On-the-Fly Exact Computation of Bisimilarity Distances[⋆]

Giorgio Bacci[1,2], Giovanni Bacci[2], Kim G. Larsen[2], and Radu Mardare[2]

[1] Dept. of Mathematics and Computer Science, University of Udine, Italy
giorgio.bacci@uniud.it

[2] Department of Computer Science, Aalborg University, Denmark
{grbacci,giovbacci,kgl,mardare}@cs.aau.dk

Abstract. This paper proposes an algorithm for exact computation of bisimilarity distances between discrete-time Markov chains introduced by Desharnais et. al. Our work is inspired by the theoretical results presented by Chen et. al. at FoSSaCS'12, proving that these distances can be computed in polynomial time using the ellipsoid method. Despite its theoretical importance, the ellipsoid method is known to be inefficient in practice. To overcome this problem, we propose an efficient on-the-fly algorithm which, unlike other existing solutions, computes exactly the distances between given states and avoids the exhaustive state space exploration. It is parametric in a given set of states for which we want to compute the distances. Our technique successively refines over-approximations of the target distances using a greedy strategy which ensures that the state space is further explored only when the current approximations are improved. Tests performed on a consistent set of (pseudo)randomly generated Markov chains shows that our algorithm improves, on average, the efficiency of the corresponding iterative algorithms with orders of magnitude.

1 Introduction

Probabilistic bisimulation for Markov chains (MCs), introduced by Larsen and Skou [12], is the key concept for reasoning about the equivalence of probabilistic systems. However, when one focuses on quantitative behaviours it becomes obvious that such an equivalence is too "exact" for many purposes as it only relates processes with identical behaviours. In various applications, such as systems biology [15], games [3], planning [6] or security [2], we are interested in knowing whether two processes that may differ by a small amount in the real-valued parameters (probabilities) have "sufficiently" similar behaviours. This motivated the development of the metric theory for MCs, initiated by Desharnais et al. [8] and greatly developed and explored by van Breugel, Worrell and others [17,16]. It consists in proposing a *bisimilarity distance (pseudometric)*, which measures the

[⋆] Work supported by Sapere Aude: DFF-Young Researchers Grant 10-085054 of the Danish Council for Independent Research, by the VKR Center of Excellence MT-LAB and by the Sino-Danish Basic Research Center IDEA4CPS.

behavioural similarity of two MCs. The pseudometric proposed by Desharnais et al. is parametric in a *discount factor* $\lambda \in (0,1]$ that controls the significance of the future in the measurement.

Since van Breugel et. al. have presented a fixed point characterization of the aforementioned pseudometric in [16], several iterative algorithms have been developed in order to compute its approximation up to any degree of accuracy [9,17,16]. Recently, Chen et. al. [4] proved that, for finite MCs with rational transition function, the bisimilarity pseudometrics can be computed exactly in polynomial time. The proof consists in describing the pseudometric as the solution of a linear program that can be solved using the *ellipsoid method*. Although the ellipsoid method is theoretically efficient, *"computational experiments with the method are very discouraging and it is in practice by no means a competitor of the, theoretically inefficient, simplex method"*, as stated in [14]. Unfortunately, in this case the simplex method cannot be used to speed up performances in practice, since the linear program to be solved may have an exponential number of constraints in the number of states of the MC.

In this paper, we propose an alternative approach to this problem, which allows us to compute the pseudometric exactly and efficiently in practice. This is inspired by the characterization of the undiscounted pseudometric using *couplings*, given in [4], which we extend to generic discount factors. A coupling for a pair of states of a given MC is a function that describes a possible redistribution of the transition probabilities of the two states; it is evaluated by the *discrepancy function* that measures the behavioural disimilarities between the two states. In [4] it is shown that the bisimilarity pseudometric for a given MC is the minimum among the discrepancy functions corresponding to all the couplings that can be defined for that MC; moreover, the bisimilarity pseudometric is itself a discrepancy function corresponding to an *optimal coupling*. This suggests that the problem of computing the pseudometric can be reduced to the problem of finding a coupling with the least discrepancy function.

Our approach aims at finding an optimal coupling using a greedy strategy that starts from an arbitrary coupling and repeatedly looks for new couplings that improve the discrepancy function. This strategy will eventually find an optimal coupling. We use it to support the design of an on-the-fly algorithm for computing the exact behavioural pseudometric that can be either applied to compute all the distances in the model or to compute only some designated distances. The advantage of using an on-the-fly approach consists in the fact that we do not need to exhaustively explore the state space nor to construct entire couplings but only those fragments that are needed in the local computation.

The efficiency of our algorithm has been evaluated on a significant set of randomly generated MCs. The results show that our algorithm performs orders of magnitude better than the corresponding iterative algorithms proposed, for instance in [9,4]. Moreover, we provide empirical evidence for the fact that our algorithm enjoys good execution running times.

One of the main practical advantages of our approach consists in the fact that it can focus on computing only the distances between states that are of particular interest. This is useful in practice, for instance when large systems are considered and visiting the entire state space is expensive. A similar issue has been considered by Comanici et. al., in [5], who have noticed that for computing the approximated pseudometric one does not need to update the current value for all the pairs at each iteration, but it is sufficient to only focus on the pairs where changes are happening rapidly. Our approach goes much beyond this idea. Firstly, we are not only looking at approximations of the bisimilarity distance, but we develop an exact algorithm; secondly, we provide a termination condition that can be checked locally, still ensuring that the local optimum corresponds to the global one. In addition, our method can be applied to decide whether two states of an MC are probabilistic bisimilar, to identify the bisimilarity classes for a given MC or to solve lumpability problems. Our approach can also be used with approximation techniques as, for instance, to provide a least over-approximation of the behavioural distance given over-estimates of some particular distances. This can be further integrated with other approximate algorithms having the advantage of the on-the-fly state space exploration.

Synopsis: The paper is organized as follows. In Section 2, we recall the basic preliminaries on Markov chains and define the bisimilarity pseudometric, for which we provide an alternative characterization in Section 3. Section 4 collects all theoretical results which are the basis for the development of the on-the-fly algorithm, presented in Section 5, for the exact computation of the pseudometric. The efficiency of our algorithm is supported by experimental results, shown in Section 6. Final remarks and conclusions are in Section 7.

2 Markov Chains and Bisimilarity Pseudometrics

In this section we give the definitions of *(discrete-time) Markov chains* (MCs) and *probabilistic bisimilarity* for MCs [12]. Then we recall the *bisimilarity pseudometric* of Desharnais et. al. [8], but rather than giving its first logical definition, we present its fixed point characterization given by van Breugel et. al. [16].

Definition 1 (Markov chain). *A* (discrete-time) Markov chain *is a tuple* $\mathcal{M} = (S, A, \pi, \ell)$ *consisting of a countable nonempty set* S *of* states*, a nonempty set* A *of* labels*, a* transition probability function $\pi \colon S \times S \to [0, 1]$ *such that, for arbitrary* $s \in S$*,* $\sum_{t \in S} \pi(s, t) = 1$*, and a* labelling function $\ell \colon S \to A$*.* $\mathcal{M}$ *is* finite *if its support set* S *is finite.*

Given a finite MC $\mathcal{M} = (S, A, \pi, \ell)$, we identify the transition probability function π with its *transition matrix* $(\pi(s, t))_{s,t \in S}$. For $s, t \in S$, we denote by $\pi(s, \cdot)$ and $\pi(\cdot, t)$, respectively, the probability distribution of exiting from s to any state and the sub-probability distribution of entering to t from any state.

The MC $\mathcal{M}$ induces an *underlying (directed) graph*, denoted by $\mathcal{G}(\mathcal{M})$, where the states act as vertices and (s, t) is an edge in $\mathcal{G}(\mathcal{M})$, if and only if, $\pi(s, t) > 0$. For a subset $Q \subseteq S$, we denote by $R_{\mathcal{M}}(Q)$ the set of states reachable from some $s \in Q$, and by $R_{\mathcal{M}}(s)$ we denote $R_{\mathcal{M}}(\{s\})$.

From a theoretical point of view, it is irrelevant whether the transition probability function of a given Markov Chain has rational values or not. However, for algorithmic purposes, in this paper we assume that for arbitrary $s, t \in S$, $\pi(s,t) \in \mathbb{Q} \cap [0,1]$. For computational reasons, in the rest of the paper we restrict our investigation to finite Markov chains.

Definition 2 (Probabilistic Bisimulation). *Let $\mathcal{M} = (S, A, \pi, \ell)$ be an MC. An equivalence relation $R \subseteq S \times S$ is a* probabilistic bisimulation *if whenever $s \mathrel{R} t$, then*

(i) $\ell(s) = \ell(t)$ and,
(ii) for each R-equivalence class E, $\sum_{u \in E} \pi(s,u) = \sum_{u \in E} \pi(t,u)$.

Two states $s, t \in S$ are bisimilar*, written $s \sim t$, if they are related by some probabilistic bisimulation.*

This definition is due to Larsen and Skou [12]. Intuitively, two states are bisimilar if they have the same label and their probability of moving by a single transition to any given equivalence class is always the same.

The notion of equivalence can be relaxed by means of a pseudometric, which tells us how far apart from each other two elements are and whenever they are at zero distance they are equivalent. The bisimilarity pseudometric of Desharnais et. al. [8] on MCs enjoys the property that two states are at zero distance if and only if they are bisimilar. This pseudometric can be defined as the least fixed point of an operator based on the Kantorovich metric for comparing probability distributions, which makes use of the notion of *matching*.

Definition 3 (Matching). *Let $\mu, \nu \colon S \to [0,1]$ be probability distributions on S. A* matching for the pair (μ, ν) *is a probability distribution $\omega \colon S \times S \to [0,1]$ on $S \times S$ satisfying*

$$\forall\, u \in S.\ \textstyle\sum_{s \in S} \omega(u,s) = \mu(u)\,, \qquad \forall\, v \in S.\ \textstyle\sum_{s \in S} \omega(s,v) = \nu(v)\,. \tag{1}$$

We call μ and ν, respectively, the left and the right marginals *of ω.*

In the following, we denote by $\mu \otimes \nu$ the set of all matchings for (μ, ν).

Remark 4. Note that, for S finite, (1) describes the constraints of a homogeneous transportation problem (TP) [7,10], where the vector $(\mu(u))_{u \in S}$ specifies the amounts to be shipped and $(\nu(v))_{v \in S}$ the amounts to be received. Thus, a matching ω for (μ, ν) induces a matrix $(\omega(u,v))_{u,v \in S}$ to be thought as a shipping schedule belonging to the transportation polytope $\mu \otimes \nu$. Hereafter, we denote by $TP(c, \nu, \mu)$ the TP with cost matrix $(c(u,v))_{u,v \in S}$ and marginals ν and μ. ■

For $\mathcal{M} = (S, A, \pi, \ell)$ an MC, and $\lambda \in (0,1]$ a *discount factor*, the operator $\Delta_\lambda^{\mathcal{M}} \colon [0,1]^{S \times S} \to [0,1]^{S \times S}$, for $d \colon S \times S \to [0,1]$ and $s, t \in S$, is defined by:

$$\Delta_\lambda^{\mathcal{M}}(d)(s,t) = \begin{cases} 1 & \text{if } \ell(s) \neq \ell(t) \\ \lambda \cdot \displaystyle\min_{\omega \in \pi(s,\cdot) \otimes \pi(t,\cdot)} \sum_{u,v \in S} d(u,v) \cdot \omega(u,v) & \text{if } \ell(s) = \ell(t) \end{cases}$$

In the above definition, $\pi(s,\cdot) \otimes \pi(t,\cdot)$ is a closed polytope so that the minimum is well defined and it corresponds to the optimal value of $TP(d, \pi(s,\cdot), \pi(t,\cdot))$.

The set $[0,1]^{S\times S}$ is endowed with the partial order $\sqsubseteq$ defined as $d \sqsubseteq d'$ iff $d(s,t) \leq d'(s,t)$ for all $s,t \in S$. This forms a complete lattice with bottom element $\mathbf{0}$ and top element $\mathbf{1}$, defined as $\mathbf{0}(s,t) = 0$ and $\mathbf{1}(s,t) = 1$, for all $s,t \in S$. For $D \subseteq [0,1]^{S\times S}$, the least upper bound $\bigsqcup D$, and greatest lower bound $\bigsqcap D$ are given by $(\bigsqcup D)(s,t) = \sup_{d\in D} d(s,t)$ and $(\bigsqcap D)(s,t) = \inf_{d\in D} d(s,t)$ for all $s,t \in S$.

In [16], for any $\mathcal{M}$ and $\lambda \in [0,1]$, $\Delta^{\mathcal{M}}_{\lambda}$ is proved to be monotonic, thus, by Tarski's fixed point theorem, it admits least and greatest fixed points.

Definition 5 (Bisimilarity pseudometric). *Let $\mathcal{M}$ be an MC and $\lambda \in (0,1]$ be a discount factor, then the λ-discounted* bisimilarity pseudometric *for $\mathcal{M}$, written $\delta^{\mathcal{M}}_{\lambda}$, is the least fixed point of $\Delta^{\mathcal{M}}_{\lambda}$.*

Hereafter, $\Delta^{\mathcal{M}}_{\lambda}$ and $\delta^{\mathcal{M}}_{\lambda}$ will be denoted simply by Δ_{λ} and δ_{λ}, respectively, when the Markov chain $\mathcal{M}$ is clear from the context.

3 Alternative Characterization of the Pseudometric

In [4], Chen et. al. proposed an alternative characterization of δ_1, relating the pseudometric to the notion of *coupling*. In this section, we recall the definition of coupling, and generalize the characterization for generic discount factors.

Definition 6 (Coupling). *Let $\mathcal{M} = (S, A, \pi, \ell)$ be a finite MC. The Markov chain $\mathcal{C} = (S\times S, A\times A, \omega, l)$ is said a* coupling for $\mathcal{M}$ *if, for all $s,t \in S$,*

1. $\omega((s,t),\cdot) \in \pi(s,\cdot) \otimes \pi(t,\cdot)$, and
2. $l(s,t) = (\ell(s), \ell(t))$.

A coupling for $\mathcal{M}$ can be seen as a probabilistic pairing of two copies of $\mathcal{M}$ running synchronously, although not necessarily independently. Couplings have been used to characterize weak ergodicity of arbitrary Markov chains [11], or to give upper bounds on convergence to stationary distributions [1,13].

Given a coupling $\mathcal{C} = (S\times S, A\times A, \omega, l)$ for $\mathcal{M} = (S, A, \pi, \ell)$ we define $\Gamma^{\mathcal{C}}_{\lambda} \colon [0,1]^{S\times S} \to [0,1]^{S\times S}$ for $d \colon S\times S \to [0,1]$ and $s,t \in S$, as follows:

$$\Gamma^{\mathcal{C}}_{\lambda}(d)(s,t) = \begin{cases} 1 & \text{if } \ell(s) \neq \ell(t) \\ \lambda \cdot \displaystyle\sum_{u,v\in S} d(u,v) \cdot \omega((s,t),(u,v)) & \text{if } \ell(s) = \ell(t) \end{cases}$$

One can easily verify that, for any $\lambda \in (0,1]$, $\Gamma^{\mathcal{C}}_{\lambda}$ is well-defined, moreover it is order preserving. By Tarski's fixed point theorem, $\Gamma^{\mathcal{C}}_{\lambda}$ admits a least fixed point, which we denote by $\gamma^{\mathcal{C}}_{\lambda}$. In Section 4.1 we will see that, for any $s,t \in S$, $\gamma^{\mathcal{C}}_{1}(s,t)$ corresponds to the probability of reaching a state (u,v) with $\ell(u) \neq \ell(v)$ starting from the state (s,t) in the underling graph of $\mathcal{C}$. For this reason we will call $\gamma^{\mathcal{C}}_{\lambda}$ the λ-*discounted discrepancy* of $\mathcal{C}$ or simply the λ-discrepancy of $\mathcal{C}$.

Lemma 7. *Let $\mathcal{M}$ be an MC, $\mathcal{C}$ be a coupling for $\mathcal{M}$, and $\lambda \in (0,1]$ be a discount factor. If $d = \Gamma^{\mathcal{C}}_{\lambda}(d)$ then $\delta_\lambda \sqsubseteq d$.*

As a consequence of Lemma 7 we obtain the following characterization for δ_λ, which generalizes [4, Theorem 8] for generic discount factors.

Theorem 8 (Minimum coupling criterion). *Let $\mathcal{M}$ be an MC and $\lambda \in (0,1]$ be a discount factor. Then, $\delta_\lambda = \min \left\{ \gamma^{\mathcal{C}}_{\lambda} \mid \mathcal{C} \text{ coupling for } \mathcal{M} \right\}$.*

Proof. For any fixed $d \in [0,1]^{S \times S}$ there exists a coupling $\mathcal{C}$ for $\mathcal{M}$ such that $\Gamma^{\mathcal{C}}_{\lambda}(d) = \Delta_\lambda(d)$. Indeed we can take as transition function for $\mathcal{C}$, the joint probability distribution ω such that, for all $s,t \in S$, $\sum_{u,v \in S} d(u,v) \cdot \omega((s,t),(u,v))$ achieves the minimum value.

Let $\mathcal{D}$ be a coupling for $\mathcal{M}$ such that $\Gamma^{\mathcal{D}}_{\lambda}(\delta_\lambda) = \Delta_\lambda(\delta_\lambda)$. By Definition 5, $\Delta_\lambda(\delta_\lambda) = \delta_\lambda$, therefore δ_λ is a fixed point for $\Gamma^{\mathcal{D}}_{\lambda}$. By Lemma 7, δ_λ is a lower bound of the set of fixed points of $\Gamma^{\mathcal{D}}_{\lambda}$, therefore $\delta_\lambda = \gamma^{\mathcal{D}}_{\lambda}$. By Lemma 7, we have also that, for any coupling $\mathcal{C}$ of $\mathcal{M}$, $\delta_\lambda \sqsubseteq \gamma^{\mathcal{C}}_{\lambda}$. Therefore, given the set $D = \left\{ \gamma^{\mathcal{C}}_{\lambda} \mid \mathcal{C} \text{ coupling for } \mathcal{M} \right\}$, it follows that $\delta_\lambda \in D$ and δ_λ is a lower bound for D. Hence, by antisymmetry of $\sqsubseteq$, $\delta_\lambda = \min D$. □

4 Exact Computation of Bisimilarity Distance

Inspired by the characterization given in Theorem 8, in this section we propose a procedure to exactly compute the bisimilarity pseudometric.

For $\lambda \in (0,1]$, the set of couplings for $\mathcal{M}$ can be endowed with the preorder $\trianglelefteq_\lambda$ defined as $\mathcal{C} \trianglelefteq_\lambda \mathcal{D}$, if and only if, $\gamma^{\mathcal{C}}_{\lambda} \sqsubseteq \gamma^{\mathcal{D}}_{\lambda}$. Theorem 8 suggests to look at all the couplings $\mathcal{C}$ for $\mathcal{M}$ in order to find an optimal one, i.e., minimal w.r.t. $\trianglelefteq_\lambda$. However, it is clear that the enumeration of all the couplings is unfeasible, therefore it is crucial to provide an efficient search strategy which prevents us to do that. Moreover we also need an efficient method for computing the λ-discrepancy.

In Subsection 4.1 the problem of computing the λ-discrepancy of a coupling $\mathcal{C}$ is reduced to the problem of computing reachability probabilities in $\mathcal{C}$. Then, Subsection 4.2 illustrates a greedy strategy that explores the set of couplings until an optimal one is eventually reached.

4.1 Computing the λ-Discrepancy

In this section, we first recall the problem of computing the reachability probability for general MCs [1], then we instantiate it to compute the λ-discrepancy.

Let $\mathcal{M} = (S, A, \pi, \ell)$ be an MC, and x_s denote the probability of reaching $G \subseteq S$ from $s \in S$. The goal is to compute x_s for all $s \in S$. The following holds

$$x_s = 1 \quad \text{if } s \in G\,, \qquad\qquad x_s = \textstyle\sum_{t \in S} x_t \cdot \pi(s,t) \quad \text{if } s \in S \setminus G\,, \tag{2}$$

that is, either G is already reached, or it can be reached by way of another state. Equation (2) defines a linear equation system of the form $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, where $S_? = S \setminus G$, $\boldsymbol{x} = (x_s)_{s \in S_?}$, $\boldsymbol{A} = (\pi(s,t))_{s,t \in S_?}$, and $\boldsymbol{b} = (\sum_{t \in G} \pi(s,t))_{s \in S_?}$.

This linear equation system always admits a solution in $[0,1]^S$, however, it may not be unique. Since we are interested in the least solution, we address this problem by fixing each free variable to zero, so that we obtain a reduced system with a unique solution. This can be easily done by inspecting the graph $\mathcal{G}(\mathcal{M})$: all variables with zero probability of reaching G are detected by checking that they cannot be reached from any state in G in the reverse graph of $\mathcal{G}(\mathcal{M})$.

Regarding the λ-discrepancy for a coupling $\mathcal{C}$, if $\lambda = 1$, one can directly instantiate the aforementioned method with $G = \{(s,t) \in S \times S \mid \ell(s) \neq \ell(t)\}$ and $S_? = (S \times S) \setminus G$. As for generic $\lambda \in (0,1]$, the discrepancy $\gamma_\lambda^{\mathcal{C}}$ can be formulated as the least solution in $[0,1]^{S \times S}$ of the linear equation system

$$\boldsymbol{x} = \lambda \boldsymbol{A} \boldsymbol{x} + \lambda \boldsymbol{b}\,. \tag{3}$$

Remark 9. If one is interested in computing the λ-discrepancy for a particular pair of states (s,t), the method above can be applied on the least independent subsystem of Equation (3) containing the variable $x_{(s,t)}$. Moreover, assuming that for some pairs the λ-discrepancy is already known, the goal set can be extended with all those pairs with λ-discrepancy greater than zero. ■

4.2 Greedy Search Strategy for Computing an Optimal Coupling

In this section, we give a greedy strategy for moving toward an optimal coupling starting from a given one. Then we provide sufficient and necessary conditions for a coupling, ensuring that its associated λ-discrepancy coincides with δ_λ.

Hereafter, we fix a coupling $\mathcal{C} = (S \times S, A \times A, \omega, l)$ for $\mathcal{M} = (S, A, \pi, \ell)$. Let $s, t \in S$ and μ be a matching for $(\pi(s,\cdot), \pi(t,\cdot))$. We denote by $\mathcal{C}[(s,t)/\mu]$ the coupling for $\mathcal{M}$ with the same labeling function of $\mathcal{C}$ and transition function ω' defined by $\omega'((u,v),\cdot) = \omega((u,v),\cdot)$, for all $(u,v) \neq (s,t)$, and $\omega'((s,t),\cdot) = \mu$.

Lemma 10. *Let $\mathcal{C}$ be a coupling for $\mathcal{M}$, $s,t \in S$, $\omega' \in \pi(s,\cdot) \otimes \pi(t,\cdot)$, and $\mathcal{D} = \mathcal{C}[(s,t)/\omega']$. If $\Gamma_\lambda^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}})(s,t) < \gamma_\lambda^{\mathcal{C}}(s,t)$, then $\gamma_\lambda^{\mathcal{D}} \sqsubset \gamma_\lambda^{\mathcal{C}}$.*

Lemma 10 states that $\mathcal{C}$ can be improved w.r.t. $\trianglelefteq_\lambda$ by updating its transition function at (s,t), if $\ell(s) = \ell(t)$ and there exists $\omega' \in \pi(s,\cdot) \otimes \pi(t,\cdot)$ such that

$$\textstyle\sum_{u,v \in S} \gamma_\lambda^{\mathcal{C}}(u,v) \cdot \omega'(u,v) < \sum_{u,v \in S} \gamma_\lambda^{\mathcal{C}}(u,v) \cdot \omega((s,t),(u,v)).$$

Notice that, an optimal schedule ω' for $TP(\gamma_\lambda^{\mathcal{C}}, \pi(s,\cdot), \pi(t,\cdot))$ enjoys the above condition, so that, the update $\mathcal{C}[(s,t)/\omega']$ improves $\mathcal{C}$. This gives us a strategy for moving toward δ_λ by successive improvements on the couplings.

Now we proceed giving a sufficient and necessary condition for termination.

Lemma 11. *For any $\lambda \in (0,1]$, if $\gamma_\lambda^{\mathcal{C}} \neq \delta_\lambda$, then there exist $s,t \in S$ and a coupling $\mathcal{D} = \mathcal{C}[(s,t)/\omega']$ for $\mathcal{M}$ such that $\Gamma_\lambda^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}})(s,t) < \gamma_\lambda^{\mathcal{C}}(s,t)$.*

The above result ensures that, unless $\mathcal{C}$ is optimal w.r.t $\trianglelefteq_\lambda$, the hypothesis of Lemma 10 are satisfied, so that, we can further improve $\mathcal{C}$ as aforesaid.

The next statement proves that this search strategy is correct.

Theorem 12. $\delta_\lambda = \gamma_\lambda^{\mathcal{C}}$ *iff there is no coupling* $\mathcal{D}$ *for* $\mathcal{M}$ *such that* $\Gamma_\lambda^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}}) \sqsubset \gamma_\lambda^{\mathcal{C}}$.

Proof. We prove: $\delta_\lambda \neq \gamma_\lambda^{\mathcal{C}}$ iff there exists $\mathcal{D}$ such that $\Gamma_\lambda^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}}) \sqsubset \gamma_\lambda^{\mathcal{C}}$. ($\Rightarrow$) Assume $\delta_\lambda \neq \gamma_\lambda^{\mathcal{C}}$. By Lemma 11, there are $s, t \in S$ and $\omega' \in \pi(s,\cdot) \otimes \pi(t,\cdot)$ such that $\lambda \cdot \sum_{u,v \in S} \gamma_\lambda^{\mathcal{C}}(u,v) \cdot \omega'(u,v) < \gamma_\lambda^{\mathcal{C}}(s,t)$. As in the proof of Lemma 10, we have that $\mathcal{D} = \mathcal{C}[(s,t)/\omega']$ satisfies $\Gamma^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}}) \sqsubset \gamma_\lambda^{\mathcal{C}}$. ($\Leftarrow$) Let $\mathcal{D}$ be such that $\Gamma_\lambda^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}}) \sqsubset \gamma_\lambda^{\mathcal{C}}$. By Tarski's fixed point theorem $\gamma_\lambda^{\mathcal{D}} \sqsubset \gamma_\lambda^{\mathcal{C}}$. By Theorem 8, $\delta_\lambda \sqsubseteq \gamma_\lambda^{\mathcal{D}} \sqsubset \gamma_\lambda^{\mathcal{C}}$. □

Remark 13. Note that, in general there could be an infinite number of couplings for a given MC. However, for each fixed $d \in [0,1]^{S \times S}$, the linear function mapping $\omega((s,t),\cdot)$ to $\lambda \sum_{u,v \in S} d(u,v) \cdot \omega((s,t),(u,v))$ achieves its minimum at some vertex in the transportation polytope $\pi(s,\cdot) \otimes \pi(t,\cdot)$. Since the number of such vertices are finite, using the optimal TP schedule for the update, ensures that the search strategy is always terminating. ■

5 The On-the-Fly Algorithm

In this section we provide an on-the-fly algorithm for exact computation of the bisimilarity distance δ_λ for generic discount factors making full use of the greedy strategy presented in Section 4.2.

Let $Q \subseteq S \times S$. Assume we want to compute $\delta_\lambda(s,t)$, for all $(s,t) \in Q$. The method proposed in Section 4.2 has the following key features:

1. the improvement of each coupling $\mathcal{C}$ is obtained by a *local* update of its transition function at some state (u,v) in $\mathcal{C}$;
2. the strategy does not depend on the choice of the state (u,v);
3. whenever a coupling $\mathcal{C}$ is considered, the over-approximation $\gamma_\lambda^{\mathcal{C}}$ of the distance can be computed by solving a system of linear equations.

Among them, only the last one requires a visit of the coupling. However, as noticed in Remark 9, the value $\gamma_\lambda^{\mathcal{C}}(s,t)$ can be computed without considering the entire linear system of Equation (3), but only its smallest independent subsystem containing the variable $x_{(s,t)}$, which is obtained by restricting on the variables $x_{(u,v)}$ such that $(u,v) \in R_C((s,t))$. This subsystem can be further reduced, by Gaussian elimination, when some values for δ_λ are known. The last observation suggests that, in order to compute $\gamma_\lambda^{\mathcal{C}}(s,t)$, we do not need to store the entire coupling, but it can be constructed on-the-fly.

The exact computation of the bisimilarity pseudometric is implemented by Algorithm 1. It takes as input a finite MC $\mathcal{M} = (S, A, \pi, \ell)$, a discount factor λ, and a query set Q. We assume the following global variables to store: $\mathcal{C}$, the current partial coupling; d, the λ-discrepancy associated with $\mathcal{C}$; $ToCompute$, the pairs of states for which the distance has to be computed; $Exact$, the pairs of states (s,t) such that $d(s,t) = \delta_\lambda(s,t)$; $Visited$, the states of $\mathcal{C}$ considered so far. At the beginning (line 1) both the coupling $\mathcal{C}$ and the discrepancy d are empty, there are no visited states, and no exact computed distances. While there are still pairs left to be computed (line 2), we pick one (line 3), say (s,t). According to the definition of δ_λ, if $\ell(s) \neq \ell(t)$ then $\delta_\lambda(s,t) = 1$; if $s = t$ then $\delta_\lambda(s,t) = 0$, so

Algorithm 1 On-the-Fly Bisimilarity Pseudometric

```
Input: MC $\mathcal{M} = (S, A, \pi, \ell)$; discount factor $\lambda \in (0, 1]$; query $Q \subseteq S \times S$.
 1. $\mathcal{C} \leftarrow$ empty; $d \leftarrow$ empty; $Visited \leftarrow \emptyset$; $Exact \leftarrow \emptyset$; $ToCompute \leftarrow Q$;   // Init.
 2. while $ToCompute \neq \emptyset$ do
 3.   pick $(s, t) \in ToCompute$
 4.   if $\ell(s) \neq \ell(t)$ then
 5.     $d(s, t) \leftarrow 1$; $Exact \leftarrow Exact \cup \{(s, t)\}$; $Visited \leftarrow Visited \cup \{(s, t)\}$
 6.   else if $s = t$ then
 7.     $d(s, t) \leftarrow 0$; $Exact \leftarrow Exact \cup \{(s, t)\}$; $Visited \leftarrow Visited \cup \{(s, t)\}$
 8.   else   // if $(s, t)$ is nontrivial
 9.     if $(s, t) \notin Visited$ then pick $\omega \in \pi(s, \cdot) \otimes \pi(t, \cdot)$; $SetPair(\mathcal{M}, (s, t), \omega)$
10.     $Discrepancy(\lambda, (s, t))$   // update $d$ as the $\lambda$-discrepancy for $\mathcal{C}$
11.     while $\exists (u, v) \in R_{\mathcal{C}}((s, t))$. $\mathcal{C}[(u, v)]$ not opt. for $TP(d, \pi(u, \cdot), \pi(v, \cdot))$ do
12.       $\omega \leftarrow$ optimal schedule for $TP(d, \pi(u, \cdot), \pi(v, \cdot))$
13.       $SetPair(\mathcal{M}, (u, v), \omega)$   // improve the current coupling
14.       $Discrepancy(\lambda, (s, t))$   // update $d$ as the $\lambda$-discrepancy for $\mathcal{C}$
15.     end while
16.     $Exact \leftarrow Exact \cup R_{\mathcal{C}}((s, t))$   // add new exact distances
17.     remove from $\mathcal{C}$ all edges exiting from nodes in $Exact$
18.   end if
19.   $ToCompute \leftarrow ToCompute \setminus Exact$   // remove exactly computed pairs
20. end while
21. return $d\restriction_Q$   // return the distance for all pairs in $Q$
```

that, $d(s, t)$ is set accordingly, and (s, t) is added to *Exact* (lines 4–7). Otherwise, if (s, t) was not previously visited, a matching $\omega \in \pi(s, \cdot) \otimes \pi(t, \cdot)$ is guessed, and the routine *SetPair* updates the coupling $\mathcal{C}$ at (s, t) with ω (line 9), then the routine *Discrepancy* updates d with the λ-discrepancy associated with $\mathcal{C}$ (line 10). According to the greedy strategy, $\mathcal{C}$ is successively improved and d is consequently updated, until no further improvements are possible (lines 11–15). Each improvement is demanded by the existence of a better schedule for $TP(d, \pi(u, \cdot), \pi(u, \cdot))$ (line 11). Note that, each improvement actually affects the current value of $d(s, t)$. This is done by restricting our attention only to the pairs that are reachable from (s, t) in $\mathcal{G}(\mathcal{C})$. It is worth to note that $\mathcal{C}$ is constantly updated, hence $R_{\mathcal{C}}((s, t))$ may differ from one iteration to another. When line 16 is reached, for each $(u, v) \in R_{\mathcal{C}}((s, t))$, we are guaranteed that $d(u, v) = \delta_\lambda(s, t)$, therefore $R_{\mathcal{C}}((s, t))$ is added to *Exact*, and these values can be used in successive computations, so the edges exiting from these states are removed from $\mathcal{G}(\mathcal{C})$. In line 19, the exact pairs computed so far are removed from *ToCompute*. Finally, if no more pairs need be considered, the exact distance on Q is returned (line 21).

Algorithm 1 calls the subroutines *SetPair* and *Discrepancy*, respectively, to construct/update the coupling $\mathcal{C}$, and to update the current over-approximation d during the computation. Now we explain how they work.

SetPair (Algorithm 2) takes as input an MC $\mathcal{M} = (S, A, \pi, \ell)$, a pair of states (s, t), and a matching $\omega \in \pi(s, \cdot) \otimes \pi(t, \cdot)$. In lines 1–2 the transition function of the coupling $\mathcal{C}$ is set to ω at (s, t), then (s, t) is added to *Visited*. The on-

Algorithm 2 *SetPair*$(\mathcal{M}, (s,t), \omega)$

```
Input: MC $\mathcal{M} = (S, A, \pi, \ell)$; $s,t \in S$; $\omega \in \pi(s,\cdot) \otimes \pi(t,\cdot)$
 1. $\mathcal{C}[(s,t)] \leftarrow \omega$   // update the coupling at $(s,t)$ with $\omega$
 2. $Visited \leftarrow Visited \cup \{(s,t)\}$   // set $(s,t)$ as visited
 3. for all $(u,v) \in \{(u',v') \mid \omega(u',v') > 0\} \setminus Visited$ do   // for all demanded pairs
 4.    $Visited \leftarrow Visited \cup \{(u,v)\}$
 5.    if $u = v$ then $d(u,v) \leftarrow 0$; $Exact \leftarrow Exact \cup \{(u,v)\}$;
 6.    if $\ell(u) \neq \ell(v)$ then $d(u,v) \leftarrow 1$; $Exact \leftarrow Exact \cup \{(u,v)\}$;
 7.    // propagate the construction
 8.    if $(u,v) \notin Exact$ then
 9.       pick $\omega' \in \pi(u,\cdot) \otimes \pi(v,\cdot)$   // guess a matching
10.       $SetPair(\mathcal{M}, (u,v), \omega')$
11.    end if
12. end for
```

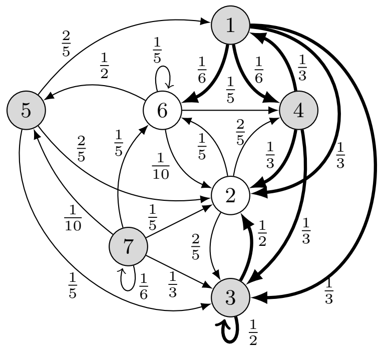
Algorithm 3 *Discrepancy*$(\lambda, (s,t))$

```
Input: discount factor $\lambda \in (0,1]$; $s,t \in S$
 1. $Nonzero \leftarrow \emptyset$   // detect non-zero variables
 2. for all $(u,v) \in R_{\mathcal{C}}((s,t)) \cap Exact$ such that $d(u,v) > 0$ do
 3.    $Nonzero \leftarrow Nonzero \cup \big\{(u',v') \mid (u,v) \rightsquigarrow (u',v') \text{ in } \mathcal{G}^{-1}(\mathcal{C})\big\}$
 4. end for
 5. for all $(u,v) \in R_{\mathcal{C}}((s,t)) \setminus Nonzero$ do   // set distance to zero
 6.    $d(u,v) \leftarrow 0$; $Exact \leftarrow Exact \cup \{(u,v)\}$
 7. end for
 8. // construct the reduced linear system over nonzero variables
 9. $\boldsymbol{A} \leftarrow (\mathcal{C}[(u,v)](u',v'))_{(u,v),(u',v') \in Nonzero}$
10. $\boldsymbol{b} \leftarrow \big(\sum_{(u',v') \in Exact} d(u',v') \cdot \mathcal{C}[(u,v)](u',v')\big)_{(u,v) \in Nonzero}$
11. $\tilde{\boldsymbol{x}} \leftarrow$ solve $\boldsymbol{x} = \lambda \boldsymbol{A}\boldsymbol{x} + \lambda \boldsymbol{b}'$   // solve the reduced linear system
12. for all $(u,v) \in Nonzero$ do   // update distances
13.    $d(u,v) \leftarrow \tilde{x}_{(u,v)}$
14. end for
```

the-fly construction of the coupling is recursively propagated to the successors of (s,t) in $\mathcal{G}(\mathcal{C})$. During this construction, if some states with trivial distances are encountered, d and *Exact* are updated accordingly (lines 5–6).

Discrepancy (Algorithm 3) takes as input a discount factor λ and a pair of states (s,t). It constructs the smallest (reduced) independent subsystem of Equation 3 having the variable $x_{(s,t)}$ (lines 9–10). As noticed in Remark 9, the least solution is computed by fixing d to zero for all the pairs which cannot be reached from any pair in *Exact* and such that its distance is greater than zero (lines 5–7). Then, the discrepancy is computed and d is consequently updated.

Next, we present a simple example of Algorithm 1, showing the main features of our method: (1) the on-the-fly construction of the (partial) coupling, and (2) the restriction only to those variables which are demanded for the solution of the system of linear equations.



$\mathcal{C}_0$

(1,4)	**1**	**2**	**3**	
2	1/3			1/3
3		1/3		1/3
4			1/6	1/6
6			1/6	1/6
	1/3	1/3	1/3	

(3,4)	**1**	**2**	**3**	
2	1/3	1/6		1/2
3		1/6	1/3	1/2
	1/3	1/3	1/3	

$\mathcal{C}_1$

(1,4)	**1**	**2**	**3**	
2		1/3		1/3
3			1/3	1/3
4	1/6			1/6
6	1/6			1/6
	1/3	1/3	1/3	

Fig. 1. Execution trace for the computation of $\delta_1(1,4)$ (details in Example 14).

Example 14 (On-the-fly computation). Consider the undiscounted distance between states 1 and 4 for the $\{\text{white}, \text{gray}\}$-labeled MC depicted in Figure 1.

Algorithm 1 guesses an initial coupling $\mathcal{C}_0$ with transition distribution ω_0. This is done considering only the pairs of states which are needed: starting from $(1,4)$, the distribution $\omega_0((1,4),\cdot)$ is guessed as in Figure 1, which demands for the exploration of $(3,4)$ and a guess $\omega_0((3,4),\cdot)$. Since no other pairs are demanded, the construction of $\mathcal{C}_0$ terminates. This gives the equation system:

$$\begin{cases} x_{1,4} = \frac{1}{3} \cdot \overbrace{x_{1,2}}^{=1} + \frac{1}{3} \cdot \overbrace{x_{2,3}}^{=1} + \frac{1}{6} \cdot x_{3,4} + \frac{1}{6} \cdot \overbrace{x_{3,6}}^{=1} = \frac{1}{6} \cdot x_{3,4} + \frac{5}{6} \\ x_{3,4} = \frac{1}{3} \cdot \overbrace{x_{1,2}}^{=1} + \frac{1}{6} \cdot \overbrace{x_{2,2}}^{=0} + \frac{1}{6} \cdot \overbrace{x_{2,3}}^{=1} + \frac{1}{3} \cdot \overbrace{x_{3,3}}^{=0} = \frac{1}{2} . \end{cases}$$

Note that the only variables appearing in the above equation system correspond to the pairs which have been considered so far. The least solution for it is given by $d^{\mathcal{C}_0}(1,4) = \frac{11}{12}$ and $d^{\mathcal{C}_0}(3,4) = \frac{1}{2}$.

Now, these solutions are taken as the costs of a TP, from which we get an optimal transportation schedule $\omega_1((1,4),\cdot)$ improving $\omega_0((1,4),\cdot)$. The distribution ω_1 is used to update $\mathcal{C}_0$ to $\mathcal{C}_1 = \mathcal{C}_0[(1,4)/\omega_1]$ (depicted in Figure 1), obtaining the following new equation system:

$$x_{1,4} = \frac{1}{3} \cdot \overbrace{x_{2,2}}^{=0} + \frac{1}{3} \cdot \overbrace{x_{3,3}}^{=0} + \frac{1}{6} \cdot x_{1,4} + \frac{1}{6} \cdot \overbrace{x_{1,6}}^{=1} = \frac{1}{6} \cdot x_{1,4} + \frac{1}{6} ,$$

which has $d^{\mathcal{C}_1}(1,4) = \frac{1}{5}$ as least solution. Note that, $(3,4)$ is no more demanded, thus we do not need to update it. Running again the TP on the improved over-approximation $d^{\mathcal{C}_1}$, we discover that the coupling $\mathcal{C}_1$ cannot be further improved, hence we stop the computation, returning $\delta_1(1,4) = d^{\mathcal{C}_1}(1,4) = \frac{1}{5}$.

It is worth noticing that Algorithm 1 does not explore the entire MC, not even all the reachable states from 1 and 4. The only edges in the MC which have been considered during the computation are highlighted in Figure 1. ■

# States	On-the-Fly (exact)		Iterating (approximated)			Approximation
	Time (s)	# TPs	Time (s)	# Iterations	# TPs	Error
5	0.019675	1.19167	0.0389417	1.73333	26.7333	0.139107
6	0.05954	3.04667	0.09272	1.82667	38.1333	0.145729
7	0.13805	6.01111	0.204789	2.19444	61.7278	0.122683
8	0.255067	8.5619	0.364019	2.30476	83.0286	0.11708
9	0.499983	12.0417	0.673275	2.57917	114.729	0.111104
10	1.00313	18.7333	1.27294	3.11111	174.363	0.0946047
11	2.15989	25.9733	2.66169	3.55667	239.557	0.0959714
12	4.64225	34.797	5.52232	4.04242	318.606	0.0865612
13	6.73513	39.9582	8.06186	4.63344	421.675	0.0977743
14	6.33637	38.0048	7.18807	4.91429	593.981	0.118971
17	11.2615	47.0143	12.8048	5.88571	908.61	0.13213
19	26.6355	61.1714	29.6542	6.9619	1328.6	0.14013
20	34.379	66.4571	38.2058	7.5381	1597.92	0.142834

Table 1. Comparison between the on-the-fly algorithm and the iterative method.

Remark 15. Notably, Algorithm 1 can also be used for computing over-approximated distances. Indeed, assuming over-estimates for some particular distances are already known, they can be taken as inputs and used in our algorithm simply storing them in the variable d and treated as "exact" values. In this way our method will return the least over-approximation of the distance agreeing with the given over-estimates. This modification of the algorithm can be used to further decrease the exploration of the MC. Moreover, it can be employed in combination with other existing approximated algorithms, having the advantage of an on-the-fly state space exploration. ■

6 Experimental Results

In this section, we evaluate the performances of the on-the-fly algorithm on a collection of randomly generated MCs[3].

First, we compare the execution times of the on-the-fly algorithm with those of the iterative method proposed in [4] in the discounted case. Since the iterative method only allows for the computation of the distance for all state pairs at once, the comparison is (in fairness) made with respect to runs of our on-the-fly algorithm with input query being the set of all state pairs. For each input instance, the comparison involves the following steps:

1. we run the on-the-fly algorithm, storing both execution time and the number of solved transportation problems,

[3] The tests have been made using a prototype implementation coded in Mathematica® (available at `http://people.cs.aau.dk/~mardare/projects/tools/mc_dist.zip`) running on an Intel Core-i7 3.4 GHz processor with 12GB of RAM.

# States	out-degree = 3		$2 \leq$ out-degree $\leq$ # States*	
	Time (s)	# TPs	Time (s)	# TPs
5	0.00594318	0.272727	0.011654	0.657012
6	0.0115532	0.548936	0.0304482	1.66696
7	0.0168408	0.980892	0.0884878	3.67706
8	0.0247971	1.34606	0.164227	5.30112
9	0.0259426	1.29074	0.394543	8.16919
10	0.0583405	2.03887	1.1124	13.0961
11	0.0766988	1.82706	2.22016	18.7228
12	0.0428891	1.62038	4.94045	26.0965
13	0.06082	1.88134	10.3606	35.1738
14	0.0894778	2.79441	20.1233	46.0775
20	0.35631	6.36833	1.5266	13.1367
30	4.66113	17.3167	74.8146	76.2642
50	27.2147	30.8217	2234.54	225.394

Table 2. Average performances of the on-the-fly algorithm on single-pair queries. In the first to columns the out-degree is 3; in the last two columns, the out-degree varies from 2 to # States. (*) For 20, 30 and 50 states, out-degree is 4;

2. then, on the same instance, we execute the iterative method until the running time exceeds that of step 1. We report the execution time, the number of iterations, and the number of solved transportation problems.
3. Finally, we calculate the approximation error between the exact solution δ_λ computed by our method at step 1 and the approximate result d obtained in step 2 by the iterative method, as $\max_{s,t \in S} \delta_\lambda(s,t) - d(s,t)$.

This has been made on a collection of MCs varying from 5 to 20 states. For each $n = 5, \dots, 20$, we have considered 80 randomly generated MCs per out-degree, varying from 2 to n. Table 1 reports the average results of the comparison.

As can be seen, our use of a greedy strategy in the construction of the couplings leads to a significant improvement in the performances. We are able to compute the exact solution before the iterative method can under-approximate it with an error of ≈ 0.1, which is a considerable error for a value in $[0, 1]$.

So far, we only examined the case when the on-the-fly algorithm is run on all state pairs at once. Now, we show how the performance of our method is improved even further when the distance is computed only for single pairs of states. Table 2 shows the average execution times and number of solved transportation problems for (nontrivial) single-pair queries for randomly generated of MCs with number of states varying from 5 to 50. In the first two columns we consider MCs with out-degree equal to 3, while the last two columns show the average values for out-degrees varying from 2 to the number of states of the MCs. The results show that, when the out-degree of the MCs is low, our algorithm performs orders of magnitude better than in the general case. This is illustrated in Figure 2, where the distributions of the execution times for out-degree 6 and 8 are juxtaposed, in the case of MCs with 14 states. Each bar in the histogram

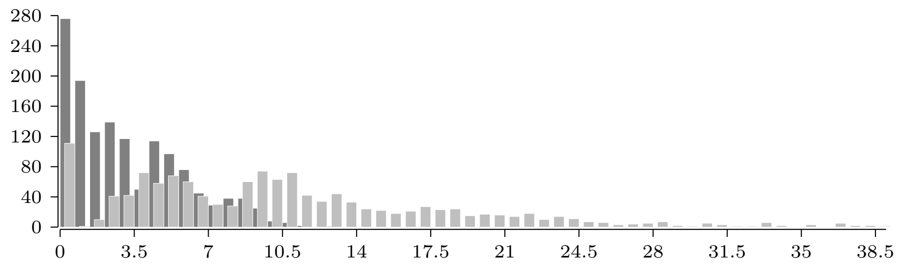


Fig. 2. Distribution of the execution times (in seconds) for 1332 tests on randomly generated MCs with 14 states, out-degree 6 (darkest) and 8 (lightest).

represents the number of tests that terminate within the time interval indicated in the x-axis.

Notably, our method may perform better on large queries than on single-pairs queries. This is due to the fact that, although the returned value does not depend on the order the queried pairs are considered, a different order may speed up the performances. When the algorithm is run on more than a single pair, the way they are picked may increase the performances (e.g., compare the execution times in Tables 1 and 2 for MCs with 14 states).

7 Conclusions and Future Work

In this paper we have proposed an on-the-fly algorithm for computing exactly the bisimilarity distance between Markov chains, introduced by Desharnais et al. in [8]. Our algorithm represents an important improvement of the state of the art in this field where, before our contribution, the known tools were only concerned with computing approximations of the bisimilarity distances and they were, in general, based on iterative techniques. We demonstrate that, using on-the-fly techniques, we cannot only calculate exactly the bisimilarity distance, but the computation time is improved with orders of magnitude with respect to the corresponding iterative approaches. Moreover, our technique allows for the computation on a set of target distances that might be done by only investigating a significantly reduced set of states, and for further improvement of speed.

Our algorithm can be practically used to address a large spectrum of problems. For instance, it can be seen as a method to decide whether two states of a given MC are probabilistic bisimilar, to identify bisimilarity classes, or to solve lumpability problems. It is sufficiently robust to be used with approximation techniques as, for instance, to provide a least over-approximation of the behavioural distance given over-estimates of some particular distances. It can be integrated with other approximate algorithms, having the advantage of the efficient on-the-fly state space exploration.

Having a practically efficient tool to compute bisimilarity distances opens the perspective of new applications already announced in previous research papers.

One of these is the state space reduction problem for MCs. Our technique can be used in this context as an indicator for the sets of neighbour states that can be collapsed due to their similarity; it also provides a tool to estimate the difference between the initial MC and the reduced one, hence a tool for the approximation theory of Markov chains.

References

1. C. Baier and J. P. Katoen. *Principles of Model Checking.* MIT Press, 2008. 3, 4.1
2. X. Cai and Y. Gu. Measuring Anonymity. In *ISPEC '09*, pages 183–194, Berlin, Heidelberg, 2009. Springer-Verlag. 1
3. K. Chatterjee, L. de Alfaro, R. Majumdar, and V. Raman. Algorithms for Game Metrics. *Logical Methods in Computer Science*, 6(3), 2010. 1
4. D. Chen, F. van Breugel, and J. Worrell. On the Complexity of Computing Probabilistic Bisimilarity. In L. Birkedal, editor, *FoSSaCS*, volume 7213 of *Lecture Notes in Computer Science*, pages 437–451. Springer, 2012. 1, 3, 3, 6, A
5. G. Comanici, P. Panangaden, and D. Precup. On-the-Fly Algorithms for Bisimulation Metrics. *International Conference on Quantitative Evaluation of Systems*, 0:94–103, 2012. 1
6. G. Comanici and D. Precup. Basis function discovery using spectral clustering and bisimulation metrics. In *AAMAS '11*, volume 3, pages 1079–1080, Richland, SC, 2011. International Foundation for Autonomous Agents and Multiagent Systems. 1
7. G. B. Dantzig. Application of the Simplex method to a transportation problem. In T. Koopmans, editor, *Activity analysis of production and allocation*, pages 359–373. J. Wiley, New York, 1951. 4, B
8. J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Metrics for labelled Markov processes. *Theoretical Compututer Science*, 318(3):323–354, 2004. 1, 2, 2, 7
9. N. Ferns, P. Panangaden, and D. Precup. Metrics for finite Markov Decision Processes. In *Proceedings of the 20th conference on Uncertainty in Artificial Intelligence*, UAI, pages 162–169. AUAI Press, 2004. 1
10. L. R. Ford and D. R. Fulkerson. Solving the Transportation Problem. *Management Science*, 3(1):24–32, 1956. 4, B
11. D. Griffeath. A maximal coupling for markov chains. *Probability Theory and Related Fields*, 31:95–106, 1975. 3
12. K. G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991. 1, 2, 2
13. M. Mitzenmacher and E. Upfal. *Probability and Computing - randomized algorithms and probabilistic analysis.* Cambridge University Press, 2005. 3
14. A. Schrijver. *Theory of linear and integer programming.* John Wiley & Sons, Inc., New York, NY, USA, 1986. 1
15. D. Thorsley and E. Klavins. Approximating stochastic biochemical processes with Wasserstein pseudometrics. *IET Systems Biology*, 4(3):193–211, 2010. 1
16. F. van Breugel, B. Sharma, and J. Worrell. Approximating a Behavioural Pseudometric without Discount for Probabilistic Systems. *Logical Methods in Computer Science*, 4(2):1–23, 2008. 1, 2, 2
17. F. van Breugel and J. Worrell. Approximating and computing behavioural distances in probabilistic transition systems. *Theoretical Computer Science*, 360(1-3):373–385, 2006. 1

A Technical proofs

In this appendix we provide the proofs for all the technical lemmas.

Proof (of Lemma 7). Assume $\mathcal{M} = (S, A, \pi, \ell)$ and $\mathcal{C} = (S \times S, A \times A, \omega, l)$. In order to prove $\delta_\lambda \sqsubseteq d$, it suffices to show that $\Delta_\lambda(d) \sqsubseteq d$. Indeed, by Tarski's fixed point theorem, δ_λ is a lower bound of $\{d \mid \Delta_\lambda(d) \sqsubseteq d\}$.

Let $s, t \in S$. If $\ell(s) \neq \ell(t)$, then $\Delta_\lambda(d)(s,t) = 1 = \Gamma^{\mathcal{C}}_\lambda(d)(s,t) = d(s,t)$. If $\ell(s) = \ell(t)$, $\Delta_\lambda(d)(s,t) = \lambda \cdot \min_{\omega' \in \pi(s,\cdot)\otimes\pi(t,\cdot)} \sum_{u,v \in S} d(u,v) \cdot \omega'(u,v)$, and $\Gamma^{\mathcal{C}}_\lambda(d)(s,t) = \lambda \cdot \sum_{u,v\in S} d(u,v) \cdot \omega((s,t),(u,v))$. Since $\omega((s,t),\cdot) \in \pi(s,\cdot)\otimes\pi(t,\cdot)$ (Definition 6), we have that $\Delta_\lambda(d)(s,t) \leq \Gamma^{\mathcal{C}}_\lambda(d)(s,t) = d(s,t)$. □

Proof (of Lemma 10). It suffices to show that $\Gamma^{\mathcal{D}}(\gamma^{\mathcal{C}}_\lambda) \sqsubset \gamma^{\mathcal{C}}_\lambda$, i.e., $\gamma^{\mathcal{C}}_\lambda$ is a strict post-fixed point of $\Gamma^{\mathcal{D}}_\lambda$. Then, the thesis follows by Tarski's fixed point theorem.

Assume $\bar{\omega}$ be the transition function of $\mathcal{D}$ and let $u, v \in S$. If $\ell(u) \neq \ell(v)$, then $\Gamma^{\mathcal{D}}_\lambda(\gamma^{\mathcal{C}}_\lambda)(u,v) = 1 = \Gamma^{\mathcal{C}}_\lambda(\gamma^{\mathcal{C}}_\lambda)(u,v) = \gamma^{\mathcal{C}}_\lambda(u,v)$. Notice that, this also means that $\ell(s) = \ell(t)$, since $\Gamma^{\mathcal{D}}_\lambda(\gamma^{\mathcal{C}}_\lambda)(s,t) < \gamma^{\mathcal{C}}_\lambda(s,t)$, by hypothesis. If $\ell(u) = \ell(v)$ and $(u,v) \neq (s,t)$, by definition of $\mathcal{D}$, we have that $\bar{\omega}((u,v),\cdot) = \omega((u,v),\cdot)$, hence $\Gamma^{\mathcal{C}}_\lambda(\gamma^{\mathcal{C}}_\lambda)(u,v) = \Gamma^{\mathcal{D}}_\lambda(\gamma^{\mathcal{C}}_\lambda)(u,v)$. This proves $\Gamma^{\mathcal{D}}(\gamma^{\mathcal{C}}_\lambda) \sqsubset \gamma^{\mathcal{C}}_\lambda$. □

Lemma 16. *Let $s, t \in S$, and $\gamma^{\mathcal{C}}_1 = \Delta_1(\gamma^{\mathcal{C}}_\lambda)$. $\gamma^{\mathcal{C}}_1(s,t) = 1$ iff $\delta_1(s,t) = 1$.*

Proof. ($\Leftarrow$) Follows by Theorem 8. ($\Rightarrow$) Assume ω be the transition function of $\mathcal{C}$. If $\ell(s) \neq \ell(t)$ the thesis follows trivially. Assume $\ell(s) = \ell(t)$.

$$\begin{aligned} 1 = \gamma^{\mathcal{C}}_1(s,t) &= \Gamma^{\mathcal{C}}_1(\gamma^{\mathcal{C}}_1)(s,t) \\ &= \textstyle\sum_{u,v\in S} \gamma^{\mathcal{C}}_1(u,v) \cdot \omega((s,t),(u,v)) \\ &\leq \textstyle\sum_{u,v\in S} \omega((s,t),(u,v)) = 1 \end{aligned}$$

Thus whenever $\omega((s,t),(u,v)) > 0$, $\gamma^{\mathcal{C}}_1(u,v) = 1$. By hypothesis, $\gamma^{\mathcal{C}}_1 = \Delta_1(\gamma^{\mathcal{C}}_\lambda)$, therefore $1 = \gamma^{\mathcal{C}}_1(s,t) = \min_{\omega' \in \pi(s,\cdot)\otimes\pi(t,\cdot)} \sum_{u,v\in S} \gamma^{\mathcal{C}}_1(u,v) \cdot \omega'(u,v)$. Hence there is no coupling that can improve the summation. Therefore, by Theorem 8, $\delta_1(s,t) = 1$. □

Lemma 17. *For any $\lambda \in (0,1]$, if $\gamma^{\mathcal{C}}_\lambda = \Delta_\lambda(\gamma^{\mathcal{C}}_\lambda)$ then $\delta_\lambda = \gamma^{\mathcal{C}}_\lambda$.*

Proof. By Definition 5, it suffices to prove that if $\gamma^{\mathcal{C}}_\lambda$ is a fixed point for Δ_λ, it is also the least one. We distinguish two cases: when $\lambda < 1$ and $\lambda = 1$.

For $\lambda < 1$, [4, Theorem 6] states that Δ_λ has a unique fixed point. By hypothesis $\gamma^{\mathcal{C}}_\lambda$ is a fixed point for Δ_λ, therefore it is also the least one.

For $\lambda = 1$, we proceed by contradiction. Assume $\delta_1 \neq \gamma^{\mathcal{C}}_1$ and ω be the transition function of $\mathcal{C}$. By $\delta_1 \neq \gamma^{\mathcal{C}}_1$ and Theorem 8, we have that $\delta_1 \sqsubset \gamma^{\mathcal{C}}_1$. Let $\Delta'' \colon [0,1]^{S\times S} \to [0,1]^{S\times S}$ defined by

$$\Delta''(d)(s,t) = \begin{cases} 0 & \text{if } \gamma^{\mathcal{C}}_1(s,t) = 0 \\ \Delta_1(d)(s,t) & \text{otherwise} \end{cases}$$

Since Δ_1 is monotonic so is Δ'', thus Δ'' admits a greatest fixed-point, say g. By $\delta_1 \sqsubset \gamma_1^{\mathcal{C}}$ there exists $s, t \in S$ such that $\delta_1(s,t) < \gamma_1^{\mathcal{C}}(s,t)$, so that $\gamma_1^{\mathcal{C}}(s,t) \neq 0$.

Suppose that $\{(s,t) \mid \gamma_1^{\mathcal{C}}(s,t) = 0\} = \sim$, by [4, Corollary 18], Δ'' has a unique fixed point which corresponds to δ_1. By $\gamma_1^{\mathcal{C}} = \Delta_1(\gamma_1^{\mathcal{C}})$, we have that $\gamma_1^{\mathcal{C}} = \Delta''(\gamma_1^{\mathcal{C}})$, which contradicts the hypothesis that $\delta_1 \neq \gamma_1^{\mathcal{C}}$. It can be shown that there exist $s, t \in S$ such that $\gamma_1^{\mathcal{C}}(s,t) \neq 0$, $s \sim t$ (as proven above), and $g(s,t) = 1$. By Lemma 16 and $\delta_1(s,t) = 0$, it holds that $\gamma_1^{\mathcal{C}}(s,t) < 1$, thus $\gamma_1^{\mathcal{C}} \sqsubset g$. Let m and M be defined as follows

$$m = \max\left\{g(s,t) - \gamma_1^{\mathcal{C}}(s,t) \mid s,t \in S\right\}, \quad M = \left\{(s,t) \mid g(s,t) - \gamma_1^{\mathcal{C}}(s,t) = m\right\}.$$

By $\gamma_1^{\mathcal{C}} \sqsubset g$, $m > 0$. We prove first two properties on M:

$$M \cap \{(s,t) \mid \ell(s) \neq \ell(t)\} = \emptyset \tag{4}$$

$$M \cap \left\{(s,t) \mid \gamma_1^{\mathcal{C}}(s,t) = 0\right\} = \emptyset \tag{5}$$

(4) follows since, for all $\ell(u) \neq \ell(v)$, $\gamma_1^{\mathcal{C}}(u,v) = 1 = g(u,v)$, and $m > 0$. (5) follows by definition of Δ'' and $m > 0$.

Let $(s,t) \in M$, then

$$\begin{aligned}
m &= g(s,t) - \gamma_1^{\mathcal{C}}(s,t) \\
&= \Delta''(g)(s,t) - \Gamma_1^{\mathcal{C}}(\gamma_1^{\mathcal{C}})(s,t) \\
&= \Delta_1(g)(s,t) - \Gamma_1^{\mathcal{C}}(\gamma_1^{\mathcal{C}})(s,t) && \text{(by (5))} \\
&= \Big(\min_{\omega' \in \pi(s,\cdot)\otimes\pi(t,\cdot)} \sum_{u,v \in S} g(u,v) \cdot \omega'(u,v)\Big) - \sum_{u,v \in S} \gamma_1^{\mathcal{C}}(u,v) \cdot \omega((s,t),(u,v)) \\
&\leq \sum_{u,v \in S} g(u,v) \cdot \omega((s,t),(u,v)) - \sum_{u,v \in S} \gamma_1^{\mathcal{C}}(u,v) \cdot \omega((s,t),(u,v)) \\
&= \sum_{u,v \in S} \big(g(u,v) - \gamma_1^{\mathcal{C}}(u,v)\big) \cdot \omega((s,t),(u,v)).
\end{aligned}$$

Since, for all $u, v \in S$, $g(u,v) - \gamma_1^{\mathcal{C}}(u,v) \leq m$ and $\sum_{u,v \in S} \omega((s,t),(u,v)) = 1$, we have that, whenever $\omega((s,t),(u,v)) > 0$, $g(u,v) - \gamma_1^{\mathcal{C}}(u,v) = m$. Thus ω has support contained in M. This means that, for all $(s,t) \in M$, $R_{\mathcal{C}}((s,t)) \subseteq M$. Thus, by (4), we have that $\gamma_1^{\mathcal{C}}(s,t) = 0$, but this contradicts (5). □

Proof (of Lemma 11). We proceed by contraposition. Suppose that for all $s, t \in S$ and for all couplings $\mathcal{D} = \mathcal{C}[(s,t)/\omega']$, $\Gamma_\lambda^{\mathcal{D}}(\gamma_\lambda^{\mathcal{C}})(s,t) \geq \gamma_\lambda^{\mathcal{C}}(s,t)$. This corresponds to say that $\gamma_\lambda^{\mathcal{C}} = \Delta_\lambda(\gamma_\lambda^{\mathcal{C}})$. Then the thesis follows from Lemma 17. □

B Transportation Problem

In 1941 Hitchcock and, independently, in 1947 Koopmans considered the problem which is usually referred to as the (homogeneous) *transportation problem*. This problem can be intuitively described as: a homogeneous product is to be shipped in the amounts $a_1, \dots, a_m$ respectively, from each of m shipping *origins* and received in amounts $b_1, \dots, b_n$ respectively, by each of n shipping *destinations*. The cost of shipping a unit amount from the i-th origin to the j-th destination is $c_{i,j}$ and is known for all combinations (i, j). The problem is to determine an optimal *shipping schedule*, i.e. the amount $x_{i,j}$ to be shipped over all routes (i, j), which minimizes the total cost of transportation.

It can be easily formalized as a linear programming problem

$$\begin{array}{rll} \mathsf{minimize} & \sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j} \cdot x_{i,j} & \\ \mathsf{such\ that} & \sum_{j=1}^{n} x_{i,j} = a_i & (i = 1, \dots, m) \\ & \sum_{i=1}^{m} x_{i,j} = b_j & (j = 1, \dots, n) \\ & x_{i,j} \geq 0 & (i = 1, \dots, m \text{ and } j = 1, \dots, n) \end{array}$$

The set of schedules feasible for a transportation problem, which is formalized as a conjunction of linear constraints, describes a (bounded) convex polytope in $\mathbb{R}^2$, often called transportation polytope.

There are several algorithms in literature which efficiently solve (not necessarily homogeneous) transportation problems. Among these we recall [7,10].

C Experimental Results (detailed data)

In this appendix we provide detailed data of all the experiments discussed in Section 6.

# States	out-degree	Time (s)	# TPs
5	2	0.00400676	0.108108
	3	0.00594318	0.272727
	4	0.0139438	0.89375
	5	0.0219477	1.30233
6	2	0.00418537	0.0829268
	3	0.0115532	0.548936
	4	0.0181834	1.31878
	5	0.0503963	3.01843
	6	0.0659359	3.26496
7	2	0.00381973	0.108844
	3	0.0168408	0.980892
	4	0.045375	2.64329
	5	0.0953065	4.56774
	6	0.171368	6.73054
	7	0.190202	6.6859
8	2	0.00492955	0.0863636
	3	0.0247971	1.34606
	4	0.0574707	3.01171
	5	0.127983	5.57985
	6	0.194809	7.02206
	7	0.316305	9.49161
	8	0.426304	10.7912
9	2	0.00858527	0.292636
	3	0.0259426	1.29074
	4	0.0742248	3.33333
	5	0.17075	5.59023
	6	0.448721	11.3662
	7	0.554762	12.2854
	8	0.745632	13.875
	9	1.11997	17.1407
10	2	0.00921778	0.201481
	3	0.0583405	2.03887
	4	0.154308	4.8458
	5	0.298055	7.61401
	6	0.75986	13.313
	7	0.899531	14.5403
	8	1.81921	22.0646
	9	2.62714	26.4177
	10	3.32188	26.5842
11	2	0.0277172	0.495146
	3	0.0766988	1.82706
	4	0.299206	6.05806
	5	0.72154	11.3061
	6	1.86622	17.7955
	7	1.70401	17.1671
	8	3.19049	33.9219
	9	3.80195	31.5558
	10	4.52554	31.0671
	11	6.20838	36.8718
12	2	0.0154774	0.455709
	3	0.0428891	1.62038
	4	0.250268	6.65647
	5	0.642051	13.3423
	6	2.76465	25.1563
	7	2.35534	23.0476
	8	3.33388	28.0218
	9	8.58611	40.2267
	10	9.53899	41.2665
	11	8.56025	43.165
	12	18.5177	64.8665
13	2	0.011122	0.286942
	3	0.06082	1.88134
	4	0.573837	8.97117
	5	0.964231	14.4766
	6	2.78855	23.8415
	7	6.00371	34.8411
	8	9.11574	39.7318
	9	10.9838	51.6655
	10	15.0667	49.8645
	11	24.1127	65.2042
	12	22.8915	66.0113
	13	32.8509	68.6738
14	2	0.0116639	0.269461
	3	0.0894778	2.79441
	4	0.549372	10.6123
	5	1.61925	17.9549
	6	3.19621	24.761
	7	7.1762	42.2557
	8	10.674	50.3776
	9	20.8513	62.4393
	10	29.753	72.7825
	11	42.3095	85.4986
	12	56.4491	90.0443
	13	75.946	99.7807
20	2	0.0173417	0.5
	3	0.35631	6.36833
	4	1.5266	13.1367
	5	10.0704	38.8383
30	2	0.07566	0.855
	3	4.66113	17.3167
	4	74.8146	76.2642
50	2	0.093105	0.93
	3	27.2147	30.8217
	4	2234.54	225.394