Approximate analysis of probabilistic processes: logic, simulation and games

Josée Desharnais*, François Laviolette*
Dép. d’informatique et de génie logiciel
Université Laval
Québec Canada
{Josee.Desharnais, Francois.Laviolette}@ift.ulaval.ca

Mathieu Tracol
LRI
Université Paris-Sud
France
tracol@lri.fr

Abstract

We tackle the problem of non robustness of simulation and bisimulation when dealing with probabilistic processes. It is important to ignore tiny deviations in probabilities because these often come from experience or estimations. A few approaches have been proposed to treat this issue, for example metrics to quantify the non bisimilarity (or closeness) of processes. Relaxing the definition of simulation and bisimulation is another avenue which we follow. We define a new semantics to a known simple logic for probabilistic processes and show that it characterises a notion of $\epsilon$-simulation. We also define two-players games that correspond to these notions: the existence of a winning strategy for one of the players determines $\epsilon$-(bi)simulation. Of course, for all the notions defined, letting $\epsilon = 0$ gives back the usual notions of logical equivalence, simulation and bisimulation. However, in contrast to what happens when $\epsilon = 0$, two-way $\epsilon$-simulation for $\epsilon > 0$ is not equal to $\epsilon$-bisimulation. Next we give a polynomial time algorithm to compute a naturally derived metric: distance between states $s$ and $t$ is defined as the smallest $\epsilon$ such that $s$ and $t$ are $\epsilon$ equivalent. Finally we show that most of these notions can be extended to deal with probabilistic systems that allow non-determinism as well.

1 Introduction

In program verification, the goal is typically to check automatically whether a system conforms to its pre-established specification. For non-probabilistic systems, one usually expects equivalence between the two, and most of the time this equivalence is chosen to be bisimulation. In the case of probabilistic systems it has been observed [14] that the comparison between processes should not be based on equivalences. One reason is that the binary information given by bisimulation is not precise enough since it identifies close processes as non bisimilar as well as would be very different processes. Another reason is that most of the time the stochastic information comes from observations, or from theoretical estimations. In this context, it is more useful to have a notion of approximate equivalence or distance. Distances or metrics have been defined for LMPs [9, 22] and some work has been done to estimate bisimulation with a certain degree of confidence [13, 11]. Relaxing the definition of simulation and bisimulation is another avenue, which we follow, to tackle the problem of non-robustness of bisimulation.

We propose $\epsilon$-bisimulation and $\epsilon$-simulation to relax the usual notions. We give logical and game characterisations for those two notions.

Games semantics have applications in linear logic, programming languages and model-checking [1]. One important contribution of this paper is such a two players game around an LMP. Simulation is a non-symmetric version of bisimulation. Roughly speaking, a process simulates another one if it can be observed that it can do at least what the other can do. We define a new semantics to a known simple logic for probabilistic processes and prove the characterisation, which is based on the following relation: we say that a state logically simulates another one if it $\epsilon$-satisfies every formula that the other one satisfies. Of course, when letting $\epsilon = 0$, we retrieve the usual notions of logical equivalence, simulation and bisimulation [8].

The notion of $\epsilon$-simulation will define a certain ball around an LMP. Simulation is a non-symmetric version of bisimulation. Roughly speaking, a process simulates another one if it can be observed that it can do at least what the other can do. We define a new semantics to a known simple logic for probabilistic processes and prove the characterisation, which is based on the following relation: we say that a state logically simulates another one if it $\epsilon$-satisfies every formula that the other one satisfies. Of course, when letting $\epsilon = 0$, we retrieve the usual notions of logical equivalence, simulation and bisimulation [8].

Games semantics have applications in linear logic, programming languages and model-checking [1]. One important contribution of this paper is such a two players game for probabilistic systems. In the game which characterises $\epsilon$-simulation, players will have to play the whole game on their assigned system, whereas in the game characterising $\epsilon$-bisimulation, the Adversary will have the possibility to force the Prover to trade places with him. Such a possibility of changing sides is intuitively coherent with the following result: $\epsilon$-bisimulation needs negation for its logical characterisation. One important consequence of this observation is that $\epsilon$-bisimulation is not equal to two-way $\epsilon$-simulation for $\epsilon > 0$, which is in contrast to what happens when $\epsilon = 0$.

*Research supported by CRSNG
We also introduce a fixpoint logic, which can be seen as a fragment of the \( \mu \)-calculus extended to the context of probabilistic systems. We generalize the notion of approximate satisifability to this logic as well. It differs from the quantitative \( \mu \)-calculus of [6, 5] in that it is not a real logic, and is adapted to the kind of relaxations we introduce.

The definition and characterisation of approximate bisimulation and simulation mentioned above are very natural. Moreover, they are tractable. We present an algorithm for computing the maximal corresponding relations on \( S \) of \( \epsilon \)-simulation and \( \epsilon \)-bisimulation that runs in polynomial time in the size of the considered system.

A natural (pseudo-)distance comes out of \( \epsilon \)-bisimulation: two states \( s, t \) are at distance less than or equal to \( \epsilon \) if they are \( \epsilon \)-bisimilar. This distance contrasts with most of the ones defined in the past [9, 21, 13, 6], in that it does not have any discount factor: it can differentiate states which have different behaviors even only in the long term. Such a property is interesting for the study of systems that are not supposed to stop ever. The only other known distance that does not discount the future [9] is intractable. Indeed, the problem of evaluating the distance between two states has recently been proven decidable by Van Breugel et al. [20] but the algorithm has a double exponential complexity. Our algorithm is polynomial.

We generalise this distance to the non-deterministic case, getting a very natural and practicable distance relation. In [10] and [6] an attempt is done in this direction, but the complexity of the procedure is not detailed. The computable distances on probabilistic systems defined in the past, essentially in [21, 13] are based on the computation of a fixed point which can be done in polynomial time only when a discount factor is introduced, which weakens the effects of long term behaviours.

The plan of the paper is as follows. Next section contains background, known definitions and results on LMPs. In Section 3, we define a loose semantics for the usual simple logic introduced Larsen and Skou [15] and collect a few results about it. Our main contribution is in Section 4. We define three notions of \( \epsilon \)-simulation and show that they coincide: a logical one, a structural one, and another one depending on the existence of a winning strategy in a two-players game. At the end of this section we introduce our fixpoint logic. In Section 5 we give an algorithm to compute \( \epsilon \)-simulation efficiently and the corresponding distance is presented in Section 6. Finally, Section 7 is dedicated to a discussion on the generalisation of this work to probabilistic automata. Missing proofs can be found in appendix.

2 Preliminaries

A Labelled Markov Process is a probabilistic transition system whose state space can be continuous. However, in this paper, we restrict to countable state space systems. We will use in that case the natural measure space \( (S, \Sigma) \), where \( \Sigma = \mathcal{P}(S) \). A sub-probability measure on \( S \) is a map \( \mu : \Sigma \rightarrow [0, 1] \) that is countably additive and gives 0 measure to the empty set. If \( X \in \mathcal{P}(S) \) and \( R \subseteq S \times S \), we define \( R(X) = \{ y \in S \mid \exists x \in X \mid xRy \} \). A set \( A \subseteq S \) is \( R \)-closed if \( R(A) \subseteq A \).

2.1 Labelled Markov Processes

A Labelled Markov Process (LMP) is a model of a stochastic system that reacts to actions taken by its environment. The system is in a state at any moment and, given an action (label) performed by the environment, it can jump to another state following a probability distribution [7].

**Definition 1 (LMP).** \( S = (S, \Sigma, h : L \times S \times \Sigma \rightarrow [0; 1], i) \) is a Labelled Markov Process (LMP) if \((S, \Sigma)\) is a measurable space, \( i \) is a distinguished state called the initial state, and for all \( a \in L \) and \( s \in S \), \( h(a, s, A) \) is a sub-probability measure as a function of \( A \).

Since \( S \) is countable we consider \( \Sigma = 2^S \) without specification in the definition of the system. \( h(a, s, A) \), that we write \( h_\delta(s, A) \), is a measure of the likelihood that the LMP will jump to a state in \( A \) given that it was at \( s \) and had to react to action \( a \). Note that a condition of \( \Sigma \)-measurability of \( h_\delta(s, A) \) as a function of \( s \) must be imposed when working with uncountable processes.

LMPs differ from standard Markov chains in that they depend on an auxiliary set \( L \) of actions and in that the transitions are only asked to be sub-probability measures instead of probability measures. Sub-probabilities are to model the possibility of blocking. This is of particular interest when studying simulation since a transition that sums up to less than one can be simulated by a transition with higher probability. With transitions defined as total probability measures, sub-probabilities may be matched make simulation for probabilistic processes very close to simulation for non-probabilistic processes.

2.2 The simple logic \( \mathcal{L} \)

Logics are powerful and well established tools to describe properties that processes must satisfy. We recall the simple logic \( \mathcal{L} \) introduced by Larsen and Skou [15].

**Definition 2 (Logic \( \mathcal{L} \)).** The syntax of \( \mathcal{L} \) is the following:

\[ \mathcal{L} : \quad \theta ::= \top \mid \theta_1 \land \theta_2 \mid \theta_1 \lor \theta_2 \mid \langle \alpha \rangle \delta \theta, \text{ with } \delta \in [0; 1]. \]

The satisfaction relation \( \models \subseteq S \times \mathcal{L} \) is defined by structural induction on the formulas of \( \mathcal{L} \) as usual for \( \top, \land, \lor \) and as
follows for the modal formula:
\[ s \models (\alpha)_\theta \text{ iff } h_\alpha(s, [\theta]) > \delta, \text{ where } [\theta] = \{ s \models \theta \}. \]

Disjunction is not necessary to obtain logical characterisation of bisimulation but it is necessary for simulation \[8\]. Apart from the presence of disjunction, the strict or non-strict inequality in the semantics of \((\alpha)_\theta\) is also a varying parameter in related papers. The original logic was introduced with a non-strict inequality. The choice depends on the use of the logic. We may also add negation to this logic, giving the logic \(L_{\cdot, \cdot}\), which is more expressive but have the same discriminating power. Indeed, there is no formula of \(L\) that is equivalent to, for example, \(\neg \top\). However, two states satisfy the same formulas of \(L_{\cdot, \cdot}\), if and only if they satisfy the same formulas of \(L\).

Simulation and bisimulation. Intuitively, a state simulates another one if it can be observed that it can do at least what the other can do. Of course, this can be formalised in many ways. In the following section we will give three different views of \(\epsilon\)-simulation: through a structural definition of LMPs, through a logic and through a game. For the usual notion of simulation, only the first two have been defined.

The following definition of simulation is a structural one and hence is based on the existence of a particular relation on the states of an LMP.

**Definition 3.** [8] Let \(S = (S, \Sigma, h, i)\) be an LMP. A binary relation \(R\) on \(S\) is a simulation if whenever \(sRt\), we have that for all \(a \in L\) and every \(R\)-closed set \(A \subseteq \Sigma\), \(h_\alpha(s, A) \leq h_\alpha(t, A)\). We say that \(s\) is simulated by \(t\) if \(sRt\) for some simulation relation \(R\).

Simulation can be lifted straightforwardly from states to LMPs through initial states.

The following characterisation shows that the logic is rich enough to represent simulation and conversely that simulation respects the logic. It includes a notion of logical simulation.

**Theorem 1** ([8]). In an LMP \(S\) such that \((S, \Sigma)\) is analytic, in particular when \(S\) is countable, if \(s, t \in S\), then \(s\) is simulated by \(t\) iff for all formulas \(\theta \in L\) we have \(s \models \theta \Rightarrow t \models \theta\).

Bisimulation is a “symmetric” simulation.

**Definition 4** ([8]). Let \(S = (S, \sigma, h, i)\) an LMP. An equivalence relation \(R\) on \(S\) is a bisimulation if whenever \(sRt\), we have that for all \(a \in L\) and every \(R\)-closed set \(A\), \(h_\alpha(s, A) = h_\alpha(t, A)\). We say that \(s\) and \(t\) are bisimilar if \(sRt\) for some bisimulation relation \(R\).

Bisimulation also has a logical characterisation.

**Theorem 2** ([7]). Let \(S\) be an LMP, and \(s, t \in S\). Then \(s\) and \(t\) are bisimilar iff for all formulas \(\theta\) of the logic \(L\) (with or without \(\lor\)) we have \(s \models \theta \iff t \models \theta\).

A direct consequence of the fact that the same logic \(L\) can characterise bisimulation and simulation is that for LMPs, bisimulation is equal to two-way simulation. That is, states \(s\) and \(t\) are bisimilar if and only if \(s\) simulates \(t\) and \(t\) simulates \(s\). When one adds external non-determinism to probabilistic processes (as in probabilistic automata [18]) this is no longer the case. The situation then imitates what is encountered in the non-probabilistic setting.

For the remaining of the paper we fix a countable LMP \(S = (S, \Sigma, h : L \times S \times \Sigma \rightarrow [0; 1], i)\); we will use freely the notation \(s, t\) for states of \(S\).

### 3 Approximate satisfiability: \(\epsilon\)-logic

We now introduce a variation of the logic \(L\) as our first step towards relaxing the notions above. The syntax is the same, only the semantics changes to permit a relaxation that can also be used for strengthening the logic. The satisfaction relation is defined relatively to a real parameter \(\epsilon\), typically in \([-1; 1]\). For \(\epsilon > 0\), \(\epsilon\)-satisfying a formula will be easier than satisfying the formula in the usual sense, whereas if \(\epsilon < 0\), \(\epsilon\)-satisfying a formula will be more difficult. We make the definition precise:

**Definition 5.** Let \(\epsilon \in [-1, 1]\). The syntax of the logic \(L_{\cdot, \cdot}\) is the same as the syntax of \(L\). The approximate satisfaction relation \(|=_{\epsilon}\subseteq S \times L_{\cdot, \cdot}\) is defined by structural induction on the formulas of \(L_{\cdot, \cdot}\) as follows:

- \(s \models_{\epsilon} \top\) for each state \(s\)
- \(s \models_{\epsilon} \neg \theta\) iff \(s \not\models_{-\epsilon} \theta\)
- \(s \models_{\epsilon} \theta_1 \land \theta_2\) iff \(s \models_{\epsilon} \theta_1\) and \(s \models_{\epsilon} \theta_2\).
- \(s \models_{\epsilon} (\alpha)_\theta\) iff \(h_\alpha(s, [\theta]_\epsilon) > \delta - \epsilon\), where \([\theta]_\epsilon = \{ s \in S \mid s \models_{\epsilon} \theta \}\).

If \(\theta \in L_{\cdot, \cdot}\), we say that \(s\) \(\epsilon\)-satisfies \(\theta\) when \(s \models_{\epsilon} \theta\).

We denote by \(L_{\cdot}\) the logic without negation.

If \(\epsilon > 0\), the semantics requires that in order for \(s\) to \(\epsilon\)-satisfy \((\alpha)_\theta\), it only suffices that with the \(\alpha\)-labelled transition from \(s\), we can jump in \([\theta]_0\), a slightly bigger set than the usual one \([\theta]\), with a probability \(\delta - \epsilon\) slightly smaller than the usual one \(\delta\). On the contrary, if \(\epsilon < 0\), we demand that in order for \(s\) to \(\epsilon\)-satisfy \((\alpha)_\theta\), it must be that with the \(\alpha\)-labelled transition from \(s\), we can jump in a slightly smaller set \([\theta]_\epsilon\), than the usual one \([\theta]\), with a probability \(\delta - \epsilon\) slightly bigger than the usual one. The case of negation is natural: if \(\epsilon > 0\), then \(s \models_{\epsilon} \neg \theta\) means \(s\) is close to satisfy \(\neg \theta\). We say that this corresponds to the fact that \(s\) does not strongly satisfy \(\theta\). Accordingly, if \(\epsilon < 0\), then \(s \models_{\epsilon} \neg \theta\) means \(s\) makes a little bit more than satisfying \(\neg \theta\). This correspond to the fact that \(s\) is not close to satisfying \(\theta\).

We will see that this new semantics diminishes the expressive power of the logic. This could be considered a
Some relation of \( \forall \epsilon \) of approximate simulation. The most important notion is \( \epsilon \text{-} \text{simulation} \).

**Definition 6.** Let \( \epsilon \in [-1; 1] \). By structural induction on the formulas of \( \mathcal{L}_\epsilon \), we construct for each \( \theta \in \mathcal{L}_\epsilon \), an associated formula \( \theta_\epsilon \in \mathcal{L}_\epsilon \): \( \tau_\epsilon = \top; (\theta_1 \land \theta_2)_\epsilon = (\theta_1)_\epsilon \land (\theta_2)_\epsilon; (\langle a \rangle \delta)_\epsilon = (\langle a \rangle)_\delta \epsilon; (\neg \theta)_\epsilon = \neg (\theta_\epsilon) \).

Here we use the fact that \( (\langle a \rangle)_\lambda \theta \) is still a valid formula, even if \( \lambda < 0 \) or \( \lambda > 1 \), which gives in turn that \( (\theta_\epsilon)_\epsilon = \theta \).

Clearly the transformation is additive: if \( \theta \in \mathcal{L}_\epsilon \) and \( \epsilon' \in [-1; 1] \), then \( \theta_\epsilon \epsilon' \) is the formula \( \theta_{\epsilon+\epsilon'} \).

**Example 1.** If \( \theta = \langle a \rangle_5 (\neg \langle a \rangle_2 \top) \), then \( \theta_\epsilon = \langle a \rangle_{5-\epsilon} (\neg \langle a \rangle_{2+\epsilon} \top) \).

**Proposition 1.** Let \( \theta \in \mathcal{L}_\epsilon \) and \( \epsilon' \in [-1; 1] \). Then \( [\theta]_\epsilon = [\theta]_\epsilon \), where \( [\theta] = \{ s \in \mathcal{S} | s \models \theta \} \).

The proof is an easy structural induction on the formulas.

This proves that \( \mathcal{L}_\epsilon \) is more expressive than \( \mathcal{L}_{\epsilon-} \). Nevertheless, it will make the correspondence between simulation and logic smoother. Indeed, it calls naturally the following approximate simulation: a state \( \hat{s} \in \mathcal{S} \) will be \( \epsilon \)-simulated by a state \( t \in \mathcal{S} \) if \( t \) can match the transitions that \( \hat{s} \) can make, up to \( \epsilon \).

**Definition 7 (Logical \( \epsilon \)-simulation).** Let \( \epsilon \in [0; 1] \) and \( s, t \in \mathcal{S} \). We will say that \( t \epsilon \)-simulates \( s \), written \( s \epsilon^L \leq t \), if for all formula \( \theta \in \mathcal{L} \) we have:

\[
s \models \theta \Rightarrow t \models^\epsilon \theta.
\]

Clearly \( \epsilon^L \) is not a transitive relation. However, using the fact that \( s \models^\epsilon \theta \Leftrightarrow t \models^\epsilon \theta \), it can be shown easily that if \( s, t, u \in \mathcal{S} \) such that \( s \epsilon^L \leq t \) and \( t \epsilon^2 \leq u \), then \( s \epsilon^L \leq u \) or \( s \epsilon^L \leq u \). Of course we cannot use the logic with negation to deal with simulation.

## 4 \( \epsilon \)-simulation and \( \epsilon \)-bisimulation

In what follows our aim is to give two characterisations of approximate simulation. The most important notion is a generalisation of the structural definition of simulation. Following Definition 3, we define relations of \( \epsilon \)-simulation. Next we give a definition in terms of a two players game, similarly as what is done in the context of concurrent systems. We will show that these three notions coincide.

**Definition 8 (Relations of \( \epsilon \)-simulation).** A relation \( \mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \) is a relation of \( \epsilon \)-simulation if whenever \( s \mathcal{R} t \), then for all \( \epsilon \in [0; 1] \), \( a \in \mathcal{L} \) and \( X \subseteq \mathcal{S} \),

\[
h_a(t, \mathcal{R}(X)) \geq h_a(s, X) - \epsilon.
\]

We say that \( t \) is \( \epsilon \)-simulated by \( s \), written \( s \epsilon \leq t \), if \( s \mathcal{R} t \) for some relation of \( \epsilon \)-simulation \( \mathcal{R} \) on \( \mathcal{S} \).

As for usual simulation [3], for a countable state space model, \( \sim_{\epsilon} \) is the largest relation of \( \epsilon \)-simulation and it can be computed as a greatest fixed point. Let \( F : 2^\mathcal{S} \times \mathcal{S} \) associate to any relation \( \mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \) the relation \( F(\mathcal{R}) \subseteq \mathcal{S} \times \mathcal{S} \) such that \( s \mathcal{R}(t, \mathcal{R}(X)) \) if for all \( a \in \mathcal{L} \) and all set \( E \subseteq \mathcal{S} \), we have \( h_a(t, \mathcal{R}(E)) \geq h_a(s, E) - \epsilon \). Then \( \sim_{\epsilon} \) will be the greatest fixpoint of \( F \) in \( 2^\mathcal{S} \times \mathcal{S} \).

**Example 2.** Let \( \epsilon > 0 \). Consider the following systems:

\[
S : \begin{array}{c}
 s \stackrel{\epsilon/2}{\longrightarrow} t \\
 E(a..9) \\
 u\end{array}
\]

\[
T : \begin{array}{c}
 t \stackrel{\epsilon/2}{\longrightarrow} u \\\n a..9+\epsilon \\
 F(a..e)
\end{array}
\]

For any \( \epsilon' > \epsilon \) we have \( s \epsilon_{\epsilon'} \leq t \) and \( t \epsilon_{\epsilon'} \leq s \).

These systems are typical of the analysis that we propose. Indeed, starting from the statement that a model of a system cannot be exact, we are interested in comparing models, but with a certain degree of tolerance. More precisely, \( s \) can be seen as a model of the real system \( t \) (and conversely): we would like to consider them as equivalent if we can tolerate an error of \( \epsilon \).

We now show that the two notions of approximate simulation that we have defined so far, logical \( \mathcal{L}_\epsilon \)-simulation and structural \( \epsilon \)-simulation, are in fact equivalent.

**Theorem 3.** \( s \epsilon_{\epsilon} \leq t \) iff \( s \epsilon \leq t \).

## 4 \( \epsilon \)-simulation and \( \epsilon \)-bisimulation

Games for \( \epsilon \)-simulation. We introduce a new notion of approximate simulation in terms of game and show that it is also equivalent to \( \epsilon \)-simulation \( \sim_{\epsilon} \).

**Definition 9 (Game \( \mathcal{S}, s, t \)).** Two players, Prover (P), whose goal is to prove that \( s \) is simulated by \( t \), and Adversary (A), are playing the following game on \( \mathcal{S} \). The game consists in the repetition of the following steps:

- **Step 1:** A chooses a label \( a \) and a set \( E \subseteq \mathcal{S} \).
- **Step 2:** P must choose a set \( F \subseteq \mathcal{S} \), such that \( h_a(t, F) \geq h_a(s, E) - \epsilon \).
- **Step 3:** A chooses \( t' \in F \) from which P will have to play in the next sequence of steps.
- **Step 4:** P chooses \( s' \in E \) from which A will have to play in the next sequence of steps.

If one player gets stuck at some point, for instance if A cannot choose any transition from \( s \), or if P cannot find a relevant set \( F \) during Step 2, he loose. If the game continues without ever stopping, Prover wins. We say that A (resp. P) has a winning strategy if there exists a sequence of choices for the steps 1 and 3 (resp. 2 and 4), responding to the choices of P (resp. A), such that anyhow being the responses of P (resp. of A), such a sequence of moves leads to a victory of A (resp. of P).

We write \( s \epsilon_{\epsilon} \leq t \) if P has a winning strategy.
This game seems quite simple but it is rich enough to characterise \( \epsilon \)-simulation.

**Theorem 4.** \( s \sim^G_\epsilon t \) iff \( s \sim_\epsilon t \) iff \( s \sim^G_0 t \).

**Proof.** Only the first equivalence is left to prove. We sketch the proof. \((\Rightarrow)\) is a direct consequence of the fact that \( \sim^G_\epsilon \) is a relation of \( \epsilon \)-simulation.

\((\Leftarrow)\): given \( R \) a relation of \( \epsilon \)-simulation on \( S \) such that \( sRt \), we construct a winning strategy for \( P \). For this we point out that at Step 2, \( P \) can choose the set \( F = R(E) \) such that \( h_a(t, F) \geq h_a(s, E) - \epsilon \). But then whatever being the choice of \( t' \in F \) at Step 3, since \( F \subseteq R(E) \), there will always exist \( s' \in E \) such that \( s'Rt' \). With this choice we are back to the initial situation, and we see that \( P \) cannot loose, which means that he has a winning strategy.

Now that we have defined the logic \( L_\epsilon \) and \( \epsilon \)-simulation, we are ready to face the question of approximate equivalence. Of course, it will not be an equivalence. We cannot ask for our approximate equivalence relation to be transitive: two states \( s, t \) with quite different behaviours could be linked by a long sequence close states. It is quite natural, however, to define \( \epsilon \)-simulation as a symmetric \( \epsilon \)-simulation.

**Definition 10** (Relation of \( \epsilon \)-bimonalisation). A relation \( R \subseteq S \times S \) is a relation of \( \epsilon \)-bimonalisation if it is symmetric and a relation of \( \epsilon \)-simulation. Two states \( s \) and \( t \) are \( \epsilon \)-bimonalisim, written \( s \sim_\epsilon t \), if there exists a relation of \( \epsilon \)-bimonalisation \( R \) on \( S \) such that \( sRt \).

In Example 2 we have \( s \sim_\epsilon t \) for all \( \epsilon' > \epsilon \).

Note that the symmetry of the relation could be replaced by the requirement that the inverse of \( R \) be a relation of \( \epsilon \)-simulation as well as \( R \).

This definition looks similar to the one introduced by Giacalone et al. [14] but it is more faithful to probabilistic notions by allowing to regroup states when matching a transition. Indeed, Giacalone et al. define the \( \epsilon \)-matching of transition probabilities through states instead of sets. This is key because matching states impose that the transition structure of the two processes be equal. This definition coincides with the usual definition of bisimulation when \( \epsilon = 0 \) and is natural when we think in terms of games.

As for \( \epsilon \)-simulation, \( \sim_\epsilon \) can be computed iteratively for countable state space model as a greatest fixpoint: \( \sim_\epsilon \) is nothing more than the greatest symmetric fixpoint of the relation transformer \( F \) on \( 2^{R \times S} \).

We now continue the parallel construction: we present a two-players game that will give another definition for an approximate bisimulation. We will show that this new notion coincides with \( \epsilon \)-bimonalisation.

**Definition 11** (Game \( S, x, y \) for \( \epsilon \)-bimonalisation). Two players, Prover (\( P \)) and Adversary (\( A \)), are playing the following game on \( S \). The game consists in a sequence of steps which are repeated:

- **Step 0**: \( A \) chooses a state \( s \in \{x, y\} \). \( P \) will play on \( s \) whereas \( P \) will play on \( t \in \{x, y\} \setminus \{s\} \).
- **Step 1**: \( A \) chooses a label \( a \) and a set \( E \subseteq S \).
- **Step 2**: \( P \) chooses a set \( F \subseteq S \), such that \( h_a(t, F) \geq h_a(s, E) - \epsilon \).
- **Step 3**: \( A \) chooses a state \( x' \) in \( E \) or \( F \).
- **Step 4**: \( P \) chooses a state \( y' \) in the set not chosen by \( A \). This way we get one state in \( E \) and one state in \( F \).

The winning conditions are the same as for the game for \( \epsilon \)-simulation. We write \( s \sim^G_\epsilon t \) if \( P \) has a winning strategy in the game \( S, s, t \).

We show that the two previous notions of approximate bisimulation coincide.

**Theorem 5.** \( P \) has a winning strategy in the game \( S, s, t \) iff there exists a relation of \( \epsilon \)-bimonalisation \( R \) on \( S \times S \) such that \( sRt \). That is \( s \sim^G_\epsilon t \) iff \( s \sim_\epsilon t \).

**Proof.** \((\Rightarrow)\) As in Theorem 4, we show that \( \sim^G_\epsilon \) is a relation of \( \epsilon \)-bimonalisation, which is a mere reformulation of the game rules. The fact that \( \sim^G_\epsilon \) is a symmetric relation comes directly from the definition of the game: we allow \( A \) to choose his starting state at each round. If \( P \) has a winning strategy, then we will have \( s \sim^G_\epsilon t \) as well as \( t \sim^G_\epsilon s \).

\((\Leftarrow)\) This direction is also the same as for Theorem 4, the fact that \( R \) is symmetric corresponding here again to the choice of \( A \) for his start point.

The next step would be to show that a logical notion of approximate equivalence, \( \sim^G_\epsilon \), and the game notion \( \sim^G_\epsilon \) or the structural notion \( \sim_\epsilon \) coincide. However something subtle happens, and we have to differentiate the case when \( \epsilon = 0 \) from the other cases. Indeed, we already know by Theorem 2 that the logical notion of bisimulation \( \sim^G_0 \) and the structural notion of bisimulation \( \sim_0 \) coincide. Something fascinating happens when \( \epsilon > 0 \): the situation then imitates what is encountered for non deterministic bisimulation. Two-way simulation and bisimulation disagree, and hence, negation is necessary in the logic to characterise \( \epsilon \)-bimonalisation for \( \epsilon > 0 \).

**Proposition 2.** For \( \epsilon \in [0; 1] \), two-way \( \epsilon \)-simulation is weaker than \( \epsilon \)-bimonalisation. That is, there exists a finite LMP with two states \( s, t \) such that \( s \sim_\epsilon t \) and \( t \sim_\epsilon s \), but \( s \not\sim^G_\epsilon t \).

**Proof.** We give a particular example with \( \epsilon = 1/8 \). Clearly this example could be modified to handle any \( \epsilon \in [0; 1] \).
Then \( s \sim_{1/8} t \) and \( t \sim_{1/8} s \), since \( x \) is 1/8-simulated by both \( s_1 \) and \( s_2 \). On the other hand, \( s \not{\sim_{1/8}} t \) because we have \( \{ s \in S \mid x \sim_{1/8} s \} = \{ x \} \). But \( h_\varnothing(t, \{ x \}) = 4/8 \), which is strictly greater than \( h_\varnothing(t, \{ x \}) + 1/8 = 3/8 \).

We now show that \( L_{\sim_\varepsilon} \) characterises \( \varepsilon \)-bisimulation.

**Definition 12** (Logical \( \varepsilon \)-equivalence \( \sim^\varepsilon \)). Let \( \varepsilon \geq 0 \). Then \( s \) and \( t \) are said to be \( \varepsilon \)-equivalent for \( L_{\sim_\varepsilon} \), written \( s \sim^\varepsilon \) \( t \), iff \( s \sim^\varepsilon \) \( t \). As usual, this means \( s \models \theta \Rightarrow t \models \theta \) for any \( \theta \in L_{\sim_\varepsilon} \).

It is easy to see that this defines a symmetric relation.

**Theorem 6.** Let \( \sim_\varepsilon \) be the largest relation of \( \varepsilon \)-bisimulation on \( S \). Then we have: \( s \sim_\varepsilon t \text{ iff } s \sim^\varepsilon \) \( t \).

**Proof.** We sketch the proof. For \( \Rightarrow \), we suppose \( s \sim_\varepsilon t \) and we show by induction on the structure of the formulas of \( L_{\sim_\varepsilon} \) that for all \( \theta \in L_{\sim_\varepsilon} \), we have \( s \models \theta \Rightarrow t \models \theta \). The interesting cases are \( \theta = \langle a \rangle \delta \phi \) and \( \theta = \neg \phi \).

Suppose \( \theta = \langle a \rangle \delta \phi \). Then \( h_\varnothing(s, [\phi]) > \delta \). By hypothesis \( s \sim_\varepsilon t \), hence if we take \( Y = \sim_\varepsilon ([\phi]) \) we get \( h_\varnothing(t, Y) \geq h_\varnothing(s, [\phi]) - \varepsilon \). But for all \( x \in \sim_\varepsilon ([\phi]) \), there exists \( y \in [\phi] \) such that \( x \sim_\varepsilon y \), hence, by induction hypothesis, for all \( x \in \sim_\varepsilon ([\phi]) \) we have \( x \models \phi \). This precisely shows that \( t \models \theta \).

Suppose \( \theta = \neg \phi \). We have to show that \( t \models \neg \phi \), i.e. that \( t \models \langle - \phi \rangle \), i.e. that \( t \models \neg \phi \), i.e. that \( t \models \neg \phi \), i.e. that \( t \models \phi \). But \( t \models \neg \phi \) would imply that induction shows \( s \models \neg \phi \), i.e. that \( s \models \phi \) which is a contradiction.

For \( \Leftarrow \), we just have to show that \( \sim^\varepsilon \) is a relation of \( \varepsilon \)-bisimulation. We can follow the proof of Theorem 3, using the fact that \( \sim^\varepsilon \) is a symmetric relation.

The presence of negation in the logic reflects the fact that we allow Adversary to change sides during the game. We give a formula that distinguishes the two states \( s \) and \( t \) of Proposition 2’s example:

\[
\phi = \langle a \rangle.49 (b) \neg \langle \langle d \rangle \rangle T.
\]

Then \( s \models \phi \), but \( t \not{\models_{1/8}} \phi \). This formula is in \( L_{\sim_\varepsilon} \).

**A fixpoint logic for LMPs** A fixpoint logic allows to deal with long term and global properties, which may be interesting in the study of probabilistic processes. Many generalisations have been proposed ([9, 6, 5, 16]). Ours differs from the quantitative \( \mu \)-calculus of [6, 5] in that it is not a real logic, and is adapted to the kind of relaxations we introduce. We restrict to finite LMPs but the results can be easily generalised to countable state space systems.

**Definition 13** (The fixpoint logic \( \mathcal{L}^\mu \)).

**Syntax.** The syntax of \( \mathcal{L}^\mu \) is defined with respect to a set \( \mathcal{V} \) of calculus variables. It is defined as follows:

\[
\mathcal{L}^\mu : \phi ::= \theta \mid \langle a \rangle \delta \phi \mid \neg \phi \mid \phi_1 \land \phi_2 \mid \phi_1 \lor \phi_2 \mid \langle a \rangle Z \mid Z \mid \mu Z.\phi \cup \nu Z.\phi.
\]

For \( \theta \) any formula of \( \mathcal{L} \), \( a \in \mathcal{L} \), \( \delta \in [0; 1] \), and calculus variables \( Z \in \mathcal{V} \). As usual in \( \mu \)-calculus, in formulas \( \mu Z.\phi \) and \( \nu Z.\phi \) we require that all occurrences of the bound variable \( Z \) in \( \phi \) occur in the scope of an even number of occurrences of \( \sim \). A formula \( \phi \) is closed if every calculus variable \( Z \) in \( \phi \) occurs in the scope of a quantifier \( \mu Z \) or \( \nu Z \). For simplicity, we will do the following abuse of notation, denoting by \( q \mu \) the set of closed formulas of \( q \mu \). We ask that the formula \( \phi \) in any sub-formula \( \langle a \rangle \delta \phi \) is a closed formula. The fact that we cannot have \( \langle a \rangle \delta \) before a non-closed formula \( \phi \), except if \( \delta = 1 \), will weaken this logic in an interesting way.

**Substitution.** As for the \( \mu \)-calculus, a variable valuation \( \xi : \mathcal{V} \rightarrow 2^S \) is a function that maps every variables \( Z \in \mathcal{V} \) to a set of states. We write \( \xi[Z \rightarrow X] \) for the valuation that agrees with \( \xi \) on all variables, except that \( Z \) is mapped to \( X \). Given a valuation \( \xi \) and a formula \( \phi \in \mathcal{L}^\mu \), the semantics of \( \mathcal{L}^\mu \) gives us a subset \( [\phi]_\xi \) of \( S \):

\[
\begin{align*}
[\theta]_\xi &= [\theta] \text{ for all } \theta \in \mathcal{L} \quad [Z]_\xi = \xi(Z) \\
[\mu Z.\phi]_\xi &= \{ S' \subseteq S \mid [\phi]_{\xi[Z \rightarrow S']} \subseteq S' \} \\
[\nu Z.\phi]_\xi &= \{ S' \subseteq S \mid [\phi]_{\xi[Z \rightarrow S']} \supseteq S' \}.
\end{align*}
\]

In this definition, \([\mu Z.\phi]_\xi \) and \([\nu Z.\phi]_\xi \) are respectively the lowest and the greatest fixpoint of the following (monotonic) function:

\[
g : \mathcal{P}(S) \rightarrow \mathcal{P}(S), S' \mapsto [\phi]_{\xi[Z \rightarrow S']}.
\]

As usual the fixpoints are well defined by the continuity and monotonicity of all operators and can be computed by Picard iteration.

**Example 3.** Let \( \phi \in \mathcal{L} \). Consider the formula \( \psi_1 = \mu Z.\langle a \rangle Z \lor \phi \). Then \([\psi_1]_\xi \) is the set of \( s \in S \) from which the system can do action \( a \) with probability \( 1 \) until it reaches a state satisfying \( \phi \).

Now we would like to extend our relaxation of the satisfiability of formulas of \( L_{\sim_\varepsilon} \) to formulas of \( \mathcal{L}^\mu \). The point of our definition will be that we relax the transition probabilities only outside fixpoint operators: we do not change the 1 in the expression \( \mu Z.\langle a \rangle Z \), but we relax the other values. We give a short example to illustrate the definition of the relaxed formula \( \phi_\varepsilon \), given \( \phi \in \mathcal{L}^\mu \).
Example 6. Let $\phi = \langle c \rangle (\mu Z.((a)_1 Z \lor \neg (b)_2 \top))$. Let $\epsilon \in [0, 1]$. Then $\phi_\epsilon = \langle c \rangle (\mu Z.((a)_1 Z \lor \neg (b)_2 \top))$.

Even if the logic $L^\epsilon$ is more expressive, it has the same discriminating power as $L$. However, $\sim^\epsilon$ has a discriminating power between $\sim$ and $\sim_\epsilon$.

Proposition 3. $\sim \subseteq \sim^\epsilon \subseteq \sim_\epsilon$.

The following example will prove the strictness of the second inclusion. It exhibits systems that could be considered as being related in the same way as do the ones of Example 2. However, they are the key to a very important difference between $\sim^\epsilon$ and $\sim_\epsilon$. One could say a weakness of $\sim_\epsilon$. However we will argue that this case is very peculiar.

Example 5. In the following processes, it could be interesting, from a probabilistic point of view, to see that $s$ and $t$ have different behaviors in the long term.

$$S: s \xrightarrow{a, \epsilon} s \xrightarrow{a, 1 - \epsilon} t; T: t \xrightarrow{a, 1}$$

Indeed $s$ "leaks": a run on the system $S$ will eventually always end in state $s_1$ from which it cannot move. However a run on the system $T$ will always continue to pass through state $t$. We know that in that case we have $s \sim_\epsilon t$, for any $\epsilon' > \epsilon$. The fixpoint logic will differentiate the two: for any $\epsilon' \geq 0$ we have $s \neq^\epsilon_{\epsilon'} t$. Consider the following $L^\epsilon$ formula: $\phi = \nu Z.((a)_1 Z)$. We construct $[\phi]^S$ and $[\phi]^T$. We write $X_i$ for the increasing sequence of Picard iterates for $S$.

We have: $X_0 = \{s, s_1\}$, $X_1 = \{[(a)_1 Z]_{Z=s, s_1} = \{s\}$, $X_2 = \{[(a)_1 Z]_{Z=s_1} = \emptyset$. Hence $[\phi]^S = \emptyset$, but $[\phi]^T = \emptyset$. As well, for any $\epsilon' > 0$, $[\phi]^S_\epsilon = \emptyset$, and $[\phi]^T_\epsilon = \emptyset$.

This proves that $s \neq^\epsilon_{\epsilon'} t$.

This result is coherent with the fact that for the distances $d^\epsilon$ of Desharnais et al., we have $\lim_{\epsilon \to 1} d^\epsilon(s, t) = 1$.

One could argue that this example shows that $\sim^\epsilon$ is better than $\sim_\epsilon$. However, we believe that this undesirable behavior of $\sim_\epsilon$ will only happen in extreme cases like this one. When we consider models involving cycles, more precisely one of which is taken with probability exactly 1 and the other with a closed to 1 probability, we are confronted to a 0/1 result. Indeed, in the limit, the first cycle will be taken infinitely often with probability 1 whereas the other will fail with probability 1. We believe that claiming value 0 or 1 is a strong commitment when studying probabilistic systems. For one thing, this cannot be obtained through experimentation.

As a conclusion we can say that the introduction of fixpoint formulas in the logic $L$ has made possible to differentiate more states, and the difference is made on variations of probabilities of long term runs.

Example 6. This is a tiny variation of Example 2. Let

$$S: s_1 \xrightarrow{b, \frac{1}{2}} s \xrightarrow{a, 1} T: t_1 \xrightarrow{b, \frac{1}{2} - \epsilon} t \xrightarrow{a, 1}$$

These systems of Example 2 are $\sim^\epsilon$-close for any $\epsilon' > \epsilon$.

5 Decision procedures for $\epsilon$-simulation and $\epsilon$-bisimulation

So far we have introduced new notions of approximate simulation and equivalence. In what follows we present a generalisation of the algorithms introduced by Baier [2] to deal with our approximate notion of simulation. We make a link between the characterisation of the simulation relations in terms of right-closed sets, and the characterisation in terms of maximum flows in suitable networks.

Given $\epsilon \geq 0$, one could be interested in computing efficiently the largest relation of $\epsilon$-simulation, that is $\sim_\epsilon$. Recall that in the finite case, $\sim_\epsilon$ can be computed as the greatest fixpoint of the relation transformer $F$, that is $\sim_\epsilon = \bigcap_{n \in \mathbb{N}} R_{n, \epsilon}$, where the $R_{n, \epsilon}$ are defined inductively on $n$ as:

- $R_{0, \epsilon} = S \times S$.
- $R_{n+1, \epsilon} \subseteq R_{n, \epsilon}$ iff for all $\alpha \in L$ and all set $E \subseteq S$, $h_\alpha(t, R_{n+1, \epsilon}(E)) \geq h_\alpha(s, E) - \epsilon$.

Similarly as is done by Baier [2], and Segala and Lynch [18] for $\sim$, one can compute $\sim_\epsilon$ efficiently as follows. Since $|R_{\epsilon, 0}| = |S|^2$ and since $R_{\epsilon} = R_{\epsilon, n}$ whenever $R_{\epsilon, n} = R_{\epsilon, n+1}$, we have that $\sim_\epsilon = \bigcap_{n \in \mathbb{N}} R_{\epsilon, n}$ and therefore it can be computed in $|S|^2$. The following important theorem will allow us to determine efficiently if, roughly speaking, a distribution $\nu$ is greater than another distribution $\mu$, up to $\epsilon$.

Theorem 7. Let $\mu, \nu$ be two distributions on a finite state space $S$, and $\mathcal{R} \subseteq S \times S$ be a binary relation on $S$. Then the following are equivalent:

- For all set $E \subseteq S$ we have $\nu(\mathcal{R}(E)) \geq \mu(E) - \epsilon$.
- The maximum flow in the network $N(\mu, \nu, \mathcal{R})$ is greater than or equal to $\mu(S) - \epsilon$.

Here the network $N(\mu, \nu, \mathcal{R})$ is a flow network of size $O(|S|)$, that we define in the appendix.

Proof. The proof uses methods of network theory. \(\square\)

We can now give an algorithm to compute $\sim_\epsilon$ iteratively.

Algorithm 1. Input: A finite state LMP $S = (S, L, h)$, $\epsilon \in [0; 1]$.

Output: The relation $\sim_\epsilon$ on $S \times S$.

method:
- Let $R_0 = S \times S$.
- For $j = 0$ to $|S|^2$ do begin:
  - $R_{j+1} = R_j$.
- For all $(s, t) \in R_j$ do begin:
  - For all $a \in L$ do begin:
If the maximum flow of the network \( \mathcal{N}(h_a(s, -), h_a(t, -), R_j) \) is strictly smaller than \( h_a(s, S) - \epsilon \), then \( R_j = R_j \setminus \{s, t\} \).

end
end
return \( R_{[S]}^2 \).

Using an algorithm that computes the maximum flow of the networks \( \mathcal{N}(h_a(s, -), h_a(t, -), R_j) \) in time \( O(|S|^3) \) (for instance \([19]\)), we get an algorithm for computing \( \sim \) on an LMP \( S \) in time \( O(|S|^7 \cdot |L|) \).

Now, if we want to compute \( \sim \), it will probably not be possible to use methods of partitions refinement \([2, 4]\). Indeed these methods, which construct the bisimulation relation \( \sim \) on an LMP more efficiently than the simulation relation \( O(n^3) \) opposed to \( O(n^7) \), rely fundamentally on the fact that \( \sim \) is an equivalence relation, what \( \sim \) is not in the general case if \( \epsilon > 0 \). Still, since \( \sim \) is the greatest symmetric fixpoint of \( F \) it can be computed as \( \sim = \cap_{n \in \mathbb{N}} R_{\epsilon, n} \), where the \( R_{\epsilon, n} \) are defined inductively on \( n \) as follows:

- \( R_{\epsilon, 0} = S \times S \).
- If \( s, t \in S \), then we have \( sR_{\epsilon, n+1} t \) iff for all \( a \in L \) and all set \( E \subseteq S \) we have:
  \[ -h_a(t, R(E)) \geq h_a(s, E) - \epsilon. \]
  \[ -h_a(s, R(E)) \geq h_a(t, E) - \epsilon. \]

As for \( \sim_{\epsilon} \), \( \sim \) can clearly be computed as \( R_{\epsilon, [S]}^2 \) if \( S \) is finite. As for \( \sim_{\epsilon} \), using maximum flow algorithms, we get an algorithm for computing \( \sim_{\epsilon} \) on an LMP \( S \) in time \( O(n^7) \).

6 A distance between states using \( \sim_{\epsilon} \)

A main issue concerning models of probabilistic systems is the problem of defining a relevant distance between states of a probabilistic model \([9, 21, 13]\). Most works are based on a distance that can be defined as the maximal difference of valuation of functions associated to formulas for the two considered states. The idea is: the longest is a distinguishing formula, the closest should be the states. An important characteristic of these metrics is that differences are accumulated while time evolves. This can be countered with the use of a discount factor which lowers the effects of long terms events, making the distances lie mostly on short time evolution of the system. Unfortunately, this approach has an undesirable side effect, that every difference become negligible after a certain horizon of time. Such a characteristic might be reasonable in the case of the study of “non-permanent” systems, that is, for instance systems which are supposed to stop working after a while. But this is a weakness in the case of the study of permanent systems, for instance for the study of the long term behavior of a protocol between two persons, which is not supposed to stop ever.

We now propose a pseudo-distance between states of a probabilistic system from the notions we presented. This distance does not accumulate differences through evolution of time and still does not suffer from a finite horizon vision as do the discounted distances.

We define \( d \) on \( S \times S \) as follows:

\[
d : S \times S \rightarrow [0; 1] \\
(s, t) \mapsto \inf \{ \epsilon \in [0; 1] \mid s \sim_{\epsilon} t \}.
\]

The function \( d \) is a pseudo-distance function on the states of the considered LMP: the triangular inequality comes from the fact that if \( \epsilon_1, \epsilon_2 \in [0; 1] \), then \( s \sim_{\epsilon_1} u \) and \( u \sim_{\epsilon_2} t \) implies \( s \sim_{\epsilon_1 + \epsilon_2} t \).

Theorem 8. The kernel of \( d \), i.e. the set of pairs at zero distance, is \( \sim_{\epsilon} \), the relation of bisimulation on \( S \).

Since we have produced in the last section an algorithm to compute relation \( \sim_{\epsilon} \) on a finite LMP \( S \) in time \( O(|S|^7) \), we can in a straightforward way, using a dichotomic approach, compute the distance \( d(s, t) \) between two states of \( S \) up to precision \( \delta \in [0; 1] \) in time \( O(|S|^7 \cdot \ln(\delta)) \).

The distance \( d \) is a new distance, incomparable with the distances presented in \([9, 21, 13]\). We give two examples to show this fact. In the following, \( d^c, c \in [0; 1] \) is the distance with discount factor \( c \) that is studied in the cited papers.

Example 7. In the following system, \( d(s, t) \leq 0.1 \) since \( s \sim_{1} t \), but for all \( c \in [0; 1] \) we have \( d^c(s, t) \geq (c^9 - c^9.95^4) \).

Example 8. In the next example for all \( c \in [0; 1] \) we have \( d^c(s, t) \leq c^9 \), but \( d(s, t) \geq .9 \) since \( s \not\sim_{.9} t \).

Example 5 also underlines the fact that anyhow we define a distance on a probabilistic system, we will always have to choose between giving more weight to close behaviours or to long term behaviours, since the two are not compatible. With the distance \( d \) we have decided to keep the possibility of distinguishing between states which behave differently only in the far future. Since we also want not to differentiate much between states which behave closely in the near
future, it is not possible for the same distance to differentiate strongly between states which behave almost the same for a long time. As we have seen, using our fixpoint logic can be a solution to enrich our distance.

The last examples underline important differences between the two distances:

- The distances $d^c$, $c \in [0; 1]$ differentiate states that have different short range evolutions, whereas distance $d$ can still separate states whose behaviors diverge only in the long range, as in Example 5.

- The distance $d$ may be unable to differentiate strongly two states which have a slightly different behavior, but for a long time.

As a conclusion we can say that these two kinds of distances are not comparable. We consider that the properties of $d$ are desirable. The first one is important because we want to measure differences for a long horizon. For instance for systems which are not supposed to stop evolving ever. The second one may be desirable because probabilities are estimates and therefore, small errors should be ignored. Also, we may want not to accumulate small errors along the run. However, as presented in example 5, in some case we want to accumulate such error. A way to handle this problem for our distance is to consider its generalisation to the fixpoint logic we introduced. We confer to the appendix for a discussion on this topic. Another important property of this distance is that it can be easily generalised to other formalisms, as is discussed in the next section.

7 Non deterministic processes

Probabilistic automata, such as presented by Segala [18], can be seen as a generalisation of LMP in that they allow non-deterministic probabilistic transitions: several probabilistic transitions with the same label can leave a state.

**Definition 14 (Probabilistic Automata [18]).** A probabilistic automaton is a tuple $(S, s_0, L, D)$, where $S$ is a set of states, $L$ is a set of actions, $s_0$ is a particular state called the initial state, and $D \subseteq S \times L \times Disc(S)$ is the transition relation, where $Disc(S)$ is the set of probabilistic distributions on $S$.

We write $s \xrightarrow{a} \mu$ instead of $(s, a, \mu)$ for a transition in $D$. We require that the system be finitely branching: from every state $s \in S$ there exists a finite number of transitions $s \xrightarrow{a} \mu$, with $a \in L$. Informally, the outgoing transitions $s \xrightarrow{a} \mu$ represent the non-deterministic alternatives in the state $s$. A transition $s \xrightarrow{a} \mu$ asserts that in state $s$ the system can perform an action $a$ that will lead with probability $\mu(t)$ to a state $t \in S$.

No practical distance has been constructed yet on the state space of such models. The distances presented in [6] and [10] are highly non-constructive, and the ones defined in [9, 13, 21], apply only on purely probabilistic systems. We will briefly show how our distance construction for LMPs can be generalised in that case as well. We first point out the small changes that have to be done to the structural definitions of $\epsilon$-simulation and $\epsilon$-bisimulation.

**Definition 15 (\(\epsilon\)-simulation on a probabilistic automata).** If $A$ is a probabilistic automata and $R \subseteq S \times S$ is a relation, then $R$ will be a relation of $\epsilon$-simulation if:

$$\forall s, t \in S \ s R t \Rightarrow \forall s \xrightarrow{a} \mu, \text{there exists a transition } t \xrightarrow{a} \nu \text{ such that } \forall X \subseteq S \ we \ have \ \nu(R(X)) \geq \mu(X) - \epsilon.$$ 

$\epsilon$-bisimulation is the symmetric notion associated to $\epsilon$-simulation, as for LMPs.

We leave the logical characterisation of $\epsilon$-simulation and $\epsilon$-bisimulation for future work. We present a game characterisation of these notions. The game steps for $\epsilon$-simulation are now:

- Step 1: A chooses a transition $s \xrightarrow{a} \mu$ and a set $E \subseteq S$.
- Step 2: P chooses a transition $t \xrightarrow{a} \nu$ from $t$ with label $a$ and a set $F \subseteq S$ such that $\nu(F) \geq \mu(E) - \epsilon$.
- Step 3: A chooses $t' \in F$ from which P will have to play in the next sequence of steps.
- Step 4: P chooses $s' \in E$ from which A will have to play in the next sequence of steps.

This game is adapted for bisimulation in the same way as for LMPs. The proof of Theorem 5 can be modified to deal with non-determinism. The only subtle point in this translation is the fact that we use the “finite branching criterion” at the end of the proof.

As for LMPs, we have an algorithm which works in polynomial time in the size of the system to compute the relation $\sim_\epsilon$ on a probabilistic automaton $A$, given $\epsilon > 0$. Once more, we can, using a dichotomous approach, compute the distance $d(s, t)$ between two states of a probabilistic automaton $A$ up to precision $\delta \in [0; 1]$ in time $O(|S|^2 \cdot \ln(\delta))$.

8 Conclusion

The main contributions of this paper are relaxed notions of bisimulation and simulation that have nice characterisations using logics, games and structural definitions. The notion of $\epsilon$-bisimulation also gives rise to a new notion of an efficiently computable distance between systems that permits to measure differences for a long horizon without accumulating differences. As we explained, we consider that the latter property is key in the probabilistic framework where models involve probabilities are obtained from estimations.
or experiments. Moreover, a long horizon is preferable for reactive systems that are expected to last. The fact that our distance is computable in polynomial time is an important achievement of this paper, since the other known long horizon distance, $d^*$, has a double exponential complexity [20].

We prove these results in the context of deterministic probabilistic systems, but we also show how they can be extended to handle non-determinism. Future work will provide a logical characterisation in this setting.

We believe that something interesting is hiding behind the fact that two-way $\epsilon$-simulation is not $\epsilon$-bisimulation for $\epsilon > 0$, contrarily to the situation when $\epsilon = 0$. Here again, as in Example 5, we observe the analysis at the bounds must be done carefully.

The point of view we introduced may allow to build more efficient states-clustering techniques for large state-space systems. Defining $\epsilon$-simulation relations on systems with continuous state spaces is also an issue; of course, it would be theoretically interesting, but it could also simplify the study of such systems, using a quotient machinery on the state space with the relation $\sim\epsilon$. As we have seen, studying global or long term properties through an approximate logic is worth full. A complement for this approach, to tackle its complexity, would be to generalise parity games, whose resolution is equivalent to $\mu$-calculus model-checking [12], to our context and to our fixpoint logic.

References


A The relation $\prec_\epsilon$ is the greatest fixpoint of $F$

**Definition 16** (The relation $R_\epsilon$). $R_\epsilon$ is the relation defined on $S \times S$ by $R_\epsilon = \bigcap_{n \geq 0} R_{\epsilon,n}$, where the relations $R_{\epsilon,n}$ are defined inductively on $n$ as follows:

- $R_{\epsilon,0} = S \times S$.
- $s R_{\epsilon,n+1} t$ iff for all $a \in L$ and all set $E \subseteq S$ we have:
  - $h_\alpha(t, R_{\epsilon,n}(E)) \geq h_\alpha(s, E) - \epsilon$

**Lemma 1.** $R_\epsilon$ is a relation of $\epsilon$-simulation.

**Proof.**

- Let $s, t \in S$ be such that $s R_\epsilon t$. Let $a \in L$, and $X \subseteq S$. We have to show that $h_\alpha(t, R_\epsilon(X)) \geq h_\alpha(s, X) - \epsilon$.

$s R_\epsilon t$ means that for all $n \in \mathbb{N}$ we have $s R_{\epsilon,n} t$. Hence by an easy induction on $n$ we can see that for all $n \geq 0$, $h_\alpha(t, R_{\epsilon,n}(X)) \geq h_\alpha(s, X) - \epsilon$.

Next we see by induction that the $R_{\epsilon,n}$ form a decreasing sequence of relations. Thus the $R_{\epsilon,n}(X)$ form a decreasing sequence of sets. By continuity of $h_\alpha$, we have $h_\alpha(t, R_\epsilon(X)) = h_\alpha(t, \bigcap_{n \geq 0} R_{\epsilon,n}(X)) = \inf_{n \geq 0} h_\alpha(t, R_{\epsilon,n}(X))$, hence $h_\alpha(s, R_\epsilon(X)) \geq h_\alpha(s, X) - \epsilon$.

We can prove now the result of this section:

**Lemma 2.** $R_\epsilon = \epsilon$.

**Proof.** We just have to show that if $R$ is a relation of $\epsilon$-simulation of $S$, then $R \subseteq R_\epsilon$. For this we fix $R$ a relation of $\epsilon$-simulation on $S$ and we show by induction on $n \in \mathbb{N}$ that for all $n \in \mathbb{N}$, $R \subseteq R_{\epsilon,n}$.

- The case $n = 0$ is clear since $R_{\epsilon,0} = S \times S$.
- Let $n \in \mathbb{N}$ and suppose $R \subseteq R_{\epsilon,n}$. Let $s, t \in S$ such that $s R t$. Let $X \subseteq S$, and $a \in L$. Then $h_\alpha(t, R(X)) \geq h_\alpha(s, X) - \epsilon$. But by induction $R(X) \subseteq R_{\epsilon,n}(X)$, hence $h_\alpha(t, R_{\epsilon,n}(X)) \geq h_\alpha(s, X) - \epsilon$. This proves $s R_{\epsilon,n+1} t$, i.e. $R \subseteq R_{\epsilon,n+1}$.

B Proof of Theorem 3

**Theorem 3:** Let $s, t$ be two states of an LMP $S$ with countable state space. Then $s \prec_\epsilon t$ iff $s \prec_\epsilon t$.

**Proof.** $\Leftarrow$. Let $R$ be a relation of $\epsilon$-simulation. We prove by structural induction on formulas that for all $\phi \in L_\epsilon$, $R(\{\phi\}) \subseteq \{\phi\}$, which will prove the result. The base case is trivial, and since $R(\cdot)$ distributes over unions, the case of disjunction is also trivial. For conjunction, the result comes from the fact that for any two sets $R(A \cap B) \subseteq R(A) \cap R(B)$ and from the following: for any two formulas $\theta_1$ and $\theta_2$, $[\theta_1 \land \theta_2]_\epsilon = [\theta_1]_\epsilon \cap [\theta_2]_\epsilon$. Let use assume the claim for $\phi$ and prove it for $\langle a, q \rangle \phi$. Let $t \in R(\{\langle a, q \rangle \phi\})$. Then there is some $s \models \langle a, q \rangle \phi$ such that $s R t$. Since $R$ is a simulation, and letting $X = \{\phi\}$ there is some $Y \subseteq R(\{\phi\})$ such that $h_\alpha(t, Y) > h_\alpha(s, \{\phi\}) - \epsilon$. By induction hypothesis, $R(\{\phi\}) \subseteq \{\phi\}$, and we obtain $h_\alpha(t, \{\phi\}) > h_\alpha(s, \{\phi\}) - \epsilon$. This implies that $t \models \langle a, q \rangle \phi$, as wanted.

$\Rightarrow$. For the other direction we prove that $s \prec_\epsilon t$. Let $a \in L$, and $E \subseteq S$. We want to show that:

$$h_\alpha(t, L_\epsilon(E)) \geq h_\alpha(s, E) - \epsilon.$$  

Where $L_\epsilon(E) := \{y \in S | \exists e \in E | e \prec_\epsilon y\}$.

Let $F = L_\epsilon(E)$ and $\lambda < h_\alpha(s, E)$. Since $S$ is countable, there exists $E' \subseteq E$ finite such that $h_\alpha(s, E') > \lambda$. Note that $E' \subseteq L_\epsilon(E)$, and $E' = \cup e \in E' \cap e \models \psi \{\psi\}$. Indeed, it is easy to check that $\cap_{\psi \models \psi} \psi$ is exactly the state sets that $L_\epsilon$-simulate $e'$. Clearly we can consider that the parameters in the formulas of $L$ are taken in $Q$, and hence we can enumerate the formulas of $L_\epsilon$ as $(\psi_i)_{i \in \mathbb{N}}$.

Since $s \prec_\epsilon t$, it implies that for all formula $\psi \in L_\epsilon$, $h_\alpha(t, \{\psi\}) > h_\alpha(s, \{\psi\}) - \epsilon$. In particular, we have for all $k \in \mathbb{N}$ we have $h_\alpha(t, \{\cup e \in E \cap e \models \psi_i \wedge E, e < \psi_i\}) > h_\alpha(s, \{\cup e \in E' \cap e \models \psi_i \wedge E, e < \psi_i\}) - \epsilon$. By continuity of $h_\alpha$, we have $h_\alpha(s, \{\cup e \in E' \cap e \models \psi_i \wedge E, e < \psi_i\}) - \epsilon$. Since $\lambda$ was arbitrary, we indeed have that $h_\alpha(t, E) \geq h_\alpha(s, E) - \epsilon$.

We can now prove that $h_\alpha(t, F) \geq \lambda - \epsilon$.

$$h_\alpha(t, F) = h_\alpha(t, L_\epsilon(E)) \geq h_\alpha(t, L_\epsilon(E')) \geq h_\alpha(s, \{\cup e \in E' \cap e \models \psi_i \wedge E, e < \psi_i\}) - \epsilon \geq h_\alpha(s, E') - \epsilon \geq \lambda - \epsilon$$

Since $\epsilon$ was arbitrary, we indeed have that $h_\alpha(t, F) \geq h_\alpha(s, E) - \epsilon$.

The preceding proof uses crucially the countability of the state space. An extension to uncountable state spaces could have to go through non trivial domain theory machinery as was done in [8].

C Proof of Theorem 4

We prove the following lemma which is used in the proof of theorem 4.

**Lemma 3.** $\prec_\epsilon$ is a relation of $\epsilon$-simulation.
Proof. This is a mere reformulation of the game rules. Let $s, t \in S$ be such that $s \prec_{\mathcal{E}}^L t$. Let $E \subseteq S$ and $a \in L$. We have to show that there exists a set $F \subseteq \mathcal{E}(E)$ such that $h_a(t, F) \geq h_a(s, E) - \epsilon$. But $P$ has a winning strategy. Hence $P$ will be able to find $s \prec_{\mathcal{E}}^L t$ such that he has a winning strategy for the game $S, s', t'$. This means that $P$ can find a set $F \subseteq E$ such that $h_a(t, F) \geq h_a(s, E) - \epsilon$ and such that for any choice $t'$ of $A$ makes in Step 3, $P$ will be able to find $s' \in E$ such that $s' \prec_{\mathcal{E}}^L t'$, i.e. $F \subseteq \mathcal{E}(E)$. Hence such a set $F$ is convenient for the fact that $\prec_{\mathcal{E}}^L$ is a relation of $\epsilon$-simulation. \qed

D Lemma for Theorem 6

Lemma 4. $\sim_{\mathcal{E}}$ is a symmetric relation.

Proof. Of course this will come from the presence of the negation in the logic. Let $s, t \in S$, such that $s \sim_{\mathcal{E}} t$. Suppose $t \models \theta$. We show that $s \models \theta$, i.e. that $s \models \theta $. If not, $s \models \neg \theta $. Hence since $s \sim_{\mathcal{E}} t$ we would have $t \models \neg \theta $, $s \models \neg \theta $, i.e. $t \models \neg \theta \rightarrow \neg \theta $, which is a contradiction. \qed

E Network theory. Proof of Theorem 7

In order to prove Theorem 7 we need to introduce some concepts of flow network theory.

A network is a tuple $N = (V,E,\bot,T,\epsilon)$ where $(V,E)$ is a finite directed graph in which every edge $(u,v) \in E$ has a non-negative, real-valued capacity $c(u,v)$. If $(u,v) \in E$ we assume $c(u,v) = 0$. We distinguish two vertices: a source $\bot$ and a sink $T$.

For $v \in V$ let $in(v)$ be the set of incoming edges to node $v$, and $out(v)$ the set of outgoing edges from node $v$.

A flow function is a real function $f : V \times V \rightarrow \mathbb{R}$ with the two following properties for all nodes $u$ and $v$:

- Capacity constraints: $0 \leq f(u,v) \leq c(u,v)$. The flow along an edge cannot exceed its capacity.

- Flow conservation: for each node $v \in V - \{\bot,T\}$, $\sum_{e \in in(v)} f(e) = \sum_{e \in out(v)} f(e)$

The flow $F(f)$ of $f$ is given by

$$F(f) = \sum_{e \in out(\bot)} f(e) - \sum_{e \in in(T)} f(e).$$

A cut of a network is a split of the nodes in $V$ into two sets $S$ and $T$, such that $\bot \in S$ and $T \in T$. The capacity of a cut $(S,T)$ is defined as:

$$c(S,T) = \sum_{u \in S, t \in T | (s,t) \in E} c(s,t).$$

We begin by a presentation of a flow network that has been used in the past few years to compute efficiently several simulation relations on the state space of a probabilistic system. (cf Baier, Hermanns, Kwiatkowska...). Our notion of simulation is formulated in a different way than theirs. We speak about sets and "right-closed" sets, whereas they use the notion of weight functions between two distributions. As a consequence our characterisation of simulation in terms of maximum flow in a network is more tricky.

We present here the flow network introduced by Baier [2], and show that our notion of simulation can fit in that context and generalise her notion of simulation.

Definition 17 (The network $N(\mu,\nu,R)$). Let $S$ be a finite set, $R$ a subset of $\mathcal{R} \times S$, and $\mu, \nu$ two distributions on $S$. Let $S' = \{t' | t \in S\}$, where $t'$ are pairwise distinct "new" states (i.e. $t' \notin S$). We choose new elements $\bot$ and $T$ not contained in $S \cup S'$, $\bot \notin T$. Then the network $N(\mu,\nu,R) = (V,E,\bot,T,\epsilon)$ is defined as follows:

- $V = S \cup S' \cup \{\bot,T\}$.

- $E = \{(s,t') | (s,t) \in R\} \cup \{((\bot,s) | s \in S\} \cup \{(t',T) | t \in S\}$.

- The capacity function $c$ is given by: $c(\bot,s) = \mu(s)$, $c(t',T) = \nu(t)$, and $c(s,t') = 1$ for all $s,t \in S$.

For the main theorem we need some notions about networks we present now. For the following we fix a finite set $S$, a relation $R$ on $S \times S$, two distribution $\mu, \nu$ on $S$, and the associated network $N = N(\mu,\nu,R)$.

We are interested in the following theorem, which generalises the result of [2].

Theorem 9. Let $S$ be a finite state space of size $k$, $R \subseteq S \times S$ a relation, and $\mu, \nu \in \text{Sub}(S)$. Then the following are equivalent:

1. The maximum flow in $N(\mu,\nu,R)$ is greater or equal than $\mu(S) - \epsilon$.

2. For all $E \subseteq S$ we have $\nu(R(E)) \geq \mu(E) - \epsilon$.

Proof.

1. $\Rightarrow$ 2. Suppose the maximum flow in $N(\mu,\nu,R)$ is greater or equal than $\mu(S) - \epsilon$, i.e. that $\mu(S) - \epsilon \leq \sum_{s \in S} f(\bot,s)$ (1). Let $E \subseteq S$. Let $f$ be a function of maximal flow in $N(\mu,\nu,R)$. We have:

$$\sum_{s \in S} f(\bot,s) \leq \mu(S - E) + \sum_{s \in E} f(\bot,s).$$

and

$$\sum_{s \in E} f(\bot,s) = \sum_{s \in E} \sum_{t \in R(E)} f(s,t') \leq \sum_{t \in R(E)} \nu(t').$$

12
F Expressiveness of the fixpoint logic $L^\mu$

We begin by a precise definition for the approximation of the fixpoint logic. As for $L$, to each formula $\phi \in L^\mu$ and $\epsilon \in [-1;1]$ we associate a new formula $\phi_\epsilon$. We can define directly the associated approximate semantics $\models_{\epsilon \subseteq S \times S}$ as follows: if $\phi \in L^\mu$ and $s \in S$, $s \models \phi$ iff $s \models \phi_\epsilon$.

Definition 18. Let $\epsilon \in [-1;1]$. To each formula $\phi \in L^\mu$ we associate a new formula $\phi_\epsilon$. The construction is defined by structural induction on the formulas of $L^\mu$ as follows:

- If $\phi \in L$, $\phi = \neg \psi$, $\phi = \langle a \rangle \psi$ with $\psi$ closed, $\phi = \psi_1 \wedge \psi_2$ or $\phi = \psi_1 \lor \psi_2$, $\phi_\epsilon$ is constructed the same way as for the logic $L$. (i.e. for instance $(\langle a \rangle \psi)_\epsilon = \langle a \rangle \psi_{\epsilon \epsilon}$).
- If $\phi = \langle a \rangle Z$, then $\phi_\epsilon = \phi$.
- If $\phi = \mu Z. \psi$ then $\phi_\epsilon = \mu Z. \psi_\epsilon$.
- If $\phi = \nu Z. \psi$ then $\phi_\epsilon = \nu Z. \psi_\epsilon$.

The important fact is that we do not relax the satisfiability condition before a fixpoint variable. This way, we impose that the paths generated when considering the satisfaction of a global or long term formula are all taken with probability $1$.

Now the proof of Proposition 3:

- The logic $L^\mu$ has the same differentiating power as $L$.
- $\sim \subseteq \sim_0 \subseteq \sim_\epsilon$, and the inclusions may be strict. That is $\sim_0^\mu$ has a differentiating power between $\sim$ and $\sim_\epsilon$.

Proof. The fact that $L$ and $L^\mu$ have the same differentiation power comes easily from the fact that the system we consider have finite state space. Hence the fixpoint sets may be computed in a finite number of Picard iterations. If we have a formula of $L^\mu$ which differentiate two states $s, t$, we can unroll the fixpoint calculation to get a formula of finite depth, i.e. a formula of $L$, which differentiates these two states.

- Clearly the last point implies $\sim \subseteq \sim_0^\mu$. And since clearly $\sim_0^\mu$, seen as a function mapping $\epsilon \geq 0$ to a subset of $S \times S$, is a growing function, we get $\sim \subseteq \sim_0^\mu$.

Finally, since $L$ is a subset of $L^\mu$, we get $\sim_0^\mu \subseteq \sim_\epsilon$. The strictness of the inclusions will follow from the examples we consider next.

G Algorithm for computing $\sim_\epsilon$ in polynomial time

Algorithm 2. Input: A finite state LMP $S = (S, L, h)$, $\epsilon \in [0;1]$. 

\[ \sum_{s \in S} f(T, s) \leq \mu(S) - \epsilon \leq \sum_{t \in R(E)} \nu(t'). \]

With (1), we get indeed $\mu(E) - \epsilon \leq \nu(R(E))$.

$2 \Rightarrow 1$. We use the powerful Max-flow Min-cut theorem to prove this way. The Max-flow Min-cut theorem shows that in our context, we just have to prove that the minimal cut on the network $N(\mu, \nu, R)$ is no less than $\mu(S) - \epsilon$. First a notation: we write $\{s_1, ..., s_n\}$ for the set $S$, and the nodes of the network are: $\{T, \bot\} \cup \{s_1, ..., s_n\} \cup \{s'_1, ..., s'_n\}$. If $(X, Y)$ is a cut on $N(\mu, \nu, R)$, we will say that $s, t \in S$ satisfy (1) for the cut $(X, Y)$ if:

\[ s \in X, t' \in Y, \text{ and } s \mathrel{R} t. \]
Theorem 8: the kernel of \( d \) is \( \sim \)

In this section \( S \) is a countable LMP. We give a proof of Theorem 8: For \( s, t \in S \), we have \( s \sim t \) if and only if \( d(s, t) = 0 \).

Proof. The fact that \( \sim \) is a subset of the kernel of \( d \) is clear.

We use the logical characterisation of \( \sim \): we show by induction on the structure of the formulas that for any \( \phi \in L \) the following are equivalent:

1. \( s \models \phi \Rightarrow t \models \phi \).
2. For all \( \epsilon > 0 \), \( s \models \phi \Leftrightarrow t \models \phi \).

By symmetry, this will give that for all \( \phi \in L \), \( s \models \phi \Leftrightarrow t \models \phi \) iff for all \( \epsilon > 0 \) \( s \models \phi \Rightarrow t \models \phi \) and \( t \models \phi \Rightarrow s \models \phi \). Hence, for all \( \phi \in L \), this will prove that:

\[
(\forall \epsilon > 0 \ s \sim \epsilon t) \Rightarrow (s \models \phi \Leftrightarrow t \models \phi)
\]

which implies \( s \sim t \) if \( d(s, t) = 0 \).

Now \( 1 \Rightarrow 2 \) is clear. We prove \( 2 \Rightarrow 1 \) by induction on the structure of the formulas:

- The case \( \phi = T \) is clear.
- The cases \( \phi = \neg \psi \), \( \phi = \psi_1 \land \psi_2 \), \( \phi = \psi_1 \lor \psi_2 \) are simple as well.
- The interesting case is this one: Suppose \( \phi = (n)_{\delta} \psi \).

Let \( \lambda < h_a(s, [\psi]) \). Suppose 2 and \( s \models \phi \). By hypothesis, for any \( \epsilon > 0 \) we have \( h_a(t, [\phi_1]) \geq \lambda - \epsilon \). By induction \( \bigcap_{n \geq 1} [\phi_1/n] = [\phi] \), hence by continuity of the operator \( h_a(t, -) \), we have \( h_a(t, \bigcap_{n \geq 1} [\phi_1/n]) = h_a(t, [\psi]) \geq \lambda \). Since this is true for any \( \lambda < h_a(s, [\psi]) \), this proves \( h_a(t, [\psi]) \geq h_a(s, [\psi]) \), hence \( t \models \phi \).

\( \square \)

Algorithm for computing \( \sim \) on a probabilistic automata in polynomial time

Algorithm 3. Input: A finite state probabilistic automata \( A = (S, L, h) \), \( \epsilon \in [0; 1] \).

Output: The relation \( \sim \) on \( S \times S \).

Method:

Let \( R_0 = S \times S \).

For \( j = 0 \) to \( |S|^2 \) do begin:

\[
R_{j+1} = R_j.
\]

For all \( (s, t) \in R_j \) do begin:

\[
R_j = R_j - \{s, t\}.
\]

end end

return \( R_{|S|^2} \).

Using the algorithm that computes the maximum flow of the networks \( \mathcal{N}(h_a(s, -), h_a(t, -), R_j) \) in time \( O(|S|^3) \), we get an algorithm which computes \( \sim \) on \( S \times S \) in time \( O(|S|^7 \cdot |L|) \).

I Algorithm for computing \( \sim \) on a probabilistic automata in polynomial time

Algorithm 3. Input: A finite state probabilistic automata \( A = (S, L, h) \), \( \epsilon \in [0; 1] \).

Output: The relation \( \sim \) on \( S \times S \).

Method:

Let \( R_0 = S \times S \).

For \( j = 0 \) to \( |S|^2 \) do begin:

\[
R_{j+1} = R_j.
\]

For all \( (s, t) \in R_j \) do begin:

\[
R_j = R_j - \{s, t\}.
\]

end end

return \( R_{|S|^2} \).

Using the algorithm that computes the maximum flow of the networks \( \mathcal{N}(\mu, \nu, R_j) \) in time \( O(|S|^3) \), we get an algorithm which computes \( \sim \) on \( S \times S \) in time \( O(|S|^7 \cdot |L| \cdot m^2) \), where \( m = \max \text{Card} \{s \mapsto \mu \in D | s \in S, \alpha \in L\} \).

Usually we have that \( m \) is a constant independent of \( n \).