho' \cdot M_\sigma^2)$ which is entailed by the induction hypothesis and the fact that $\rho \cdot M_\sigma^1 \approx \rho' \cdot M_\sigma^2$.

In practice, we eliminate from each ϕ_i the equality constraints that are redundant (*i.e.* that are linear combination of the others) and thus we have at most $n_1 + n_2$ constraints in each ϕ_i. Since $(n \times n)$-matrix multiplication and linear independence checking can be done in time $O(n^3)$, computing ϕ^* can be done in time $O((n_1+n_2)^4)$. Finally, we check if $\phi^*(\rho_0^1, \rho_0^2)$ holds, that is whether $\rho_0^1 \approx \rho_0^2$ which is equivalent to check equivalence of $\mathcal{A}_1$ and $\mathcal{A}_2$ by Theorem 1. ■

Notice that the formula ϕ^* defined in the above proof characterizes all the initial distributions that make the two automata equivalent. Hence, to solve the equivalence problem for the same automata with different initial distributions $\rho_0^{1'}$ and $\rho_0^{2'}$, we only need to check that $\phi^*(\rho_0^{1'}, \rho_0^{2'})$ holds which is done in $O((n_1 + n_2)^2)$. This is in contrast with the algorithm of [6] that keeps the same complexity.

3.2. Transformation of labeled Markov chains to probabilistic automata

Theorem 3 *For every labeled Markov chain $\mathcal{M}$ over Σ, we can construct in linear time a probabilistic automaton $\mathcal{A}_\mathcal{M}$ such that $P_\mathcal{A}(w) = P_{\mathcal{A}_\mathcal{M}}(w)$ for all $w \in \Sigma^+$.*

Proof Let $\mathcal{M} = \langle Q, \pi_0, \Sigma, \mathcal{L}, \delta \rangle$. We construct $\mathcal{A}_\mathcal{M} = \langle S, \rho_0, \Sigma, M, F \rangle$ as follows:

- $S = Q \cup \{\mathsf{sink}\}$;
- $\rho_0(\mathsf{sink}) = 0$ and $\rho_0(q) = \pi_0(q)$ for all $q \in Q$;
- For each $\sigma \in \Sigma$ and $q \in Q$, let $M_\sigma(\mathsf{sink}, q) = 0$ and $M_\sigma(\mathsf{sink}, \mathsf{sink}) = 1$; for each $q, q' \in Q$ and $\sigma \in \Sigma$, let

$$M_\sigma(q, q') = \begin{cases} \delta(q)(q') & \text{if } \sigma = \mathcal{L}(q) \\ 1 & \text{if } \sigma \neq \mathcal{L}(q) \text{ and } q' = \mathsf{sink} \\ 0 & \text{otherwise} \end{cases}$$

- $F = Q$; ■

By Theorems 2 and 3, we derive the following corollary.

Corollary 1 *The equivalence of two LMC can be decided in time $O((n_1+n_2)^4)$ where n_1, n_2 are the respective number of states of the LMCs.*

Remark 2 Several other models of probabilistic finite-state systems exist that can be transformed in into equivalent probabilistic automata, which also provide equivalence algorithms for these models. This is the case for instance for the probabilistic finite-state automata of [7] and the hidden Markov chains of [8].

Example Consider the labeled Markov chains shown on Figure 2. The set of observations is $\Sigma = \{A, B, C\}$, and the parameters $\lambda, \mu_1, \mu_2, \nu$ are fixed real numbers in the interval $[0, 1]$. Using the proof of Theorem 2, we compute below the largest bisimulation between the two probabilistic automata obtained by the transformation of Theorem 3 (remember that there is just one additional rejecting sink state, all the other states being accepting). It is easy to see that we can omit the variables x_{sink} and y_{sink} corresponding to the sink states since their value must be 0 (see the proof of Theorem 3). Let $X = (x_0, x_1, x_2, x_3, x_4)$ and $Y = (y_0, y_1, y_2, y_3, y_4)$. Then,

$$\begin{array}{l}
X \cdot M^1_A = (0, \lambda x_0, (1-\lambda)x_0, (1-\mu_1)x_1 + \mu_2 x_2, \mu_1 x_1 + (1-\mu_2)x_2) \\
X \cdot M^1_B = (0, 0, 0, x_3, 0) \\
X \cdot M^1_C = (0, 0, 0, 0, x_4) \\
Y \cdot M^2_A = (0, \nu y_0, (1-\nu)y_0, y_1, y_2) \\
Y \cdot M^2_B = (0, 0, 0, y_3, 0) \\
Y \cdot M^2_C = (0, 0, 0, 0, y_4)
\end{array}$$

So, we have

$$\begin{array}{rcl}
\phi_0(X,Y) & \equiv & x_0 + x_1 + x_2 + x_3 + x_4 = y_0 + y_1 + y_2 + y_3 + y_4 \\
\phi_1(X,Y) & \equiv & \phi_0(X,Y) \wedge \bigwedge_{\sigma \in \{A,B,C\}} \phi_0(X \cdot M^1_\sigma, Y \cdot M^2_\sigma) \\
& \equiv & \left\{ \begin{array}{l} x_0 + x_1 + x_2 = y_0 + y_1 + y_2 \\ x_3 = y_3 \\ x_4 = y_4 \end{array} \right. \\
\phi_2(X,Y) & \equiv & \phi_1(X,Y) \wedge \phi_1(X \cdot M^1_A, Y \cdot M^2_A) \wedge \underbrace{\bigwedge_{\sigma \in \{B,C\}} \phi_1(X \cdot M^1_\sigma, Y \cdot M^2_\sigma)}_{\text{implied by } \phi_1(X,Y)} \\
& \equiv & \left\{ \begin{array}{l} x_0 + x_1 + x_2 = y_0 + y_1 + y_2 \\ x_3 = y_3 \\ x_4 = y_4 \\ x_0 = y_0 \\ (1-\mu_1)x_1 + \mu_2 x_2 = y_1 \\ \mu_1 x_1 + (1-\mu_2)x_2 = y_2 \end{array} \right.
\end{array}$$

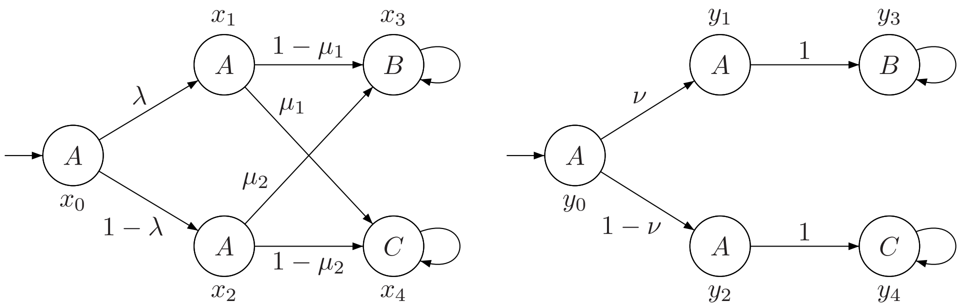

Figure 2: The LMCs $\mathcal{M}_1$ and $\mathcal{M}_2$ are equivalent iff $(1-\mu_1)\lambda + \mu_2(1-\lambda) = \nu$.

$$
\begin{aligned}
&\equiv \left\{\begin{array}{l} x_3 = y_3 \\ x_4 = y_4 \\ x_0 = y_0 \\ (1-\mu_1)x_1 + \mu_2 x_2 = y_1 \\ \mu_1 x_1 + (1-\mu_2)x_2 = y_2 \end{array}\right. \\
\phi_3(X,Y) &\equiv \phi_2(X,Y) \wedge \phi_2(X \cdot M_A^1, Y \cdot M_A^2) \wedge \underbrace{\bigwedge_{\sigma \in \{B,C\}} \phi_1(X \cdot M_\sigma^1, Y \cdot M_\sigma^2)}_{\text{implied by } \phi_2(X,Y)} \\
&\equiv \left\{\begin{array}{l} x_3 = y_3 \\ x_4 = y_4 \\ (1-\mu_1)x_1 + \mu_2 x_2 = y_1 \\ \mu_1 x_1 + (1-\mu_2)x_2 = y_2 \\ (1-\mu_1)\lambda x_0 + \mu_2(1-\lambda)x_0 = \nu y_0 \\ \mu_1 \lambda x_0 + (1-\mu_2)(1-\lambda)x_0 = (1-\nu)y_0 \end{array}\right.
\end{aligned}
$$

It is easy to check that $\phi_4(X,Y) \equiv \phi_3(X,Y)$. For $X_0 = (1,0,0,0,0) = Y_0$, we have $X_0 \approx Y_0$ if and only if $(1-\mu_1)\lambda + \mu_2(1-\lambda) = \nu$.

3.3. Equivalence for weighted probabilistic automata

Weighted automata have numerical weights on transitions [5]. The weight can be interpreted as the amount of some resource that the system needs in order to perform a transition, *e.g.*, memory consumption or power consumption. The value of a finite path in the automaton can be defined as the Max of the transition weights on the path, corresponding to the peak consumption of the resource.

In a weighted probabilistic automaton, the value of a finite word w is the expected value of all paths labeled by w, defining a quantitative language, *i.e.* a mapping $\Sigma^+ \to \mathbb{R}$.

Formally, a *weighted probabilistic automaton* is a tuple $\mathcal{A} = \langle S, \rho_0, \Sigma, M, F, \gamma \rangle$ where

- $\langle S, \rho_0, \Sigma, M, F \rangle$ is a probabilistic automaton;

- $\gamma : Q \times \Sigma \times Q \to \mathbb{Q}$ is a weight function, where $\mathbb{Q}$ is the set of rational numbers.

For each finite word $w = \sigma_1 \dots \sigma_n \in \Sigma^+$, let $\mathsf{Path}(w)$ be the set of *paths* $\pi = s_0\sigma_1 s_1\sigma_1 \dots \sigma_n s_n$ where $s_i \in S$ for all $0 \le i \le n$ and $s_n \in F$. The probability of such a path is $P(\pi) = \rho_0(s_0) \cdot \prod_{i=1}^{n} M_{\sigma_i}(s_{i-1}, s_i)$ and its value is $\gamma(\pi) = \max\{\gamma(s_{i-1}, \sigma_i, s_i) \mid 1 \le i \le n\}$. The *quantitative language* of $\mathcal{A}$ assigns to each word $w \in \Sigma^+$ the expected value of the paths over w:

$$L_{\mathcal{A}}(w) = \sum_{\pi \in \mathsf{Path}(w)} P(\pi) \cdot \gamma(\pi)$$

For technical reasons, we define $L_{\mathcal{A}}(\epsilon) = \sum_{s \in F} \rho_0(s)$, that is $L_{\mathcal{A}}(\epsilon)$ is the probability that the empty word ϵ is accepted by $\mathcal{A}$.

Definition 3 *Two weighted probabilistic automata $\mathcal{A}_1, \mathcal{A}_2$ with alphabet Σ are* equivalent *if $L_{\mathcal{A}_1}(w) = L_{\mathcal{A}_2}(w)$ for all $w \in \Sigma^*$.*

We can decide if two weighted probabilistic automata are equivalent using the technique of Section 3.1. We need a slightly different notion of bisimulation, based on a (non-probabilistic) infinite-state transition graph whose vertices are probability distributions over pairs (s, v) of states s and weight v. In this graph, the distribution X is reached after reading a word w if for all states s and weight v, the probability is $X(s, v)$ to reach in $\mathcal{A}$ the state s through a path whose maximal weight is v. The successor of X by σ is therefore the distribution X' such that for all states s' and weight v', we have

$$\begin{aligned} X'(s', v') \quad &= \sum_{s \mid \gamma(s,\sigma,s') < v'} X(s, v') \cdot M_\sigma(s, s') \\ &+ \sum_{s \mid \gamma(s,\sigma,s') = v'} \; \sum_{v \le v'} X(s, v) \cdot M_\sigma(s, s') \end{aligned}$$

We denote the distribution X' by $\delta(X, \sigma)$. Intuitively, we reach s' with weight v' when reading σ if either we start from a state s with weight v' and the weight of the transition (s, σ, s') is less than v', or we start from a state s with weight at most v' and the weight of the transition (s, σ, s') is v'.

Now, we define bisimulation for two weighted probabilistic automata $\mathcal{A}_i = \langle S^i, \rho_0^i, \Sigma, M^i, F^i, \gamma_i \rangle$ $(i = 1, 2)$. For $i = 1, 2$, let V_i be the set of weights that occur in $\mathcal{A}_i$ and assume without loss of generality that the minimal weight $v_{\min}$ of $V_1 \cup V_2$ belongs to $V_1 \cap V_2$.

Definition 4 *A* bisimulation *for two weighted probabilistic automata $\mathcal{A}_1$ and $\mathcal{A}_2$ is a relation $\approx \subseteq \mathcal{D}(S^1 \times V_1) \times \mathcal{D}(S^2 \times V_2)$ such that for all $X \approx Y$, we have (i) $\sum_{s_1 \in F^1} \sum_{v_1 \in V_1} v_1 \cdot X(s_1, v_1) = \sum_{s_2 \in F^2} \sum_{v_2 \in V_2} v_2 \cdot Y(s_2, v_2)$ and (ii) $\delta^1(X, \sigma) \approx \delta^2(X, \sigma)$ for all $\sigma \in \Sigma$. We say that $\mathcal{A}_1$ and $\mathcal{A}_2$ are* bisimilar *if there exists a bisimulation for them containing (θ_0^1, θ_0^2) where for $i = 1, 2$, $\theta_0^i(s_i, v_i) = \rho_0^i(s_i)$ if $v_i = v_{\min}$, and $\theta_0^i(s_i, v_i) = 0$ otherwise.*

By very similar arguments as for Theorems 1 and 2, it is easy to establish the following result.

Theorem 4 *Two weighted probabilistic automata are equivalent if and only if they are bisimilar. The equivalence of two weighted probabilistic automata can be decided in time $O((n_1 \cdot m_1 + n_2 \cdot m_2)^4)$, where n_1, n_2 and m_1, m_2 are the respective number of states and transitions of the automata.*

The complexity bound follows from the fact that we need $|S^1| \cdot |V_1| + |S^2| \cdot |V_2|$ variables to encode a pair (X_1, X_2) of distributions $X_i \in \mathcal{D}(S^i \times V_i)$ $(i = 1, 2)$.

4. Markov decision processes

Labeled Markov decision processes A *labeled Markov decision process* (LMDP) over Σ is a tuple $\mathcal{P} = \langle Q, \pi_0, \Sigma, \mathcal{L}, \Gamma, \delta \rangle$ where

- $Q, \pi_0, \Sigma, \mathcal{L}$ are defined as for labeled Markov chains;
- Γ is a finite set of moves;
- $\delta : Q \times \Gamma \to \mathcal{D}(Q)$ is a probabilistic transition function labeled by moves.

A *scheduler* for $\mathcal{P}$ is a function $\lambda : Q^+ \to \mathcal{D}(\Gamma)$. The scheduler λ for $\mathcal{P}$ defines an infinite-state labeled Markov chain $\mathcal{P}(\lambda) = \langle Q^+, \pi'_0, \Sigma, \mathcal{L}', \hat{\delta} \rangle$ where

- $\mathcal{L}'(q_1 q_2 \dots q_n) = \mathcal{L}(q_n)$;
- $\pi'_0(q_1 q_2 \dots q_n) = 0$ if $n \geq 2$, and $\pi'_0(q) = \pi_0(q)$;
- for all $\bar{q}, \bar{q}' \in Q^+$, if $\bar{q} = q_0 q_1 \dots q_n$ and $\bar{q}' = q_0 q_1 \dots q_{n+1}$ then $\hat{\delta}(\bar{q})(\bar{q}') = \sum_{\gamma \in \Gamma} \lambda(\bar{q})(\gamma) \cdot \delta(q_n, \gamma)(q_{n+1})$, and otherwise $\hat{\delta}(\bar{q})(\bar{q}') = 0$.

Given a scheduler λ for $\mathcal{P}$ and a finite sequence $\bar{q} = q_1 \dots q_k$ of states, define the scheduler $\lambda_{[\bar{q}]}$ for $\mathcal{P}$ by $\lambda_{[\bar{q}]}(\bar{q}')(\gamma) = \lambda(\bar{q}.\bar{q}')(\gamma)$ for all $\bar{q}' \in Q^+$ and $\gamma \in \Gamma$. That is $\lambda_{[\bar{q}]}$ is playing like λ under history $\bar{q}$.

Now, define $\delta_\lambda : Q \to \mathcal{D}(Q)$ by $\delta_\lambda(q)(q') = \sum_{\gamma \in \Gamma} \lambda(q)(\gamma) \cdot \delta(q, \gamma)(q')$ for all $q, q' \in Q$. Define $\Delta_\lambda : \mathcal{D}(Q) \times Q^+ \to [0, 1]$, the function that gives the probability $\Delta_\lambda(X, \bar{q})$ of observing the sequence of states $\bar{q}$ when $\mathcal{P}(\lambda)$ is started with the initial distribution X as follows: $\Delta_\lambda(X, q) = X(q)$ and inductively, $\Delta_\lambda(X, \bar{q}.q'.q'') = \Delta_\lambda(X, \bar{q}.q') \cdot \delta_{\lambda_{[\bar{q}]}}(q')(q'')$ for all $\bar{q} \in Q^*$ and $q', q'' \in Q$. We extend Δ_λ to nonempty sequences of observations as follows: for all $w \in \Sigma^+$, let $\Delta_\lambda(X, w) = \sum_{\{\bar{q} \mid \mathcal{L}(\bar{q}) = w\}} \Delta_\lambda(X, \bar{q})$.

We say that a scheduler λ has *finite memory* if there exists a finite set M (the memory), a memory cell $m_0 \in M$, and two functions *update* $: M \times Q \to M$ and $\mu : M \to \mathcal{D}(\Gamma)$ such that $\lambda(\bar{q}) = \mu(\mathit{update}(m_0, \bar{q}))$ for all $\bar{q} \in Q^+$, where we extend *update* to sequences of states as follows: $\mathit{update}(m, \bar{q}.q') = \mathit{update}(\mathit{update}(m, \bar{q}), q')$ for all $m \in M$, $\bar{q} \in Q^+$ and $q' \in Q$.

Refinement Given an LMC $\mathcal{M}$ and two LMDPs $\mathcal{P}_1$, $\mathcal{P}_2$, we write

- $\mathcal{M} \sqsubseteq \mathcal{P}_1$ if there exists a scheduler λ for $\mathcal{P}_1$ such that $\mathcal{M}$ and $\mathcal{P}_1(\lambda)$ are equivalent;

- $\mathcal{P}_1 \sqsubseteq \mathcal{M}$ if $\mathcal{M}$ and $\mathcal{P}_1(\lambda)$ are equivalent for all schedulers λ for $\mathcal{P}_1$;
- $\mathcal{P}_1 \sqsubseteq \mathcal{P}_2$ if for all schedulers λ for $\mathcal{P}_1$, there exists a scheduler μ for $\mathcal{P}_2$ such that $\mathcal{P}_1(\lambda)$ and $\mathcal{P}_2(\mu)$ are equivalent; if so, we say that $\mathcal{P}_1$ refines $\mathcal{P}_2$.
- $\mathcal{P}_1 \equiv \mathcal{P}_2$ if $\mathcal{P}_1 \sqsubseteq \mathcal{P}_2$ and $\mathcal{P}_2 \sqsubseteq \mathcal{P}_1$, and we say that $\mathcal{P}_1$ and $\mathcal{P}_2$ are equivalent.

The decidability of all the above relations is open. We show below that only the schedulers that are observation-based need to be considered in the refinement and equivalence relations. We also give an insight of the difficulty of these problems by showing that infinite memory can be needed for a scheduler to establish for instance $\mathcal{M} \sqsubseteq \mathcal{P}_1$.

Remark 3 - Observation-based schedulers are sufficient. A scheduler λ for $\mathcal{P}$ is *observation-based* if $\lambda(\rho.q) = \lambda(\rho'.q)$ for all $q \in Q$ and $\rho, \rho' \in Q^*$ such that $\mathcal{L}(\rho) = \mathcal{L}(\rho')$.

Lemma 1 *Given a scheduler λ for an LMDP $\mathcal{P}$, there exists an observation-based scheduler μ for $\mathcal{P}$ such that the LMCs $\mathcal{P}(\lambda)$ and $\mathcal{P}(\mu)$ are equivalent.*

Proof. Let $\mathcal{P} = \langle Q, \pi_0, \Sigma, \mathcal{L}, \Gamma, \delta \rangle$. Define the observation-based scheduler μ as follows, for all $\bar{q}_1 \in Q^*$, $q \in Q$ and $\gamma \in \Gamma$:

$$\mu(\bar{q}_1.q)(\gamma) = \sum_{\{\bar{q}_2 \mid \mathcal{L}(\bar{q}_2) = \mathcal{L}(\bar{q}_1)\}} c_{\bar{q}_2.q} \cdot \lambda(\bar{q}_2.q)(\gamma)$$

$$\text{where } c_{\bar{q}_2.q} = \frac{\Delta_\lambda(\pi_0, \bar{q}_2.q)}{\sum_{\{\bar{q}_3 \mid \mathcal{L}(\bar{q}_3) = \mathcal{L}(\bar{q}_2)\}} \Delta_\lambda(\pi_0, \bar{q}_3.q)}$$

Clearly, μ is a well-defined scheduler since the coefficients $c_{\bar{q}_2.q}$ sum up to 1 when $\bar{q}_2$ is such that $\mathcal{L}(\bar{q}_2) = \mathcal{L}(\bar{q}_1)$. Let us show by induction on the length of w that for all $q \in Q$, we have[a] $\Delta_\lambda(\pi_0, w.q) = \Delta_\mu(\pi_0, w.q)$ for all $w \in \Sigma^*$. This will immediately imply that $\Delta_\lambda(\pi_0, w) = \Delta_\mu(\pi_0, w)$ for all $w \in \Sigma^+$ and thus $\mathcal{P}(\lambda)$ and $\mathcal{P}(\mu)$ are equivalent.

First, for the empty word $w = \epsilon$, we have $\Delta_\lambda(\pi_0, \epsilon.q) = \pi_0(q) = \Delta_\mu(\pi_0, \epsilon.q)$. Second, assume that $\Delta_\lambda(\pi_0, w.q) = \Delta_\mu(\pi_0, w.q)$ holds for all $q \in Q$ and $w \in \Sigma^*$ of length $\leq k$, and let $w' = w\sigma$ be a word of length $k+1$. We have:

$$\begin{aligned}
\Delta_\lambda(\pi_0, w'.q) &= \sum_{\{\bar{q}_1 \mid \mathcal{L}(\bar{q}_1) = w'\}} \Delta_\lambda(\pi_0, \bar{q}_1.q) \\
&= \sum_{\{\bar{q}_2 \mid \mathcal{L}(\bar{q}_2) = w\}} \sum_{\{q' \mid \mathcal{L}(q') = \sigma\}} \sum_{\gamma \in \Gamma} \Delta_\lambda(\pi_0, \bar{q}_2.q') \cdot \lambda(\bar{q}_2.q')(\gamma) \cdot \delta(q', \gamma)(q) \\
&= \sum_{\{q' \mid \mathcal{L}(q') = \sigma\}} \sum_{\gamma \in \Gamma} \delta(q', \gamma)(q) \cdot \sum_{\{\bar{q}_2 \mid \mathcal{L}(\bar{q}_2) = w\}} \Delta_\lambda(\pi_0, \bar{q}_2.q') \cdot \lambda(\bar{q}_2.q')(\gamma)
\end{aligned}$$

[a] We denote by $\Delta_\lambda(\pi_0, w.q)$ the probability of observing a sequence $q_1 \dots q_k$ with $w = \mathcal{L}(q_1 \dots q_{k-1})$ and $q_k = q$ when $\mathcal{P}(\lambda)$ is started in distribution π_0.

$$= \sum_{\{q' \mid \mathcal{L}(q')=\sigma\}} \sum_{\gamma\in\Gamma} \delta(q',\gamma)(q) \cdot \left(\sum_{\{\bar{q}_2 \mid \mathcal{L}(\bar{q}_2)=w\}} \Delta_\lambda(\pi_0, \bar{q}_2.q') \right) \cdot \mu(\bar{q}'_2.q')(\gamma)$$

for any sequence $\bar{q}'_2$ such that $\mathcal{L}(\bar{q}'_2) = w$ since μ is observation-based.

$$= \sum_{\{q' \mid \mathcal{L}(q')=\sigma\}} \sum_{\gamma\in\Gamma} \delta(q',\gamma)(q) \cdot \Big(\Delta_\lambda(\pi_0, w.q') \Big) \cdot \mu(\bar{q}'_2.q')(\gamma)$$

$$= \sum_{\{q' \mid \mathcal{L}(q')=\sigma\}} \sum_{\gamma\in\Gamma} \delta(q',\gamma)(q) \cdot \Big(\Delta_\mu(\pi_0, w.q') \Big) \cdot \mu(\bar{q}'_2.q')(\gamma)$$

(by the induction hypothesis)

$$= \sum_{\{q' \mid \mathcal{L}(q')=\sigma\}} \sum_{\gamma\in\Gamma} \delta(q',\gamma)(q) \cdot \sum_{\{\bar{q}_2 \mid \mathcal{L}(\bar{q}_2)=w\}} \Delta_\mu(\pi_0, \bar{q}_2.q') \cdot \mu(\bar{q}_2.q')(\gamma)$$

$$= \Delta_\mu(\pi_0, w'.q)$$

■

Remark 4 - Infinite memory may be necessary. Consider the example in Figure 3 showing a 4-states LMDP $\mathcal{P}$ with one nondeterministic choice in state x_2, and a 6-states LMC $\mathcal{M}$. A scheduler for this LMDP should give the value $\lambda(w)$ of the parameter λ for each sequence of observations $w \in \Sigma^+$ ending with symbols BC. We claim that a scheduler needs infinite memory for witnessing the relation $\mathcal{M} \sqsubseteq \mathcal{P}$ (for $x_0 = y_0 = 1$).

Lemma 2 *There exists an LMDP $\mathcal{P}$ and an LMC $\mathcal{M}$ such that $\mathcal{M}$ and $\mathcal{P}(\lambda)$ are equivalent for some scheduler λ, but there is no scheduler λ' with finite memory such that $\mathcal{M}$ and $\mathcal{P}(\lambda')$ are equivalent.*

Intuitively, whenever we observe two consecutive 'C', we know for sure that we have to simulate the y_2, y_4, y_5-component of the LMC, and so we play $\lambda = \frac{1}{2}$. On the other hand, if we observe only 'BC' sequences, we cannot surely discriminate between the y_1, y_3-component and the y_2, y_4, y_5-component. However, when the number n of 'BC' sequences increases, it becomes more likely that we should simulate the y_1, y_3-component. Therefore, we should increase the probability of playing as in the x_1, x_3-component, that is take $\lambda \to 0$ when $n \to \infty$. More precisely,

- if w contains two consecutive 'C', then play $\lambda(w) = \frac{1}{2}$;
- otherwise, let n be the number of occurrences of the sequence 'BC' in w. Play $\lambda(w) = \frac{1}{2+2^n}$;

We show that there exists no finite memory scheduler λ such that $\mathcal{M}$ and $\mathcal{P}(\lambda)$ are equivalent. To reach a contradiction, assume that such a finite memory scheduler exists, defined by M, $m_0 \in M$, *update* and μ. Consider the sequence $m_i = \mathit{update}(m_0, x_0(x_1x_2)^i)$ for $i = 1, 2, \ldots$. Since M is finite, there must exist $n_1 < n_2$ such that $m_{n_1} = m_{n_2}$. Let λ_i be the probability given by $\mu(m_i)$ to jump from x_2 to x_3, and let $p_1 = \prod_{i=1}^{n_1-1} 1 - \lambda_i$ be the probability to observe the sequence $A(BC)^{n_1}$, and let $p_2 = \prod_{i=n_1}^{n_2-1} 1 - \lambda_i$. If $p_2 = 1$, then the word

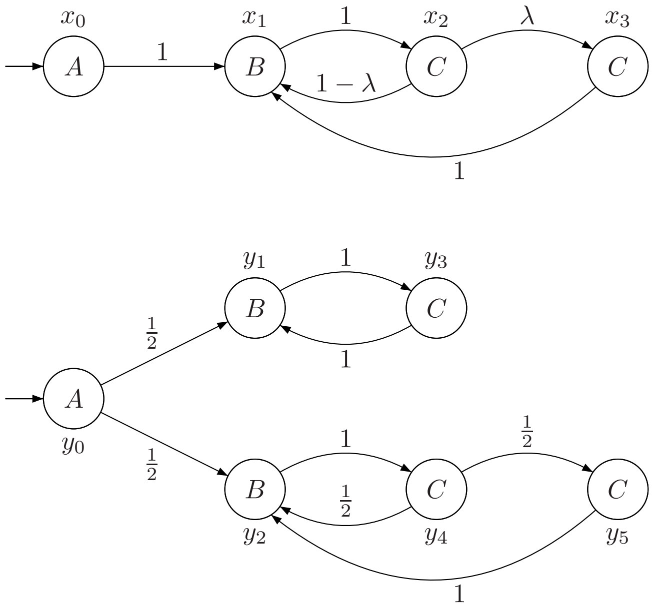

Figure 3: Infinite memory is required for the LMDP (above) to be equivalent to the LMC (below).

$A(BC)^{n_1}C$ has probability 0 in $\mathcal{P}(\lambda)$, and probability $\frac{1}{2^{1+n_1}} \neq 0$ in $\mathcal{M}$. Therefore $p_2 < 1$. Now, consider the infinite word[b] $w = A(BC)^\omega$. We have $P_{\mathcal{M}}(w) = \frac{1}{2}$, while $P_{\mathcal{P}(\lambda)}(w) = \lim_{n\to\infty} p_1.p_2^n = 0$. Hence the finite memory scheduler λ cannot exist.

Finally, notice that if we allow randomization in the memory update, that is $\mathit{update} : M \times Q \to \mathcal{D}(M)$, then a finite memory scheduler exists: initially, we take $m \in \{0, 1\}$ uniformly at random. Next, we play $\lambda = 0$ forever if $m = 0$, and $\lambda = \frac{1}{2}$ if $m = 1$. We do not know if finite randomized memory is sufficient in general.

5. Conclusion

We gave a new algorithm to decide in polynomial time the equivalence of labeled Markov chains. The algorithm is based on the computation of bisimilar state distributions. The refinement and equivalence problems for LMDPs remains open. Another important open problem about labeled Markov chains is the question of minimization, that is, to find an efficient procedure to construct an LMC with the minimal number of states which is equivalent to a given LMC. The question of the existence of a unique minimal labeled Markov chain (up to isomorphism) is also open.

[b] Given a function $f : \Sigma^* \to [0, 1]$, we write $f(w) = a$ for $w \in \Sigma^\omega$ if $\forall \epsilon > 0 \cdot \exists n : |f(w_{[n]}) - a| \leq \epsilon$ where $w_{[n]}$ is the prefix of w of length n.

References

1. Hans Hansson. *Time and Probability in Formal Design of Distributed Systems*. PhD thesis, Uppsala University, 1991.
2. Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991.
3. Azaria Paz. *Introduction to probabilistic automata*. Computer Science and Applied Mathematics. Academic Press, New York, 1971.
4. Michael Oser Rabin. Probabilistic automata. *Information and Control*, 6(3):230–245, 1963.
5. Marcel Paul Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2-3):245–270, 1961.
6. Wen-Guey Tzeng. A polynomial-time algorithm for the equivalence of probabilistic automata. *SIAM J. Comput.*, 21(2):216–227, 1992.
7. Enrique Vidal, Franck Thollard, Colin de la Higuera, Francisco Casacuberta, and Rafael C. Carrasco. Probabilistic finite-state machines-part I. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(7):1013–1025, 2005.
8. Enrique Vidal, Franck Thollard, Colin de la Higuera, Francisco Casacuberta, and Rafael C. Carrasco. Probabilistic finite-state machines-part II. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(7):1026–1039, 2005.