UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE MATEMÁTICA, ASTRONOMÍA Y FÍSICA

SERIE “A”

TRABAJOS DE INFORMÁTICA

Nº 2/09

Partial Order Reduction for Probabilistic Systems Assuming Distributed Schedulers

Sergio Giro - Pedro R. D’Argenio

Editores: Pedro R. D’argenio – Gabriel Infante López

CIUDAD UNIVERSITARIA – 5000 CÓRDOBA

REPÚBLICA ARGENTINA

Partial Order Reduction for probabilistic systems assuming distributed schedulers

Sergio Giro and Pedro R. D'Argenio

FaMAF, Universidad Nacional de Córdoba - CONICET
Ciudad Universitaria - 5000 Córdoba - Argentina

Abstract. In the verification of probabilistic systems, distributed schedulers are used to obtain tight bounds on worst-case probabilities, these bounds being more realistic than the ones obtained by considering unrestricted full-history dependent schedulers. In this paper, we define two classes of distributed schedulers. We present undecidability results related to the automatic verification under these classes of schedulers. In previous literature, we have proven that the model checking problem is undecidable for distributed schedulers. However, in this paper we show that, by assuming that the schedulers are in a given class, the technique of partial order reduction (POR) for LTL properties can be applied in a more efficient way than usual, thus yielding a system with less states and transitions than if reduced assuming unrestricted schedulers. The reduced system can then be analysed using well-known algorithms for full-history dependent schedulers. Our partial order reduction technique may also obtain bounds strictly tighter than the ones obtained by considering unrestricted schedulers (of course, such bounds are safe with respect to the class of schedulers under consideration). We explain that the two variants we present are obtained from a general theorem, thus raising the question of whether there are other "natural" classes of schedulers for which POR variants can be developed.

1 Introduction

Markov decision processes (MDPs) are widely used in diverse fields ranging from ecology to computer science. They are useful to model and analyse systems in which both probabilistic and nondeterministic choices interact. Particularly, composition oriented versions of MDPs like probabilistic automata [19] or probabilistic modules [12] are specially suitable to model concurrent systems such as distributed systems. MDPs can be automatically analysed using quantitative model checkers such as PRISM [17] or LiQuor [8].

Since nondeterminism is involved, analysis techniques for MDPs require to consider the resolution of all nondeterministic choices in order to obtain the desired result. The resolution of such nondeterminism is given by the so called *schedulers* (called also adversaries or policies, see e.g. [4, 19]). So, a quantitative model checker can be used to find out the best/worst probability value of

email: {sgiro,dargenio}@famaf.unc.edu.ar

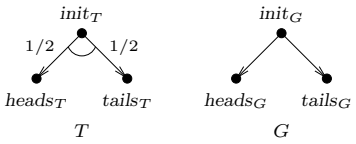


Fig. 1. T tosses a coin and G has to guess

reaching a goal under any possible scheduler (a concrete instance being "the probability of arrival of a package is above the bound 0.95").

Partial order reduction (POR, [16, 9]) is a well-known technique to cope with the state explosion problem. The works [1, 10] introduce partial order reduction for model checking of LTL properties on MDPs. In this paper, we explain that POR for probabilistic systems can be improved in case the schedulers are assumed to be *distributed*. Such assumption has been proven to be useful in previous literature, since it allows to calculate more realistic bounds on worst-case probabilities and it allows some restricted forms of compositional reasoning [11, 12, 7]. Our variants of POR can be used to reduce the state space more efficiently than according to [1, 10]. In addition, for some systems, our variants allow to calculate more realistic bounds on worst-case probabilities.

In the traditional MDP setting, in which the schedulers are *not* assumed to be distributed, a scheduler is a function mapping paths to transitions (or, in the more general case, paths to distributions on transitions). Given that the execution up to some state s is known (namely, the history path), the scheduler "chooses" to perform one transition out of all transitions enabled in state s. Quantitative model checkers such as [17, 8] are based on the technique introduced in [4], in which calculations are performed considering the set of *all* possible schedulers. However, considering all the schedulers may yield unexpected results.

The following example illustrates distributed schedulers, and also shows why it is convenient to consider only these schedulers when calculating worst-case probabilities. A man tosses a coin and another one has to guess heads or tails. Fig. 1 depicts the models of these men in terms of MDPs. Man T, who tosses the coin, has only one transition which represents the toss of the coin: with probability $\frac{1}{2}$ he moves to state $heads_T$ and with probability $\frac{1}{2}$ he moves to state $tails_T$. Instead, man G has two possible transitions, each one representing his choice: $heads_G$ or $tails_G$. An *almighty* scheduler for this system may let G guess the correct answer with probability 1 according to the following sequence: first, it lets T toss the coin, and then it chooses for G the transition leading to heads if T tossed a head or the transition leading to tails if T tossed a tail. Therefore, the maximum probability of guessing obtained by quantifying over these almighty schedulers is 1, even if T is a smart player that always hides the outcome until G reveals his choice. So, quantitative model checkers based on [4], though safe, yield an overestimation of the correct value. In this example, in which T and G do not share all information, we would like the maximum probability of guessing (i.e., of reaching any of the states $(heads_T, heads_G)$ or $(tails_T, tails_G)$) to be $\frac{1}{2}$.

This observation is fundamental in distributed systems in which components share little information with each other, as well as in anonymity protocols, where

the possibility of information hiding is a fundamental assumption (a well-known example of these systems being the *dining cryptographers problem* [6]). The phenomenon we illustrated has been first observed in [19] from the point of view of compositionality, and has been studied later in other settings [11, 12, 7].

In order to avoid considering the unrealistic behaviours we illustrated, *distributed* schedulers were proposed in previous literature. *Local* schedulers for each component of the system are defined in the usual way (that is, the choices are based on the complete history of the component) and the distributed schedulers are defined to be the schedulers that can be obtained by composing these local schedulers. Distributed schedulers have been studied in [12] in a synchronous setting and in [7] in an asynchronous setting. In addition, distributed schedulers are related to the partial-information policies of [11].

We remark that the "almighty" scheduler of the example would not be a valid scheduler in this new setting since the choice for G depends only on information which is external to (and not observable by) G. Then, a local scheduler for G takes the decision having no information about the actual state of T, and so the choice cannot be changed according to the outcome of T.

The first work to focus on model checking techniques when schedulers are restricted to be distributed is [13]. Unfortunately, in [13] we showed that the bounds for the probability of properties cannot be calculated nor even approximated if the schedulers are restricted to be distributed. Actually, the framework in [13] is the same synchronous setting of [12].

Since we focus on distributed systems, in this paper we work on an asynchronous setting based on [7] (however, the order in which the components execute is not restricted by the token structure in [7], and so our interleaving mechanism is more general). In Sec. 2 we present this setting and the first version of schedulers, that we call *distributed schedulers*. These schedulers focus on preserving the behaviour of a component hidden from other components (unless, of course, the component reveals it), thus ruling out the "almighty" schedulers as the one explained before. We will show later that this class of schedulers may still peek at some hidden information producing an undesired global result. Therefore, in Sec. 3 we go further restricting this set to what we call *strongly distributed schedulers*. The undecidability of the model checking problem for distributed schedulers is proven in [13]. Here, we show that the restriction we impose to strongly distributed schedulers also leads to undecidability. In addition, we present new undecidability results related to qualitative properties for both distributed and strongly distributed schedulers.

In Sec. 4 we present two variants of POR, corresponding to distributed and strongly distributed schedulers, respectively. Although we use Probabilistic I/O Automata as our underlying theoretical framework, one of the variants of POR we present (the more powerful one, corresponding to strongly distributed schedulers) can be applied directly to MDP models without input/output restrictions as the PRISM language [17]. With respect to the other variant, we show how can it be applied in case the model has no input/output distinctions. Finally, in Sec. 5 we explain that the two variants we present are obtained from a general

A generative structure with two transitions A reactive structure with two transitions

Fig. 2. Reactive and generative structures

theorem, thus leaving open the question of whether there are other "natural" classes of schedulers for which POR variants can be developed.

2 Interleaved Probabilistic Input/Output Automata

We present a framework based on the Switched PIOA [7]. It uses reactive and generative structures (see [15]). For a finite set S, we denote by $\mathsf{DiscDist}(S)$ the set of all discrete probability distributions over the set S. Given a set ActLab of action labels and a set S of states, the set of generative transitions T_G on $(\mathsf{S}, \mathsf{ActLab})$ is $\mathsf{DiscDist}(\mathsf{S} \times \mathsf{ActLab})$, and the set T_R of reactive transitions is $\mathsf{DiscDist}(\mathsf{S})$. A generative structure on $(\mathsf{S}, \mathsf{ActLab})$ is a function $G : \mathsf{S} \to \mathcal{P}(T_G)$ and a reactive structure on $(\mathsf{S}, \mathsf{ActLab})$ is a function $R : \mathsf{S} \times \mathsf{ActLab} \to \mathcal{P}(T_R)$. Figure 2 depicts an example of these structures. Generative transitions model both communication and state change. The component executing a generative transition chooses both a label a to output (the ! in the figure represents output) and a new state s according to a given distribution. Reactive transitions specify how a component reacts to a given input (the ? in the figure represents input). Since the input is not chosen, reactive transitions are simply distributions on states.

In our framework, a system is obtained by composing several *probabilistic I/O atoms*. Each atom is a probabilistic automata having reactive and generative transitions.

Definition 1. *A probabilistic I/O atom is a 5-tuple* $(\mathsf{S}, \mathsf{ActLab}, G, R, \mathsf{init})$*, where* S *is a finite set of states,* ActLab *is a finite set of actions labels, and* G *(*R*, resp.) is a generative (reactive, resp.) structure in* $(\mathsf{S}, \mathsf{ActLab})$*.* $\mathsf{init} \in \mathsf{S}$ *is the initial state. We require the atoms to be input-enabled, so* $R(s, a) \neq \emptyset$ *for every* $s \in \mathsf{S}$*,* $a \in \mathsf{ActLab}$*. We often write* S_i *to denote the set of states of an atom* A_i *and similarly for the other elements of the 5-tuple. In addition, we write* T_{G_i} *(*T_{R_i}*, resp.) for the set of generative (reactive, resp.) transitions on* $(\mathsf{S}_i, \mathsf{ActLab}_i)$*.*

An interleaved probabilistic I/O system P is a set $\mathrm{Atoms}(P)$ of probabilistic I/O atoms $A_1, \cdots, A_N$. The set of states of the system is $\prod_i \mathsf{S}_i$, and the initial state of the system is $\mathsf{init} = (\mathsf{init}_1, \cdots, \mathsf{init}_N)$.

In order to define how the system evolves, we define *compound transitions*, which are the transitions performed by the system as a whole. In such compound transitions, all the atoms having the same action label in their alphabet must synchronize and *exactly one* of them must participate with an output (generative) transition (thus modelling multicasting). Formally, a compound transition is a tuple $(g_i, a, r_{j_1}, \cdots, r_{j_m})$ (we require $i \neq j_k$ and $j_k \neq j_{k'}$ for all $k \neq k'$) where

g_i is a generative transition in the atom A_i (the *active* atom), $a \in \mathsf{ActLab}_i$ is an action label, the r_{j_k} are reactive transitions in the atoms A_{j_k} (the *reactive* atoms) and $\{A_i, A_{j_1}, \cdots, A_{j_m}\}$ is the set of all the atoms such that $a \in \mathsf{ActLab}_j$. We say that A_i, A_{j_1}, ..., A_{j_m} are the atoms *involved* in the compound transition. A compound transition $(g_i, a, r_{j_1}, \cdots, r_{j_m})$ is enabled in a given state $(s_1, \cdots, s_N)$ if $g_i \in G_i(s_i)$ and $r_{j_k} \in R_{j_k}(s_{j_k}, a)$. The action label a of a compound transition c is indicated by $label(c)$. The probability $c(s, s')$ of reaching a state $s' = (s'_1, \cdots, s'_N)$ from a state $s = (s_1, \cdots, s_N)$ using a compound transition $c = (g_i, a, r_{j_1}, \cdots, r_{j_m})$ is $g_i(s'_i, a) \cdot \prod_{k=1}^{m} r_{j_k}(s'_{j_k})$ if $s_t = s'_t$ for every atom not involved in the transition. Otherwise, $c(s, s') = 0$.

In order to ease some definitions, we introduce a fictitious "stutter" compound transition ς. Intuitively, this transition is executed iff the system has reached a state in which no atom is able to generate a transition. The probability $\varsigma(s, s')$ of reaching s' from s using ς is 1, if $s = s'$, or 0, otherwise.

A path σ of P is a sequence of the form $s_1.c_1.s_2.c_2 \cdots c_{n-1}.s_n$ where each s_i is a (compound) state and each c_i is a compound transition. A path can be finite or infinite. For a finite path σ as before, the set $[\sigma]$ contains all the infinite paths starting with σ. We define $\mathrm{last}(\sigma) = s_n$ and $\mathrm{len}(\sigma) = n$.

In the following, we suppose that input-enabled atoms $A_1, \ldots, A_N$ are given, and we are considering the system P comprising all the atoms A_i. We call this system "the compound system". The states (paths, resp.) of the compound system are called global states (global paths, resp.) and the states (paths, resp.) of each atom are called local states (local paths, resp.).

The probability of a set of executions depends on how the nondeterminism is resolved. A scheduler transforms a nondeterministic choice into a probabilistic choice by assigning probabilities to the available transitions. Given a system and a scheduler, the probability of a set of executions is completely determined.

Usually, schedulers assign probabilities to the available transitions taking into account the complete history of the system. So, *arbitrary* schedulers are defined as functions mapping paths to distributions on transitions. As we have seen, it may be unrealistic to assume that the schedulers are able to see the full history of all the components in the system. In the following, we define restricted classes of schedulers in order to avoid considering unrealistic behaviours.

Distributed schedulers. In a distributed setting as the one we are introducing, different kinds of nondeterministic choices need to be resolved. An atom needs a corresponding *output* scheduler to choose the next generative transition. In addition, it may be the case that many reactive transitions are enabled for a single label in the same atom. So, for each atom we need an *input* scheduler in order to choose a reactive transition for each previous history and for each label. Output and input schedulers are able to make their decisions based only on the local history of the atom. So, we need the notion of *projection*.

Given a path σ, the projection $\sigma[i]$ of the path σ over an atom A_i is defined inductively as follows: **(1)** $(\mathsf{init}_1, \cdots, \mathsf{init}_N)[i] = \mathsf{init}_i$, **(2)** $\sigma.c.s[i] = \sigma[i]$ if A_i is not involved in c, and **(3)** $\sigma.c.s[i] = \sigma[i]\,.label(c).\pi_i(s)$ (where π_i is the usual

projection on tuples), otherwise. The set of all the projections of paths over an atom A_i is denoted by $\mathsf{Proj}_i(P)$.

An output scheduler for the atom A_i is a function $\Theta_i : \mathsf{Proj}_i(P) \to \mathsf{DiscDist}(T_{G_i})$ such that, if $G_i(\text{last}(\sigma)) \neq \emptyset$ then $\Theta_i(\sigma[i])(g) > 0 \implies g \in G_i(\text{last}(\sigma[i]))$. An input scheduler for an atom A_i is a function $\Upsilon_i : \mathsf{Proj}_i(P) \times \mathsf{ActLab}_i \to \mathsf{DiscDist}(T_{R_i})$ s.t. $\Upsilon_i(\sigma[i], a)(r) > 0 \implies r \in R_i(\text{last}(\sigma[i]), a)$. Note that, if the output scheduler Θ_i fixes a generative transition for a given local path $\sigma[i]$, then the actions in the generative transition can be executed in every global path σ' s.t. $\sigma'[i] = \sigma[i]$, since we require the atoms to be input-enabled.

An important modification with respect to the framework in [7] is the addition of an *interleaving scheduler* that chooses the next component to perform an output. We compare previous approaches to interleaving in Sec. 5.

An interleaving scheduler is a map that, for a given (global) history, (probabilistically) chooses an active atom that will be the next to execute an output transition (according to its output scheduler). Formally, an *interleaving scheduler* is a function $\mathcal{I} : \mathsf{Paths}(P) \to \mathsf{DiscDist}(\{1, \cdots, N\})$ such that, if there exists i such that $G_i(\text{last}(\sigma[i])) > 0$ (that is, if there is some atom being able to generate a transition) then $\mathcal{I}(\sigma)(i) > 0 \implies G_i(\text{last}(\sigma[i])) \neq \emptyset$. Note that, even if interleaving schedulers are unrestricted, compound schedulers for the compound system are still restricted, since the local schedulers can only see the portion of the history corresponding to the component.

A scheduler for the compound system results by the appropriate composition of the interleaving scheduler and the output and input schedulers of each atom.

Given an interleaving scheduler $\mathcal{I}$, input schedulers Υ_i and output schedulers Θ_i for each atom i, the distributed scheduler η obtained by composing $\mathcal{I}$, Θ_i and Υ_i is defined as:

$$\eta(\sigma)(g_i, a, r_{j_1}, \cdots, r_{j_m}) = \mathcal{I}(\sigma)(i) \cdot \Theta_i(\sigma[i])(g_i) \cdot \prod_{k=1}^{m} \Upsilon_{j_k}(\sigma[j_k], a)(r_{j_k}) \ ,$$

where A_{j_k} are all the atoms such that $a \in \mathsf{ActLab}_{j_k}$. The set of distributed schedulers of P is denoted by $\mathsf{Dist}(P)$.

Usually, schedulers are defined to map into distributions on transitions. However, it may be the case that $\sum_c \eta(\sigma)(c) > 1$ for a distributed scheduler η. This is because action labels are *not* chosen by the scheduler (they are chosen by the generative transition). However, for every label a, $\sum_{\{c | label(c)=a\}} \eta(\sigma)(c) = 1$.

The probability of the extension sets $[\sigma]$ is inductively defined as follows: the probability $\Pr^\eta([\mathsf{init}])$ of the extensions of the initial state is 1. If there is i s.t. $G_i(\text{last}(\sigma[i])) \neq \emptyset$, then the probability $\Pr^\eta([\sigma.c.s])$ is $\Pr^\eta(\sigma) \cdot \eta(\sigma)(c) \cdot c(\text{last}(\sigma), s)$. If there is no such i, then the system cannot generate any transition. In this case, we let $\Pr^\eta([\sigma.c.s]) = \Pr^\eta([\sigma])$ if $c = \varsigma$ and $s = \text{last}(\sigma)$, or 0 otherwise.

Note that, if $c = (g_i, a, r_{j_1}, \cdots, r_{j_m})$, then $\eta(\sigma)(c) \cdot c(s, s')$ is

$$\mathcal{I}(\sigma)(i) \cdot \Theta_i(\sigma[i])(g_i) \cdot \prod_{k=1}^{m} \Upsilon_{j_k}(\sigma[j_k], a)(r_{j_k}) \cdot g_i(s'_i, a) \cdot \prod_{k=1}^{m} r_{j_k}(s'_{j_k}) \ ,$$

which implies $\sum_{c,s'} \eta(\sigma)(c) \cdot c(\text{last}(\sigma), s') = 1$. This probability can be extended to the least σ-field containing all the extension sets in the standard way.

Undecidability for distributed schedulers. In [13] we have proven (in a similar setting, but the proof strategy applies as well) that the maximum probability that some of the states in a given set are reached cannot be calculated

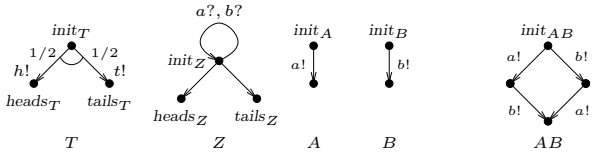


Fig. 3. Motivating strongly distributed schedulers

nor even approximated. A new result in this direction is that it cannot be decided whether or not there exists a scheduler reaching some of the states in a given set with probability 1. Note that this is a result of a more "qualitative" nature. The proof of this result differs from that of [13], and uses a reduction of the Post correspondence problem. For a formal statement and the proof of the qualitative result, see the Appendix.

3 Strongly distributed schedulers

Distributed schedulers model the fact that components can only look at their local history to choose the next transition to perform. However, under distributed schedulers, it is still possible that the hidden state of a component affects the behaviour of an unrelated group of components.

We explain how this leak of information occurs using atoms depicted in Fig. 3. Consider the system P having atoms T, Z, A, B. In this system, T is a process that tosses a coin. For the labels $h!$ and $t!$ corresponding to heads and tails, we have $h!, t! \notin \mathsf{ActLab}_Z \cup \mathsf{ActLab}_A \cup \mathsf{ActLab}_B$. So, according to this model, T keeps the outcome as a secret (coins whose output are assumed to be secrets can be found in probabilistic security protocols such as the solution to the dining cryptographers problem, see [6]). Atom Z models an attacker trying to guess the outcome of the coin. Atoms A and B are two process that Z is able to observe.

Consider the maximum probability that attacker Z guesses the outcome (i.e. the probability that a state of the form $(heads_T, heads_Z, \cdots)$ or $(tails_T, tails_Z, \cdots)$ is reached). Since the attacker is able to to see only the actions of A and B (and these atoms cannot, in turn, see the outcome of T) the attacker has no information about T, and so the maximum probability should be 1/2. Unfortunately, there exists a distributed scheduler that yields probability 1: the interleaving scheduler chooses T in the first place, and then it chooses either (A and then B) or (B and then A), according to the outcome of the probabilistic transition. Finally, the interleaving scheduler chooses A. The order in which $a!$ and $b!$ were output is part of the local history of Z, so the output scheduler for Z can always choose the transition agreeing with the outcome of the coin.

Note that the leak of information arises from the fact that the interleaving scheduler can look at the complete history of the system. In the following we derive restrictions on interleaving schedulers that prevent the leak presented above. Then, *strongly distributed schedulers* are defined as distributed schedulers whose interleaving scheduler complies with such condition.

In the example above, the state of T affects the execution of atoms A and B. Distributed schedulers were defined in such a way that the state of an atom

cannot affect the execution of another atom. Note that, if we regard A and B as a single component AB, we end up in a situation very similar to the one depicted in Fig. 1: in the case in which the coin lands heads AB chooses to perform the transition $a!$, while in the other case it chooses to perform the transition $b!$. In fact, if we consider the system P' such that $\mathrm{Atoms}(P') = \{T, Z, AB\}$, no output scheduler for AB can be defined in such a way that the order of execution of $a!$ and $b!$ depends on the outcome of T (since the outcome of T does not affect the state of AB). Then, there is no distributed scheduler for P' that can simulate the behaviour in P explained in the previous paragraph. Therefore, we would like that the new scheduler works just like distributed schedulers would do when A and B are considered as a single atom.

Let P be a compound system containing atoms A and B. Let AB be a single atom representing the composition of A and B and P' another compound system such that $\mathrm{Atoms}(P') = \big(\,\mathrm{Atoms}(P) \setminus \{A, B\}\,\big) \cup \{AB\}$. In general, we want to restrict interleaving schedulers such that, for every distributed scheduler η on P complying to such restriction, there is a distributed scheduler η' on P' that defines the same probabilistic behaviour.

To motivate the restriction, consider a scheduler for the system P with T, A and B in Fig. 3. Consider a distributed scheduler η whose interleaving scheduler complies $\mathcal{I}(\mathsf{init}) = (\frac{1}{2}T + \frac{2}{6}A + \frac{1}{6}B)$. We seek for a restriction on $\mathcal{I}$ s.t. it is possible to find a distributed scheduler for P' containing atoms T and AB in Fig. 3. When AB is in state $(\mathsf{init}_A, \mathsf{init}_B)$, the output scheduler Θ_{AB} chooses a distribution on $\{a!, b!\}$. To respect the choice of $\mathcal{I}$ in P, it must hold that $\Theta_{AB}(\mathsf{init}_{AB})(a!) = 2 \cdot \Theta_{AB}(\mathsf{init}_{AB})(b!)$, since, according to $\mathcal{I}$, the probability of executing $a!$ is twice the probability of executing $b!$. So,

$$\Theta_{AB}(\mathsf{init}_{AB})(a!) = \tfrac{2}{3} \quad \text{and} \quad \Theta_{AB}(\mathsf{init}_{AB})(b!) = \tfrac{1}{3}\,. \tag{1}$$

Suppose $(\mathsf{init}_T, \mathsf{init}_A, \mathsf{init}_B) \xrightarrow{t!} (\mathit{heads}_T, \mathsf{init}_A, \mathsf{init}_B)$ in P. The corresponding path in P' is $(\mathsf{init}_T, \mathsf{init}_{AB}) \xrightarrow{t!} (\mathit{heads}_T, \mathsf{init}_{AB})$. Call both these paths σ_{heads} (ambiguity is resolved according to whether it is used in the context of P or P').

Since $\sigma_{\mathit{heads}}[AB] = \mathsf{init}_{AB} = (\mathsf{init}_T, \mathsf{init}_{AB})[AB]$, we have that $\Theta_{AB}((\mathsf{init}_T, \mathsf{init}_{AB})[AB])(a!) = \Theta_{AB}(\sigma_{\mathit{heads}}[AB])(a!) = \Theta_{AB}(\mathsf{init}_{AB})(a!) = \frac{2}{3}$ and similarly for $b!$. Therefore $\Theta_{AB}(\sigma_{\mathit{heads}}[AB])(a!) = 2\Theta_{AB}(\sigma_{\mathit{heads}}[AB])(b!)$. This relation has to be maintained in P by $\mathcal{I}(\sigma_{\mathit{heads}})$. That is, whichever is the probabilistic choice in $\mathcal{I}(\sigma_{\mathit{heads}})$ w.r.t. other atoms, the relation $\mathcal{I}(\sigma_{\mathit{heads}})(a!) = 2 \cdot \mathcal{I}(\sigma_{\mathit{heads}})(b!)$ has to be maintained.

This suggests that, in the general case, for two executions that cannot be distinguished by any of two atoms A and B, the *relative probabilities* of choosing A over B (or B over A) should be the same. Or better stated: conditioned to the fact that the choice is between atoms A and B, the probability should be the same in two executions that cannot be distinguished by any of the two atoms.

Formally, given any two atoms A, B of a system P, for all σ, σ' s.t. $\sigma[A] = \sigma'[A]$ and $\sigma[B] = \sigma'[B]$, it must hold that

$$\frac{\mathcal{I}(\sigma)(A)}{\mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B)} = \frac{\mathcal{I}(\sigma')(A)}{\mathcal{I}(\sigma')(A) + \mathcal{I}(\sigma')(B)} \tag{2}$$

provided that $\mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B) \neq 0$ and $\mathcal{I}(\sigma')(A) + \mathcal{I}(\sigma')(B) \neq 0$.

Definition 2. *A scheduler η is* strongly distributed *iff η is distributed and equation (2) holds on the interleaving scheduler $\mathcal{I}$ that defines η. The set of strongly distributed schedulers of P is denoted by* $\mathsf{SDist}(P)$.

We emphasize that strongly distributed schedulers are useful depending on the particular model under consideration. In case we are analysing an agreement protocol and each atom models an independent node in a network, then the order in which nodes A and B execute cannot depend on information not available to none of them, and so strongly distributed schedulers give more realistic worst-case probabilities. However, in case the interleaving scheduler represents an entity that is able to look at the whole state of the atoms (for instance, if the atoms represent processes running on the same computer, and the interleaving scheduler plays the role of the kernel scheduler), then the restriction above may rule out valid behaviours, and so distributed schedulers should be considered.

The following theorem is the generalization of the fact that, for every strongly distributed scheduler η on $P = \{T, Z, A, B\}$ as in Fig. 3 there is a distributed scheduler η' on $P' = \{T, Z, AB\}$ that defines the same probabilistic behaviour.

Theorem 1. *Let P be a system such that $A, B \in \mathrm{Atoms}(P)$. Consider the system P' such that $\mathrm{Atoms}(P') = \big(\,\mathrm{Atoms}(P) \setminus \{A, B\}\,\big) \cup \{AB\}$, where AB is the usual cross-product of A and B (as in, for instance, [7, p. 99]). Then, for every strongly distributed scheduler η for P, there exists a strongly distributed scheduler η' for P' yielding the same probability distribution on paths as η.*

One may wonder what happens if, instead of considering two atoms A and B in (2), two *disjoint sets* $\mathcal{A}, \mathcal{B}$ of atoms are considered. The (apparently more general) condition on sets holds whenever condition (2) on atom holds (see the Appendix for a formal statement and proofs).

Undecidability for strongly distributed schedulers. We obtained undecidability results similar to the case of distributed schedulers. In addition, we proved that neither quantitative nor qualitative model checking can be performed *in case there is no nondeterministic choices internal to the components*. That is, the condition on strongly distributed schedulers suffices to introduce undecidability, regardless of the undecidability introduced by input/output schedulers. Proofs for the case of distributed schedulers must be adapted to these schedulers (see the Appendix).

4 Partial order reduction under distributed schedulers

In this section, we develop two variants of POR for probabilistic systems (each variant corresponding to a class of schedulers) using the *ample sets* in [9]. Such variants exploit the distributedness assumptions on schedulers in order to improve the reduction.

Our variants allow to construct an MDP that can be analysed using the well-known model checking algorithm in [4]. The results obtained in this analysis are safe bounds on the probability of $\mathsf{LTL}_{\neg\,\mathsf{next}}$ properties under the corresponding class. We also explain how our technique can be used in languages without input/output restrictions such as the PRISM language.

4.1 From Interleaved PIOA to MDPs

Since probabilistic model checkers are based on MDPs we explain our technique in this setting and formally interpret interleaved PIOAs in terms of MDPs.

Definition 3 (MDP). *An* MDP *is tuple* $M = (\mathsf{S}, \mathsf{Actions}, \mathsf{P}, \mathsf{init})$*, where* S *is a finite set of states,* $\mathsf{Actions}$ *is a finite set of actions identifiers,* $\mathsf{P} : (\mathsf{S} \times \mathsf{Actions} \times \mathsf{S}) \to [0,1]$ *is the (three-dimensional) probability matrix,* $\mathsf{init} \in \mathsf{S}$ *is the initial state.* $\mathsf{Actions}(s)$ *denotes the set of actions enabled in state* s*, i.e. the set of actions* α *such that* $P(s,\alpha,t) > 0$ *for some* $t \in \mathsf{S}$*. For every state* $s \in \mathsf{S}$*, we require that* $\mathsf{Actions}(s) \neq \emptyset$ *and* $\sum_{s' \in \mathsf{S}} \mathsf{P}(s,\alpha,s') = 1$ *for every action* $\alpha \in \mathsf{Actions}(s)$*. (In particular, we assume that* M *does not have terminal states.)*

Given an Interleaved PIOA we can construct an equivalent MDP as follows. Since MDPs have no concept of action labels, we encode the last action label as part of the state. So, the set of states of the MDP is $\mathsf{S} = (\mathsf{ActLab} \cup \{a_{\text{Init}}\}) \times \prod_i \mathsf{S}_i$, where a_{Init} is a fictitious label introduced because the initial state has no previous label. Each action in the MDP specifies a generative transition and, in addition, it specifies how the other atoms react to the generative transition. So, each element in $\mathsf{Actions}$ is of the form $(g_i, f_1, \cdots, f_{i-1}, f_{i+1}, \cdots, f_N)$, where g_i is a generative transition of the atom i and $f_j : \mathsf{ActLab}_j \to T_{R_j}$ (recall Def. 1). Each action of the form $(g_i, f_1, \cdots, f_{i-1}, f_{i+1}, \cdots, f_N)$ corresponds to several compound transitions: namely, there is one compound transition for each label in ActLab_i. Given an action as before and a label a, we obtain the compound transition $(g_i, a, f_{r_1}(a), \cdots, f_{r_k}(a))$, where $r_1, \cdots, r_k$ are the atoms that react to a. An action α in the MDP is enabled in s iff the corresponding compound transitions are enabled in the Interleaved PIOA. The probability matrix is defined as $\mathsf{P}((a,s),\alpha,(a',s')) = c(s,s')$, where $\alpha = (g_i, f_1, \cdots, f_{i-1}, f_{i+1}, \cdots, f_N)$ and $c = (g_i, a', f_{r_1}(a'), \cdots, f_{r_k}(a'))$. The initial state is $(a_{\text{Init}}, \mathsf{init})$, where init is the initial state of the Interleaved PIOA.

Schedulers for MDPs map paths to probability distributions on $\mathsf{Actions}$. The probability $\Pr^\eta(\sigma)$ of a path $\sigma = s_0 \xrightarrow{\alpha_1} s_1 \cdots \xrightarrow{\alpha_n} s_n$ is defined inductively: the probability of the initial state is 1. The probability of a path $\sigma \xrightarrow{\alpha_n} s_n$ is $\Pr^\eta(\sigma) \cdot \eta(\alpha_n) \cdot \mathsf{P}(\text{last}(\sigma), \alpha, s_n)$. The set of schedulers of M is denoted by $\mathsf{Sched}(M)$.

4.2 Partial order reduction for $\mathsf{LTL}_{\neg\,\mathbf{next}}$

Given a system under verification, the technique of partial order reduction yields another system with less transitions. The reduced system must respect certain restrictions so that it satisfies exactly the same set of properties as the original

system. When probabilities come into play, an extra restriction on the reduced system is needed. For the example in Fig. 1, the extra restriction prevents the reduction in which the probabilistic choice is eliminated from the initial state, because this reduction does not preserve the behaviour in which the nondeterministic choice is resolved according to the outcome of the probabilistic choice. However, our technique allows this reduction: although the described behaviour is not preserved, such behaviour corresponds to a scheduler that is not distributed. So, the variants of POR we present prevent more behaviours than the POR in [1, 10] does. From our correctness proof, it is inferred that all the eliminated behaviours are unrealistic behaviours corresponding to schedulers that do not comply with the distributedness assumption. So, if the POR eliminates unrealistic behaviours that are significant to the bounds to be obtained, the analysis of the reduced system yields more realistic results than the ones obtained by analysing the original system.

A description of the temporal logic LTL can be found in [9]. Given a set AP of atomic propositions and an LTL formula ϕ, the validity of ϕ depends on a function $\mathsf{L} : \mathsf{S} \to 2^{\mathsf{AP}}$, that indicates the set of atomic propositions that are valid in each state. We use Sat_ϕ to denote the set of all paths satisfying ϕ.

The set of transitions enabled at some state s in the reduced system is called the *ample set* of s and is denoted by $\mathsf{ample}(s)$. Restrictions to the ample sets are based on the notion of *independence*. In the following, $\mathsf{Inv}(\alpha)$ denotes the set of all the atoms that may be involved in the execution of α. If g_i is the generative transition in α, $\mathsf{Inv}(\alpha) = \{A \mid \exists a, s : g_i(s, a) > 0 \wedge a \in \mathsf{ActLab}_A\}$.

We say that two actions α, β are *independent* iff $(\exists s : \{\alpha, \beta\} \in \mathsf{Actions}(s)) \implies \mathsf{Inv}(\alpha) \cap \mathsf{Inv}(\beta) = \emptyset$.

So, two actions are independent only if the execution of one of them does not interfere with the execution of the other one. Note that the order of execution is irrelevant and that neither of them can disable the other. This definition of independence is widely used in practice, and, in fact, it is suggested as a practical criterion in [16, p. 28]. However, using the general theorem upon which Theorem 2 and Theorem 3 rely (see Sec. 5), we also obtain improvements using the definition of independence in [1, 10] (see the Appendix).

We need some additional definitions before presenting the restrictions for POR. A path $s_0 \xrightarrow{\alpha_0} s_1 \cdots$ is *possible* iff $\mathsf{P}(s_i, \alpha_i, s_{i+1}) > 0$ for all i. An action α is stutter iff $\mathsf{P}(s, \alpha, s') = 0$ for all s' such that $\mathsf{L}(s) \neq \mathsf{L}(s')$. An *end component* (EC) is a pair (T, A) where $A : T \to \mathcal{P}(\mathsf{Actions})$ and T is a set of states such that: **(1)** $\emptyset \neq A(s) \subseteq \mathsf{Actions}(s)$ for all $s \in T$, **(2)** $\mathsf{P}(s, \alpha, t) > 0$ implies $t \in T$, for all $s \in T$, $\alpha \in A$ **(3)** for every $s, t \in T$ there exists a possible path from s to t.

The restrictions for the ample sets to preserve $\mathsf{LTL}_{\neg\,\mathrm{next}}$ properties under unrestricted full-history dependent schedulers are listed below. $\hat{\mathsf{S}}$ denotes the set of reachable states in the reduced system $\hat{M}$, which is constructed by taking $\mathsf{ample}(s)$ to be the set of enabled actions in $s \in \hat{\mathsf{S}}$.

(**A1**) *For all states* $s \in \mathsf{S}$, $\emptyset \neq \mathsf{ample}(s) \subseteq \mathsf{ActLab}(s)$,

(**A2**) *If* $s \in \hat{\mathsf{S}}$ *and* $\mathsf{ample}(s) \neq \mathsf{Actions}(s)$ *each* $\alpha \in \mathsf{ample}(s)$ *is a stutter action,*

(**A3**) *For each path* $\sigma = s \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_n} s_n \xrightarrow{\gamma} \cdots$ *in* M *where* $s \in \hat{\mathsf{S}}$ *and* γ *is dependent on* $\mathsf{ample}(s)$ *there exists an index* $1 \leq i \leq n$ *such that* $\alpha_i \in \mathsf{ample}(s)$,
(**A4**) *If* (T, A) *is an EC in* $\hat{M}$ *and* $\alpha \in \bigcap_{t \in T} A(t)$, *then* $\alpha \in \bigcup_{t \in T} \mathsf{ample}(t)$
(**A5**) *If* $s \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n \xrightarrow{\gamma} s_{n+1}$ *is a* possible *path in* M *where* $s \in \hat{\mathsf{S}}$, $\alpha_1, \cdots, \alpha_n, \gamma \notin \mathsf{ample}(s)$ *and* γ *is probabilistic (i.e.* $0 < \mathsf{P}(s', \gamma, t') < 1$ *for some* s', t'*) then* $|\mathsf{ample}(s)| = 1$.

In case we assume that the schedulers are distributed, we can replace **A5** by (**A5′**) *If* $s \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n \xrightarrow{\gamma} s_{n+1}$ *is a* possible *path in* M *where* $s \in \hat{S}$, $\alpha_1, \cdots, \alpha_n, \gamma \notin \mathsf{ample}(s)$ *and* γ *is probabilistic then all the outputs in* $\mathsf{ample}(s)$ *are generated by the same atom* A

as formalized in the following theorem.

Theorem 2. *Let* ϕ *be an* $\mathsf{LTL}_{\neg\,\mathrm{next}}$ *formula,* M *be the MDP corresponding to a given Interleaved PIOA* P, *and let* $\hat{M}$ *be the system obtained by reducing* M *according to conditions* **A1**–**A4**, **A5′**. *Then,* $\sup_{\eta \in \mathsf{Dist}(P)} \Pr^{\eta}(\mathrm{Sat}_{\phi}) \leq \sup_{\eta \in \mathsf{Sched}(\hat{M})} \Pr^{\eta}(\mathrm{Sat}_{\phi})$.

In case we assume *strongly distributed schedulers*, **A5** can be disregarded [1].

Theorem 3. *Let* ϕ, M *and* P *be as in Theorem 2. Let* $\hat{M}$ *be the system obtained by reducing* M *according to conditions* **A1**–**A4**. *Then,* $\sup_{\eta \in \mathsf{SDist}(P)} \Pr^{\eta}(\mathrm{Sat}_{\phi}) \leq \sup_{\eta \in \mathsf{Sched}(\hat{M})} \Pr^{\eta}(\mathrm{Sat}_{\phi})$.

As an example, suppose that we are interested in finding the supremum probability that a system fails under strongly distributed schedulers. Since the restrictions on the ample set for strongly distributed schedulers do not consider input/output distinctions, the model P can be written in a language without such distinctions (such as the PRISM or LiQuor language). P can be seen as an Interleaved PIOA P_0 for which the I/O distinctions were eliminated, since the reduction algorithm do not consider them. In this case, the corresponding MDP M is equivalent to P. Suppose that 0.1 is the highest probability of failure allowed by the specification. Moreover, suppose that, by using the model checking algorithm in [4], we calculate that the supremum probability of a failure quantifying over all schedulers is 0.15. According to this analysis, the system would not meet the specification. However, the schedulers yielding probabilities greater than 0.1 might be "unrealistic" schedulers as the ones explained in the introduction and Sec. 3. Suppose that we construct $\hat{M}$ as described above. Then, we can use the algorithm in [4] to calculate $S = \sup_{\eta \in \mathsf{Sched}(\hat{M})} \Pr^{\eta}(\mathrm{Sat}_{\mathrm{True\,UFail}})$. If $S = 0.05$, then the theorem above ensures that $\sup_{\eta \in \mathsf{Dist}(P_0)} \Pr^{\eta}(\mathrm{Sat}_{\mathrm{True\,UFail}}) \leq 0.05$, and so the system meets the specification. In this sense, the bounds are *safe* with respect to $\mathsf{Dist}(P_0)$ (strictly speaking they are safe for every P_0 such that, by

[1] In the Appendix, we illustrate the need of **A5′** under distributed schedulers, and the impact of its elimination in favour of strongly distributed schedulers.

eliminating the I/O distinctions, P_0 becomes P [2]). Note that, in this case, the reduction has prevented some schedulers that are not strongly distributed and so the verification on $\hat{M}$ is more accurate than the verification on P.

With respect to distributed schedulers, the restriction **A5′** relies on the input/output distinctions. In case the model is written in a language that does not have such distinctions, one may think that every transition in the model can play the role of an output and also the role of an input. Then, one can obtain restrictions that imply **A5′**. One of such restrictions being: if $\mathsf{ample}(s) \neq \mathsf{Actions}(s)$, then either the ample set has a single action, or all actions in the ample set correspond to internal transitions (i.e., transitions without synchronization) in the same atom.

Reduction of the state space and transitions. In order to compare our approach to POR against the existing one, we reduced the model of the dining philosophers problem in [3]. Our reduced model is generated by a program written ad hoc for this problem. The program inserts additional conditions that disallow transitions not present in the ample sets. Table (a) below compares the ratio of reduction (that is, # in the reduced system / # in the original system): **(1)** assuming strongly distributed schedulers, **(2)** reported in [1] (in which the reduction is also not automated), and **(3)** reported in [3] (obtained using LiQuor). N indicates the amount of philosophers. Our approach clearly improves the results in [1]. With respect to the states, our results are similar to the ones in [3]. However, with respect to the transitions, we obtain a significant improvement. The amount of transitions is also important, since it heavily influences the time spent by the algorithms to analyse MDPs. So, our approach improves not only the results in [1], but also the results obtained with a fully functional tool in [3].

	(1)		**(2)**	**(3)**	
N	States	Transitions	States	States	Transitions
3	0.42	0.24	0.76	0.70	0.40
4	0.36	0.20	0.68	0.36	0.36
5	0.36	0.21	0.66	0.33	0.31

(a)

m	p_m **(1)**	p_m **(2)**
9	0.11	1.00
13	0.01	0.78
17	0.00	0.05

(b)

Tables: Experimental results

The Appendix contains details on the experiments, as well as a description of ample sets that are only possible using our approach.

Improving the bounds. For the example shown in the introduction, both of our variants allow the reduction in which the ample set for the initial state does not have the probabilistic transition. Using this reduction, the maximum probability of guessing is 1/2, as expected.

Using this kind of "smart" reductions, one can obtain tighter bounds on worst-case probabilities. We used ad hoc reductions in order to analyse a simple

[2] In particular, the bounds are safe for the Interleaved PIOA that has both an output and an input transition for each transition in P. Such Interleaved PIOA covers all possibilities of input/output interaction.

protocol for *anonymous* fair service. The protocol is described in the Appendix, and here we give a brief description. A server must serve two clients in a fair fashion regardless of the rates at which they ask for service. In addition, the clients cannot be identified, and so the server cannot simply count how much times it has served each of the clients. The protocol tosses a coin in order to choose the order in which incoming requests are served. If the clients were allowed to see the outcome of the coin, they would be able to ask for service in such a way that one of them is served more times than the other one.

Since we focus on reachability properties (and we are still not able to verify long-run properties), we modelled the system so that it stops after one of the clients is served 20 times. Then, we used PRISM to calculate the maximum probability p_m that, at any point of execution, the amount n_1 of replies to client 1 is greater than or equal to $n_2 + m$, where n_2 is the amount of replies to client 2 and m is a parameter of the property.

Table (b) shows the values of p_m for $m = 9, 13, 17$ both for **(1)** the reduced system and **(2)** the original system under unrestricted schedulers.

5 Discussion and further work

Related work. In the existing frameworks which are similar to our, there is no explicit nondeterminism as a consequence of interleaving. In [11], there is no treatment of composition, so nondeterminism is treated abstractly without explicit relation to interleaving. In this work schedulers are restricted in a way they are only allowed to partially observe a state along history (so different states may be observed as equivalent), and are thus related to partially observable Markov decision processes (POMDPs). A valuable source of information on POMDPs is [5]. The framework in [12] is synchronous: a step of the whole system is obtained by taking a step in every component, so, there is no interleaving at all. Closest to our work is [7] in which a token is used to decide the next component to execute. Since the behaviour of the token is specified by the components themselves, there is no explicit interleaving nondeterminism. Instead, we presented a framework in which the interleaving nondeterminism is made explicit in the usual way as a consequence of composition. This framework is the basis of the study of distributed probabilistic systems in a more realistic setting than usual. We restricted the set of valid schedulers to what we think is an adequate interpretation of distributed behaviour so that bounds of the maximum/minimum probabilities that a property holds are realistic.

Generality of our result. Theorems 2 and 3 are proven using a more general theorem. For a formal statement and a proof of such theorem, see the Appendix. Here, we present an overview. Given a class $\mathcal{S}$ of *deterministic* (also called non-randomized) schedulers complying with certain conditions, the general theorem states that **A1**–**A4**, **A5*** is a set of valid restrictions on ample sets, where **A5*** is defined as follows:

(**A5***) *For every scheduler* $\eta \in \mathcal{S}$,

$$\left(\eta(\sigma s \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n) = \beta \wedge \eta(\sigma s \xrightarrow{\alpha'_1} s'_1 \xrightarrow{\alpha'_2} s'_2 \cdots \xrightarrow{\alpha'_{n'}} s'_{n'}) = \beta'\right) \implies \beta = \beta',$$

where σs is a path whose last state is s, $\alpha_i, \alpha'_i \notin \mathsf{ample}(s)$, $\beta, \beta' \in \mathsf{ample}(s)$.

Theorem 2 is thus obtained by specializing $\mathbf{A5}^*$ for the case in which $\mathcal{S}$ is the set of deterministic distributed schedulers (deterministic distributed schedulers are as powerful as general distributed schedulers). Theorem 3 is proven by obtaining a superset Q of deterministic strongly distributed schedulers (considering this superset is necessary, since deterministic strongly distributed schedulers are not sufficiently powerful). Then, the general theorem is specialized for Q, yielding Theorem 3.

One important question is to which extent there exist other "natural" classes of schedulers for which better bounds and/or smaller systems can be obtained using POR.

Appendix

1 Deterministic schedulers

In this section, we consider deterministic (also called nonrandomized) schedulers and prove some results that will be used in later sections.

We defined schedulers so that they map into distributions on transitions. We say that a scheduler (distributed or strongly distributed) is *deterministic* if all the choices in all the input (output, interleaving, resp.) schedulers choose a reactive transition (generative transition, atom, resp.) with probability 1.

Let $\mathsf{DetDist}(P)$ be the subset of all deterministic distributed schedulers of P. Similarly, let $\mathsf{DetSDist}(P)$ be the set of all deterministic strongly distributed schedulers of a system P, respectively.

Fortunately, for every system P, the class of deterministic distributed schedulers (denoted by $\mathsf{DetDist}(P)$) is equally expressive as the class of all distributed schedulers (denoted by $\mathsf{Dist}(P)$) if we aim to find the maximum (or minimum) probability of a given measurable set of infinite paths.

Theorem 4. *For any set S of infinite traces, S being measurable, we have that*

$$\sup_{\eta \in \mathsf{DetDist}(P)} \Pr^{\eta}(S) = \sup_{\eta \in \mathsf{Dist}(P)} \Pr^{\eta}(S)$$

The proof of this theorem is very long and so we split it in several lemmata.

First, we need some elements from probability theory. These definitions and the proofs not given here can be found at [21].

Definition. *Given a set Σ, a semi-ring is a set $\mathcal{S} \subseteq \mathcal{P}(\Sigma)$ complying:*

- $\emptyset \in \mathcal{S}$,
- $A, B \in \mathcal{S} \implies A \cap B \in \mathcal{S}$,
- $A, B \in \mathcal{S} \implies \exists n \geq 0, \exists A_i \in \mathcal{S} : A \setminus B = \biguplus_{i=1}^{n} A_i$.

A ring is a set $\mathcal{R} \subseteq \mathcal{P}(\Sigma)$ complying:

- $\emptyset \in \mathcal{R}$,
- $A, B \in \mathcal{R} \implies A \cup B \in \mathcal{S}$,
- $A, B \in \mathcal{S} \implies A \setminus B \in \mathcal{S}$.

The ring $\mathcal{R}(\mathcal{S})$ generated by a semi-ring $\mathcal{S}$ is the least ring containing $\mathcal{S}$.

It can be proven that each element in the ring generated by a semi-ring $\mathcal{S}$ is of the form $\biguplus_{i=1}^{n} A_i$ with $A_i \in \mathcal{S}$. The set of whose elements are all the sets $[\sigma]$ forms a semi-ring. In the following, we denote this semi-ring by $\mathcal{S}$.

The following lemma states that the probability of any measurable set can be approximated as the probability of a countable disjoint union of sets of extensions.

Lemma 1. *Let $\mathcal{C}^{\omega}$ be the set*

$$\left\{ \{A_i\}_{i=1}^{\infty} \mid \forall i, j, i \neq j \bullet A_i \in \mathcal{S} \wedge A_i \cap A_j = \emptyset \right\}.$$

For every measurable set of infinite paths S, we have

$$\Pr^{\eta}(S) = \inf_{\{C \in \mathcal{C}^{\omega} \mid S \subseteq \biguplus_{A \in C} A\}} \sum_{A \in C} \Pr^{\eta}(A) .$$

Proof. An $\mathcal{R}$-cover of a set S is a set $\{B_i\}_{i=1}^{\infty}$ where $B_i \in \mathcal{R}$ and $S \subseteq \bigcup_{n=1}^{\infty} B_i$. Let $\mathcal{P}(S)$ be the set of all the $\mathcal{R}$-covers of S. The probability of a measurable set S in the σ-algebra generated by the semi-ring $\mathcal{S}$ can be defined as

$$\inf_{\{B_i\}_{i=1}^{\infty} \in \mathcal{P}(S)} \sum_{i=1}^{\infty} \Pr^{\eta}(B_i)$$

(see [21]) Given an $\mathcal{R}$-cover $\{B_i\}$ for S where each B_i is of the form $\biguplus_{k=0}^{n_i} A_k^i$, we define an element C in $\mathcal{C}^\omega$ as follows: $A \in C$ iff $A = A_k^i$ for some i,k and there is no $A_{k'}^{i'}$ such that $A_k^i \subset A_{k'}^{i'}$. Since our semi-ring is the set of extension sets, in the construction of C we dropped the extensions $[\sigma']$ such that there exists $[\sigma]$ with σ being a prefix of σ'.

Then, we have

$$\sum_{n=1}^{\infty} \Pr^\eta(B_n) = \sum_{i=1}^{\infty}\sum_{k=0}^{n_i} \Pr^\eta(A_k^i) \geq \sum_{A\in C} \Pr^\eta(A)$$

In addition, C is an $\mathcal{R}$-cover of S, since in the construction of C we only dropped sets of extensions included in other sets of extensions.

So, for each $\mathcal{R}$-cover we found another $\mathcal{R}$-cover in $\mathcal{C}^\omega$ yielding less or equal probability, thus completing the proof. □

The following lemma concerns the infimum probability of "finite-horizon" properties of the form $\biguplus_{i=1}^{n}[\sigma_i]$. Note that the only choices affecting such probability are the choices for the paths having length less than $N = \max_i\{\text{len}(\sigma_i)\}$.

Lemma 2. *For all sequences of finite paths $\{\sigma_i\}_{i=1}^n$ such that $[\sigma_i] \cap [\sigma_j] = \emptyset$ for all $i \neq j$, there exists a deterministic distributed scheduler η^d such that*

$$\Pr^{\eta^d}(\biguplus_{i=1}^{n}[\sigma_i]) = \inf_{\eta\in \mathsf{Dist}(P)} \Pr^\eta(\biguplus_{i=1}^{n}[\sigma_i]) \ .$$

Proof. Similarly as in Lemma 3 in [13], given any distributed scheduler η and any local path σ^* we obtain a deterministic distributed scheduler $\det(\eta, \sigma^*)$ such that η chooses deterministically for σ^* and $\det(\eta, \sigma^*)$ yields less probability than η. In order to obtain the deterministic scheduler η^d, we successively transform η to choose deterministically for all the local paths whose length is less than N, where $N = \max_i\{\text{len}(\sigma_i)\}$. That is, we consider the scheduler $\eta^N = \det(\det(\cdots\det(\eta, \sigma^1), \cdots), \sigma^N)$, where $\sigma^1 \cdots \sigma^N$ are all the local paths whose length is less than N. Given the scheduler η^N, we consider each local path of length greater than or equal to N, and for these paths we define the new scheduler η^d to deterministically choose a transition (the particular transition chosen is not relevant, since the choices for paths of length greater than or equal to N do not affect the value of $\Pr^{\eta^d}(\biguplus_{i=1}^{n}[\sigma_i])$.

The existence of such η^d ensures that the infimum quantifying over deterministic schedulers is less than or equal to the infimum quantifying over possibly nondeterministic schedulers. In addition, we conclude that there exists a scheduler yielding the infimum probability, since there are only finitely many combinations of deterministic choices for the paths of length less than N.

With respect to the construction, the only difference with respect to the proof in [13] is that the choices must be made deterministic for every local path and for every input and output scheduler. In addition, the choices must be made deterministic for the interleaving scheduler, by considering every global path.

In order to show that our input/output mechanism does not introduce any issue compromising the construction, we illustrate how to transform the choices for the input schedulers, by mimicking the proof in [13]. In the proof, we manipulate finite paths. In order to do this, for a path $\sigma = s_1.c_1.\cdots.c_{n-1}.s_n$ we define $\sigma(i) = s_i$ and $\sigma\langle i\rangle = c_i$. In addition $\sigma\downarrow_i = s_1.c_1 \cdots c_{i-1}.s_i$, $\text{last}(\sigma) = s_n$ and $\text{len}(\sigma) = n$.

Let σ^* be a path of an atom A_i and let $a \in \mathsf{ActLab}_i$. We show how to make the choice deterministic for the input scheduler of A_i when a occurs in σ^*. Let r_{σ^*} be the set of all the paths in $\{\sigma_i\}_{i=1}^n$ such that "a occurs in σ^*", that is, there exists k_σ such that $\sigma\downarrow_{k_\sigma}[A_i] = \sigma^*$ and $label(\sigma\langle k_\sigma\rangle) = a$. The probabilities of the paths in r_{σ^*} are the only ones to be changed, since we are only changing $\Upsilon_i(\sigma^*, a)$. So, we show only that, for this set, the scheduler in which the choice is deterministic yields a probability less than or equal to the probability yielded by the original scheduler.

Let A_{g_σ} be the atom that generates the output a in $\sigma\downarrow_{k_\sigma}$ and g^σ be the corresponding generative transition. Let r_j^σ be the reactive transition executed by A_j when a occurs in σ in the k_σ-th step.

We will focus on $\Upsilon_i(\sigma^*, a)$. The probability of a path σ in r_{σ^*} is $\Upsilon_i(\sigma^*, a) \cdot r_i^\sigma(\sigma(k_\sigma + 1)) \cdot Q_\sigma$, where

$$\begin{array}{rcl} Q_\sigma = \Pr^\eta([\sigma\downarrow_{k_\sigma}]) \cdot \mathcal{I}(\sigma\downarrow_{k_\sigma})(A_{g_\sigma}) & \cdot & \Theta_{g_\sigma}(\sigma\downarrow_{k_\sigma}[g_\sigma])(g^\sigma) \cdot \prod_{w\in\{1,\cdots,m\}\setminus\{i\}} \Upsilon_{j_w}(\sigma\downarrow_{k_\sigma}[j_w], a)(r^\sigma_{j_w}) \\ & \cdot & g^\sigma(\pi_{g_\sigma}(\sigma(k_\sigma+1)), a) \cdot \prod_{w\in\{1,\cdots,m\}\setminus\{i\}} r^\sigma_{j_w}(\pi_{j_w}(\sigma(k_\sigma+1))) \\ & \cdot & \prod_{t=k_\sigma+1}^{\mathrm{len}(\sigma)-1} \eta(\sigma\downarrow_t)(\sigma\langle t\rangle) \cdot \sigma\langle t\rangle(\sigma(t), \sigma(t+1)) \end{array}$$

Now, we calculate,

$$\begin{array}{ll} & \sum_{\sigma\in r_{\sigma^*}} \Pr^\eta([\sigma]) \\ = & \{\text{Definition of probabilities for extensions}\} \\ & \sum_{\sigma\in r_{\sigma^*}} r_i^\sigma(\,\pi_i(\sigma(k_\sigma+1))\,)\ \Upsilon_i(\sigma^*)(r_i^\sigma)\ Q_\sigma \\ = & \{\text{Rearrange sums}\} \\ & \sum_{r_i}\sum_s \sum_{\{\sigma\in r_{\sigma^*}\mid r_i^\sigma = r_i \wedge \pi_i(\sigma(k_\sigma+1))=s\}} r_i^\sigma(\,\pi_i(\sigma(k_\sigma+1))\,)\ \Upsilon_i(\sigma^*)(r_i^\sigma)\ Q_\sigma \\ = & \{\text{Rearrange sums}\} \\ & \sum_{r_i}\sum_s \sum_{\{\sigma\in r_{\sigma^*}\mid r_i^\sigma = r_i \wedge \pi_i(\sigma(k_\sigma+1))=s\}} r_i(s)\ \Upsilon_i(\sigma^*)(r_i)\ Q_\sigma \\ = & \sum_{r_i} \Upsilon_i(\sigma^*)(r_i) \sum_s r_i(s) \sum_{\{\sigma\in r_{\sigma^*}\mid r_i^\sigma = r_i \wedge \pi_i(\sigma(k_\sigma+1))=s\}} Q_\sigma \end{array}$$

Let

$$r^* = \arg\min_{r_i} \Upsilon_i(\sigma^*)(r_i) \sum_s r_i(s) \sum_{\{\sigma\in r_{\sigma^*}\mid r_i^\sigma = r_i \wedge \pi_i(\sigma(k_\sigma+1))=s\}} Q_\sigma\,.$$

Since $\sum_{r_i} \Upsilon_i(\sigma^*)(r_i) = 1$, we have

$$\sum_{\sigma\in r_{\sigma^*}} \Pr^\eta([\sigma]) \geq \Upsilon_i(\sigma^*)(r^*) \sum_s r^*(s) \sum_{\{\sigma\in r_{\sigma^*}\mid r_i^\sigma = r^* \wedge \pi_i(\sigma(k_\sigma+1))=s\}} Q_\sigma\ ,$$

which is the probability using the scheduler $\det(\eta)$ that mimics η excepting for Υ_i. Now, $\Upsilon_i'(\sigma^*, a)(r^*) = 1$.

The choices for the output schedulers can be made deterministic in an easier way (since labels need not to be considered).

With respect to the interleaving scheduler, let σ^* be a path of the system of length less than N. Let r_{σ^*} be the set of all the paths σ_i having σ^* as suffix. Let $k = \mathrm{len}(\sigma^*)$. As before, for every $\sigma \in r_{\sigma^*}$, let g_σ be the atom that performs an output in the k-th step, and g^σ be the corresponding generative transition. Moreover, let a_σ be the label after the k-th step in σ and let r_j be the reactive transition executed by A_j after the k-th step. Let

$$\begin{array}{rcl} Q_\sigma = \Pr^\eta([\sigma\downarrow_k]) & \cdot & \prod_{w\in\{1,\cdots,m\}} \Upsilon_{j_w}(\sigma\downarrow_{k_\sigma}[j_w], a)(r_{j_w}) \\ & \cdot & \prod_{w\in\{1,\cdots,m\}} r_{j_w}(\pi_{j_w}(\sigma(k_\sigma+1))) \\ & \cdot & \prod_{t=k+1}^{\mathrm{len}(\sigma)-1} \eta(\sigma\downarrow_t)(\sigma\langle t\rangle)\ \sigma\langle t\rangle(\sigma(t), \sigma(t+1)) \end{array}$$

Then, we proceed similarly as before:

$$\begin{array}{ll} & \sum_{\sigma\in r_{\sigma^*}} \Pr^\eta([\sigma]) \\ = & \sum_{\sigma\in r_{\sigma^*}} \mathcal{I}(\sigma^*)(A_{g_\sigma})\ \Theta_{g_\sigma}(\sigma\downarrow_k[g_\sigma])(g^\sigma)\ \ g^\sigma(\pi_{A_{g_\sigma}}(\sigma(k+1)), a_\sigma)\ Q_\sigma \\ = & \\ & \sum_{A_i}\ \sum_{\sigma_i, g_i} \sum_{s_i, a} \\ & \quad \sum_{\{\sigma\in r_{\sigma^*}\mid A_{g_\sigma}=A_i \wedge \not\downarrow_k[i] = \sigma_i \wedge \pi_{A_i}(\sigma(k+1))=s_i \wedge g^\sigma = g_i \wedge a_\sigma = a\}} \\ & \qquad \mathcal{I}(\sigma^*)(A_{g_\sigma})\ \Theta_{g_\sigma}(\sigma\downarrow_k[A_{g_\sigma}])(g^\sigma)\ g^\sigma(\pi_{A_{g_\sigma}}(\sigma(k+1)), a_\sigma)\ Q_\sigma \\ = & \\ & \sum_{A_i}\ \sum_{\sigma_i, g_i} \sum_{s_i, a} \\ & \quad \sum_{\{\sigma\in r_{\sigma^*}\mid A_{g_\sigma}=A_i \wedge \not\downarrow_k[i] = \sigma_i \wedge \pi_{A_i}(\sigma(k+1))=s_i \wedge g^\sigma = g_i \wedge a_\sigma = a\}} \\ & \qquad \mathcal{I}(\sigma^*)(A_i)\ \Theta_{g_i}(\sigma\downarrow_k[A_i])(g_i)\ g_i(\pi_{A_i}(s), a)\ Q_\sigma \\ = & \\ & \sum_{A_i} \mathcal{I}(\sigma^*)(A_i) \sum_{\sigma_i, g_i} \Theta_i(\sigma_i)(g_i)\ \ \sum_{s_i, a} g_i(s_i, a) \\ & \qquad \sum_{\{\sigma\in r_{\sigma^*}\mid A_{g_\sigma}=A_i \wedge \not\downarrow_k[i] = \sigma_i \wedge \pi_{A_{g_\sigma}}(\sigma(k+1))=s_i \wedge g^\sigma = g_i \wedge a_\sigma = a\}} Q_\sigma \end{array}$$

As before, we take

$$A_{i^*} = \arg\min_i \mathcal{I}(\sigma^*)(i) \sum_{\sigma_i, g_i} \Theta_i(\sigma_i)(g_i) \sum_{s_i, a} g_i(s_i, a) \sum_{\{\sigma \in r_{\sigma^*} \mid g_\sigma = i \wedge \Updownarrow_k[i] = \sigma_i \wedge \pi_{g_\sigma}(\sigma(k+1)) = s_i \wedge g^\sigma = g_i \wedge a_\sigma = a\}} Q_\sigma$$

and we define $\mathcal{I}'(\sigma) = A_{i^*}$. □

Notation 1. According to our convenience, we may denote a deterministic scheduler η as a function mapping global paths to n-tuples of the form $(g_i, f_1, \cdots, f_N)$, where $f_j : \mathsf{ActLab}_j \to T_{R_j}$ (recall Def. 1). Each n-tuple of the form $(g_i, f_1, \cdots, f_{i-1}, f_{i+1}, \cdots, f_N)$ corresponds to several compound transitions: namely, there is one compound transition for each label in ActLab_i. Given an n-tuple as before and a label a, we obtain the compound transition $(g_i, a, f_{r_1}(a), \cdots, f_{r_k}(a))$, where $r_1, \cdots, r_k$ are the atoms that react to a. So, if η is obtained by composing $\mathcal{I}, \Theta_1, \cdots, \Theta_N$, $\Upsilon_1, \cdots, \Upsilon_N$ we write $\eta(\sigma) = (g_i, f_1, \cdots, f_{i-1}, f_{i+1}, \cdots, f_N)$ iff $\mathcal{I}(\sigma) = i$ and $\Theta_i(\sigma[i]) = g_i$ and $\Upsilon_j(\sigma[j], a) = f_j(a)$ for all atom j, for all $a \in \mathsf{ActLab}_i$.

Note that a function η mapping histories to n-tuples is not necessarily a distributed scheduler. In general, we call to these functions *arbitrary schedulers*. Given an arbitrary scheduler η, η is a distributed scheduler iff for all i, σ, σ' s.t. $\sigma[i] = \sigma'[i]$, **(1)** $\eta(\sigma) = (g_i, f_1, \cdots, f_N)$ implies that $\eta(\sigma')$ is of the form $(g_i, f'_1, \cdots, f'_N)$ and **(2)** $\eta(\sigma) = (g_j, \cdots, f_i, \cdots)$ implies that $\eta(\sigma')$ is of the form $(g'_{j'}, \cdots, f_i, \cdots)$. Since we focus on distributed schedulers, schedulers are supposed to be distributed, except when stated otherwise.

The following lemma concerns "infinite-horizon" properties of the form $\biguplus_{i=1}^{\infty}[\sigma_i]$, and shows how to construct an optimal scheduler for such properties using optimal schedulers for the "finite-horizon" approximations of $\biguplus_{i=1}^{\infty}[\sigma_i]$. This optimal scheduler will be used in the proof of Theorem 4. Our construction resembles the "limit construction" in [7, Sec. 4.3].

Lemma 3. *For all sequences of finite paths $\{\sigma_i\}_{i=1}^{\infty}$ such that $[\sigma_i] \cap [\sigma_j] = \emptyset$ let S_N be the set $\biguplus\{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N\}$. If there exists a sequence $\{\eta_N\}_{N=1}^{\infty}$ of deterministic schedulers such that, for all N,*

$$\Pr^{\eta_N}(S_N) = \inf_{\eta} \Pr^{\eta}(S_N)$$

*then there exists a deterministic arbitrary scheduler η^d such that **(1)** for all N exists $N' > N$ such that $\eta^d(\sigma) = \eta_{N'}(\sigma)$ for all path σ s.t. $\mathrm{len}(\sigma) \leq N$ and **(2)** $\eta^d = \inf_\eta \Pr^\eta(\biguplus_i[\sigma_i])$.*

Proof. In order to construct η^d, we will construct a sequence of schedulers $\{\eta^N\}_{N=0}^{\infty}$. Then, we simply define $\eta^d(\sigma) = \eta^{\mathrm{len}(\sigma)}(\sigma)$. The idea behind the construction of the schedulers η^N is that η^N must comply the following property: there exists a sequence $\{Z_i^N\}_{i=1}^{\infty}$ such that

$$\eta^N(\sigma) = \eta_{Z_i^N}(\sigma) \tag{3}$$

for all σ having length less than or equal to N, for all i.

The scheduler η^0 is simply η_1. The sequence $\{Z_i^0\}$ is the sequence $\{i\}_{i=1}^{\infty}$. It trivially complies with (3), since there are no paths of length 0.

In order to construct the scheduler η^N from the scheduler η^{N-1}, we define schedulers $\eta^{N-1,Q}$, where Q is a set of paths of length N. In addition, each scheduler $\eta^{N-1,Q}$ has a corresponding sequence $\{Z_i^{N-1,Q}\}_{i=1}^{\infty}$. Once this schedulers are defined, we define $\eta^N = \eta^{N-1,Q_N}$ and $Z^N = Z^{N-1,Q_N}$, where Q_N is the set of all paths of length N. We will construct the schedulers $\eta^{N-1,Q}$ in such a way that $\eta^{N,Q}(\sigma) = \eta_{Z_i^{N,Q}}(\sigma)$ for all σ such that $\sigma \in Q$ or $\mathrm{len}(\sigma) \leq N-1$. The scheduler $\eta^{N,\{\}}$ is η^{N-1}. Now, we show how to construct $\eta^{N,Q\cup\{\sigma^*\}}$ from $\eta^{N,Q}$.

We consider the sequence $\{\eta_{Z_i^{N,Q}}(\sigma^*)\}_{i=1}^{\infty}$. In this sequence, at least one element a^* is repeated infinitely many times. We let $\eta^{N,Q\cup\{\sigma^*\}}(\sigma^*) = a^*$, and let $Z^{N,Q\cup\{\sigma^*\}}$ be the infinite subsequence of $Z^{N,Q}$ complying $\eta_{Z^{N,Q\cup\{\sigma^*\}}}(\sigma^*) = a^*$ (this infinite subsequence is ensured to exist since a^* appears infinitely many times in $\{\eta_{Z_i^{N,Q}}(\sigma^*)\}_{i=1}^{\infty}$).

Now, we prove the properties for η^d enunciated in the theorem.

1. Given any N, we take any N' in the sequence Z^N such that $N' > N$. So, the property for η^d is implied by the property (3) for η^N.

2. Suppose, towards a contradiction, that $\Pr^{\eta^d}(\biguplus_i \sigma_i) > \inf_\eta \Pr^\eta(\biguplus_i \sigma_i)$. Since $\Pr^{\eta^d}(\biguplus_i \sigma_i) = \sum_i \Pr^{\eta^d}(\sigma_i)$, there exists N such that

$$\Pr^{\eta^d}(\bigUplus_i \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N\}) > \inf_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(\bigUplus_i \sigma_i) \tag{4}$$

Let $N' > N$ such that $\eta^d(\sigma) = \eta_{N'}(\sigma)$ for all paths σ such that $\mathrm{len}(\sigma) \leq N$ (its existence is ensured by the previous property) and let $\eta^{\inf}$ be such that $\Pr^{\eta^{\inf}}(\biguplus_i \sigma_i) < \Pr^{\eta^d}(\biguplus_i \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N\})$ (its existence is ensured because of (4)). Now, we reason

$$\begin{array}{ll} & \Pr^{\eta^d}(\biguplus \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N\}) \\ = & \Pr^{\eta_{N'}}(\biguplus \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N\}) \\ \leq & \Pr^{\eta_{N'}}(\biguplus \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N'\}) \ . \end{array} \tag{5}$$

In addition,

$$\begin{array}{ll} & \Pr^{\eta^d}(\biguplus_i \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N\}) \\ > & \Pr^{\eta^{\inf}}(\biguplus_i \sigma_i) \\ \geq & \Pr^{\eta^{\inf}}(\biguplus \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N'\}) \\ \geq & \{\text{Optimality condition for } \eta_{N'} \text{ (see theorem statement)}\} \\ & \Pr^{\eta_{N'}}(\biguplus \{[\sigma_i] \mid \mathrm{len}(\sigma_i) \leq N'\}) \ . \end{array}$$

This contradicts (5).

□

The following lemma simply combines Lemma 2 and Lemma 3 in order to show that deterministic schedulers are sufficient to obtain the infimum probability of an "infinite-horizon" property as before.

Lemma 4. *For all sequences of finite paths $S = \{\sigma_i\}_{i=1}^\infty$ s.t. $[\sigma_i] \cap [\sigma_j] = \emptyset$ for all $i \neq j$, there exists a deterministic distributed scheduler η^* such that $\Pr^{\eta^*}(\biguplus_{\{\sigma \in S\}}[\sigma]) = \inf_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(\biguplus_{\{\sigma \in S\}}[\sigma])$.*

Proof. For each n, Lemma 2 ensures the existence of a deterministic distributed scheduler η_n such that $\Pr^{\eta_n}(\biguplus_{\{\sigma \in S\}}[\sigma]) = \inf_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(\biguplus_{\{\sigma \in S \mid \mathrm{len}(\sigma_i) \leq n\}}[\sigma])$. So, Lemma 3 ensures the existence of an arbitrary scheduler η^d such that $\eta^d = \inf_\eta \Pr^\eta(\biguplus_{\{\sigma \in S \mid \mathrm{len}(\sigma_i) \leq n\}}[\sigma])$.

Now, we prove that this arbitrary scheduler is indeed distributed. Suppose, towards a contradiction, that there exist two paths σ, σ' and an atom A_i complying $\sigma[i] = \sigma'[i]$ such that

- $\eta^d(\sigma) = (g_i, f_1, \cdots, f_N)$ and $\eta^d(\sigma') = (g'_i, f'_1, \cdots, f'_N)$ with $g_i \neq g'_i$. Or
- $\eta^d(\sigma) = (g_j, \cdots, f_i, \cdots)$ and $\eta^d(\sigma') = (g'_j, \cdots, f'_i, \cdots)$ with $f_i \neq f'_i$.

Let $M = \max\{\mathrm{len}(\sigma), \mathrm{len}(\sigma')\}$. Then, by Lemma 3 there exists $M' > M$ such that $\eta_{M'}(\sigma) = \eta^d(\sigma)$ and $\eta_{M'}(\sigma') = \eta^d(\sigma')$. So, $\eta_{M'}$ would not be distributed, contradicting the hypothesis for the schedulers η_n. □

Proof (of Theorem 4). Given $\epsilon > 0$, we will find a deterministic distributed η^d such that $\sup_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(S) - \Pr^{\eta^d}(S) < \epsilon$.

Let η^s be such that $\sup_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(S) - \Pr^{\eta^s}(S) < \epsilon/2$. By Lemma 1 (applied to the set $\complement S$), there exists a sequence $\{[\sigma_i]\}_{i=1}^\infty$ of disjoint extensions sets such that $(\complement S) \subseteq \biguplus_i [\sigma_i]$ and

$$\Pr^{\eta^s}(\bigUplus_i [\sigma_i]) - \Pr^{\eta^s}(\complement S) < \epsilon/2 \ . \tag{6}$$

By Lemma 4, there exists a deterministic distributed scheduler η^d such that $\Pr^{\eta^d}(\biguplus_i[\sigma_i]) = \inf_{\eta\in\mathsf{Dist}(P)}\Pr^{\eta}(\biguplus_i[\sigma_i])$. In particular, $\Pr^{\eta^d}(\biguplus_i[\sigma_i]) \leq \Pr^{\eta^s}(\biguplus_i \sigma_i)$. So, from (6) we have

$$\Pr^{\eta^d}(\biguplus_i[\sigma_i]) - \Pr^{\eta^s}(\mathbb{C}S) < \epsilon/2 \ .$$

From which we obtain

$$1 - \Pr^{\eta^d}(\mathbb{C}\biguplus_i[\sigma_i]) - (1 - \Pr^{\eta^s}(S)) < \epsilon/2 \ ,$$

this inequation being equivalent to

$$\Pr^{\eta^s}(S) - \Pr^{\eta^d}(\mathbb{C}\biguplus_i[\sigma_i]) < \epsilon/2 \ . \tag{7}$$

Since $\mathbb{C}S \subseteq \biguplus_i[\sigma_i]$ we have that $\mathbb{C}\biguplus_i[\sigma_i] \subseteq S$. So, $\Pr^{\eta^d}(\mathbb{C}\biguplus_i[\sigma_i]) \leq \Pr^{\eta^d}(S)$. From (7) we obtain $\Pr^{\eta^s}(S) - \Pr^{\eta^d}(S) < \epsilon/2$. Then,

$$\sup_\eta \Pr^{\eta}(S) - \Pr^{\eta^d}(S) = \sup_\eta \Pr^{\eta}(S) - \Pr^{\eta^s}(S) + \Pr^{\eta^s}(S) - \Pr^{\eta^d}(S) < \epsilon/2 + \epsilon/2 = \epsilon \ .$$

□

Unfortunately the property in Theorem 4 is not valid for strongly distributed schedulers. In the following, we define the set of *reinforced strongly distributed schedulers*, denoted by $\mathsf{RSDist}(P)$. This set is a superset of $\mathsf{SDist}(P)$. The set $\mathsf{DetRSDist}(P)$, comprising the deterministic schedulers in $\mathsf{RSDist}(P)$, is as powerful as $\mathsf{RSDist}(P)$.

The set $\mathsf{RSDist}(P)$ is defined using a notion similar to projections, the *visible portions* of paths. The visible portions allow the schedulers to see more information than projections do. Intuitively, you may think that, when two atoms synchronize, each one pass all the information it has to the other atoms. The visible portion of σ is, then, the information an atom A has about σ when information is passed in synchronizations. A visible portion is a sequence

$$(s^1_{i^1_1},\cdots,s^1_{i^1_{N^1}}).(g^1_{j^1_1},a_1,r^1_{j^1_2},\cdots,r^1_{j^1_{M^1}}).\cdots.(g^{n-1}_{j^{n-1}_1},a_{n-1},r^{n-1}_{j^{n-1}_2},\cdots,r^{n-1}_{j^{n-1}_{M^{n-1}}}).(s^n_{i^n_1},\cdots,s^n_{i^n_{N^n}}) \ .$$

(Note that projections are visible portions.) We denote the set of visible portions by $\mathsf{Visibles}$. In order to define the portion $[\![\sigma]\!]_A$ of a path $\sigma = s_1.c_1.\cdots.c_{n-1}.s_n$ visible to an atom A, we describe an iterative procedure. In each step, we obtain a prefix of $[\![\sigma]\!]_A$. Let $[\![\sigma]\!]_A{\downarrow_k}$ be the prefix of $[\![\sigma]\!]_A$ having length k. In the first step, we obtain $[\![\sigma]\!]_A{\downarrow_1} = \mathsf{init}_A$. In the k-th step, we consider the set $K_{k,A}$ comprising all atoms A' such that there exist compound transitions $c_{k^{A'}_1},\cdots,c_{k^{A'}_p}$ in σ complying

(1) $k-1 \leq k^{A'}_1 < \cdots < k^{A'}_p$ (it may be the case that there exist a single compound transition. In this case, $p=1$ and hence it is possible that $k-1 = k^{A'}_1 = k^{A'}_p$), and
(2) $label(k^{A'}_1) \in \mathsf{ActLab}_{A'}$, and
(3) $label(k^{A'}_p) \in \mathsf{ActLab}_A$, and
(4) there exist atoms $A^{A'}_2,\cdots,A^{A'}_{p-1}$ such that for all $1 \leq q < p$, $label(c_{k^{A'}_q}) \in A^{A'}_q \cap A^{A'}_{q+1}$, where $A^{A'}_1 = A'$ and $A^{A'}_p = A$.

Note that it may be the case that $A \in K_{k,A}$. In particular, $label(c_{k-1}) = A \implies A \in K_{k,A}$, since we can take $k-1 = k^A_1 = k^A_p$. If $K_{k,A} = \emptyset$, then we let $[\![\sigma]\!]_A{\downarrow_k} = [\![\sigma]\!]_A{\downarrow_{k-1}}$. Note that, if $K_{k,A} \neq \emptyset$, then the atom A_g that generates the output in c_{k-1} is in $K_{k,A}$ (we can take $k-1 = k^{A_g}_1 = k^{A_g}_p$). If $K_{k,A} = \{m_1,\cdots,m_{N^k}\}$, w.l.o.g. we assume $A_g = m_1$. Then, let $[\![\sigma]\!]_A{\downarrow_k} = [\![\sigma]\!]_A{\downarrow_{k-1}}.(g_{m_1},a,r_{m_2},\cdots,r_{m_{N^k}}).(s_{m_1},\cdots,s_{N^k})$.

We define $\mathsf{RSDist}(P)$ as the set of schedulers such that condition Eqn. (2) holds on the interleaving schedulers, when projections are replaced by visible portions.

The following lemma is useful to give an alternative characterisation of $\mathsf{RSDist}(P)$. Roughly speaking, this lemma states that, if σ_A and σ_B are together in a global path σ (and the interleaving scheduler assigns positive probability to A), and σ_B and σ_C are together in a global path σ', then σ_A, σ_B and σ_C are together in a path σ'', in which all the other local paths are as in σ.

Lemma. *If* $\Pr^\eta([\sigma]) > 0$, $\Pr^\eta([\sigma']) > 0$, $[\![\sigma]\!]_B = [\![\sigma']\!]_B$, $\mathcal{I}(\sigma)(A) > 0$, *then there exists* σ'' *such that* $[\![\sigma'']\!]_T = [\![\sigma]\!]_T$ *for every* $T \neq C$, $[\![\sigma'']\!]_C = [\![\sigma]\!]_C$.

Proof. This lemma can be proven by induction on $\mathrm{len}(\sigma) = \mathrm{len}(\sigma')$. □

Lemma. *If* $\eta \in \mathsf{RSDist}(P)$, *then for every atom there exist a function* $\mathsf{rate}_A : \mathsf{Visibles} \to \mathbf{R}_{\geq 0}$ *such that* $\mathsf{rate}_A(\sigma_A) > 0$ *iff* A *has generative transitions enabled in* $\mathrm{last}(\sigma_A[A])$ *and for every* σ *there exists a set* $S_\sigma \subseteq \mathrm{Atoms}(P)$ *such that* $\mathcal{I}(\sigma)(A) = \frac{\mathsf{rate}([\![\sigma]\!]_A)}{\sum_{A \in S_\sigma} \mathsf{rate}([\![\sigma]\!]_A)}$.

Then, the analogous to Lemma 2 can be proven as Lemma 2 in [14].

Lemma 5. *For all sequences of finite paths* $\{\sigma_i\}_{i=1}^n$ *such that* $[\sigma_i] \cap [\sigma_j] = \emptyset$ *for all* $i \neq j$, *there exists* $\eta^d \in \mathsf{DetRSDist}(P)$ *such that*

$$\Pr^{\eta^d}(\biguplus_{i=1}^{n}[\sigma_i]) = \inf_{\eta \in \mathsf{RSDist}(P)} \Pr^\eta(\biguplus_{i=1}^{n}[\sigma_i]) \,.$$

The same argument as in the case of distributed schedulers allows to conclude the following theorem from Lemma 5.

Theorem 5. *For any set* S *of infinite traces,* S *being measurable, we have that*

$$\sup_{\eta \in \mathsf{DetRSDist}(P)} \Pr^\eta(S) = \sup_{\eta \in \mathsf{RSDist}(P)} \Pr^\eta(S)$$

2 Undecidability results for distributed schedulers

Given a set of states U, we denote by $\Pr^\eta(\mathrm{reach}(U))$ the probability $\Pr^\eta(\{\rho \mid \exists n \bullet \rho(n) \in U\})$ of reaching any of the states in U.

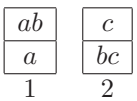
Theorem 6. *There are no algorithms that, given a set of states* U *and* $\epsilon > 0$, ***(1)*** *calculate* r *such that* $\| \sup_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(\mathrm{reach}(U)) - r\| < \epsilon$ *nor* ***(2)*** *whether or not* $\sup_{\eta \in \mathsf{Dist}(P)} \Pr^\eta(\mathrm{reach}(U)) = 1$ *nor* ***(3)*** *whether or not there exists* $\eta \in \mathsf{Dist}(P)$ *such that* $\Pr^\eta(\mathrm{reach}(U)) = 1$.

Proof. With respect to **(1)**, the proof is the same as in [13]. Indeed, in Sec. 5 in [13] it is explained how the proof strategy can be used for the Switched PIOA in [7]. Since only one of the two atoms in the reduction has generative transitions, the same strategy applies here as well.

With respect to **(2)** and **(3)**, our proof strategy is similar to the ones in [20]: we reduce the Post correspondence problem (PCP), which is known to be undecidable.

The PCP problem can be stated as follows: given words $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ over an alphabet $\mathcal{S}$. Is there a finite non-empty sequence of indices $k = k_1 \cdots k_m$ such that $u_{k_1} \cdots u_{k_m} = u_{k_1} \cdots u_{k_m}$?

Intuitively, we can think that we are given n blocks with two words, as shown in the following example:



In this example, the sequence of indices 1 2 is a solution of the problem.

We say that (w, k) is an upper pair iff $w = u_{k_1} \cdots u_{k_n}$. We say that (w, k) is a lower pair iff $w = v_{k_1} \cdots v_{k_n}$. Note that a word w can appear in an upper pair (in this case, we say that the word is an *upper word*) iff w is in the regular language $(u_1 + \ldots + u_n)^*$ (which we call the *upper language*), and similarly for the words that can appear in a lower pair.

Then, an instance of the PCP problem has a solution iff there exists an upper pair (w, k) such that (w, k) is also a lower pair.

Given a PCP instance $u_1, \ldots, u_n,\ v_1, \ldots, v_n$, we construct three atoms W, S, I. Roughly speaking, W chooses either "upper" or "lower". If W chooses "upper", then W probabilistically chooses an upper word w , communicating the symbols in w to S and the indices k_i to I (and similarly if W chooses "lower"). Once w ends (the end of w is also decided probabilistically, then W outputs *stop*. After *stop*, I is able to output any sequence of indices to S (some of the behaviours we will be interested in are the behaviours in which I communicates the indices it has received from W). Then, S has to guess whether W has chosen either "upper" or "lower". The set of states U is the set in which S has guessed correctly.

The set ActLab_W is $\mathcal{S} \cup \{1, \cdots, n\} \cup \{stop, \tau_W\}$. The behaviour of W is as follows: W has no nondeterministic choices. In the initial state there is a probabilistic transition $(\frac{1}{2} \xrightarrow{\tau_W!} initUp + \frac{1}{2} \xrightarrow{\tau_W!} \mathsf{initLo})$. The states *initUp* and *initLo* represent the fact that W has chosen "upper" or "lower" respectively. In *initUp* there is a probabilistic transition $(\frac{1}{n} \xrightarrow{1!} startU_1 + \cdots + \frac{1}{n} \xrightarrow{n!} startU_n)$. The states $startU_i$ represent the fact that the word w will start with u_i. Similarly, the states $startL_i$ represent the fact that word w will start with v_i. In each state $startU_i$ there is a transition $(1 \xrightarrow{u_{i_1}!} U_{i_1})$, where u_{i_1} is the first symbol in u_i [1] and U_{i_1} represents the fact that the first symbol in U_i has been output. From each state U_{i_j} with $j < \text{len}(u_i) - 1$ there is a transition $(1 \xrightarrow{u_{i_{j+1}}!} U_{i_{j+1}})$. In the state $U_{i_{\text{len}(u_i)-1}}$, there is a transition $(\frac{1}{2} \xrightarrow{u_{i_{\text{len}(w)}}!} initUp + \frac{1}{2} \xrightarrow{stop!} endWU)$. The state *endWU* indicates that the upper word has ended (similar definitions must be done in case W chooses "lower", where we have the state *endLU*). Since W must be input-enabled, each state has input transitions for each $l \in \mathsf{ActLab}_W$. However, because of the definition of the atoms, the paths in which the labels are output by other atoms have probability 0 for all schedulers, and so the definitions of the input transitions are irrelevant.

The set ActLab_I is $\mathcal{S} \cup \{1, \cdots, n\} \cup \{1', \cdots, n'\} \cup \{stop, stop'\}$. The labels $\{1', \cdots n'\}$ are indices to be communicated to S. However, such labels must be different from the labels $\{1, \cdots, n\}$ output by W, since S is not allowed to see such labels. The label *stop*′ simplifies the construction for similar reasons. In the initial state init_I there are input transitions $(1 \xrightarrow{i?} \mathsf{init}_I)$ for each $1 \leq i \leq n$ and also an input transition $(1 \xrightarrow{stop?} outputI)$. Other input transitions are irrelevant. In the state *outputI* there are transitions $(1 \xrightarrow{i'!} outputI)$ for each $1 \leq i \leq n$, and also a transition $(1 \xrightarrow{stop'!} endI)$.

The set ActLab_S is $\mathcal{S} \cup \{1', \cdots, n'\} \cup \{stop', \tau_S\}$. In the initial state there are input transitions $(1 \xrightarrow{a?} \mathsf{init}_S)$ for every $l \in \mathcal{S} \cup \{1', \cdots, n'\}$ and transition $(1 \xrightarrow{stop'?} guessS)$. In *guessS* there are two transitions: $(1 \xrightarrow{\tau_S!} tryUp)$ and $(1 \xrightarrow{\tau_S!} tryLo)$.

So, the set U to be reached is $\{(endWU, endI, tryUp), (endWL, endI, tryLo)\}$. We prove the following: there exists a distributed scheduler such that $\Pr^{\eta}(\text{reach}(U)) = 1$ iff the PCP problem has not a solution. In addition, $\sup_{\eta \in \mathsf{Dist}(P)} \Pr^{\eta}(\text{reach}(U)) = 1$ iff the PCP problem has not a solution.

Suppose that the problem has no solution. Then every pair (w, k) can be an upper or a lower pair, but it cannot be both. We can construct the following distributed scheduler for P: input and output schedulers for W are uniquely defined (there are no nondeterministic choices). The output scheduler for I chooses the transitions that output the indices in order as they were output by W. The output scheduler for S has to decide only between going to *tryUp* or going *tryLo*. The only

[1] For simplicity, we omitted the case in which some of the words u_k (v_k, resp.) are empty. In this case, when the index k is output in the state *initUp*, W returns to *initUp* instead of moving to $startU_k$.

paths with probability greater than 0 in which this choice is performed have a sequence of action labels of the form $a_1 \cdots a_q k_1 \cdots k_r stop'$. If $(a_1 \cdots a_q\,,\, k_1 \cdots k_r)$ is an upper pair, then the output scheduler chooses *tryUp*, otherwise it chooses *tryLo*. If the path has positive probability, and $a_1 \cdots a_r k_1 \cdots k_q$ is an upper pair, then, by construction of W, W is in state *endWU*. Conversely, if $a_1 \cdots a_r k_1 \cdots k_q$ is a lower pair, then W is in state *endWL*, and so the scheduler we constructed reaches U with probability 1.

Now assume that the PCP problem has a solution. Suppose (towards a contradiction) that $\sup_{\eta \in \mathsf{Dist}(P)} \Pr^{\eta}(\mathrm{reach}(U)) = 1$ (to get **(3)**, suppose that there exists $\eta \in \mathsf{Dist}(P)$ such that $\Pr^{\eta}(\mathrm{reach}(U)) = 1$). Then, by Theorem 4 for every $\epsilon > 0$ there exists a deterministic distributed scheduler η_ϵ such that $\Pr^{\eta_\epsilon}(\mathrm{reach}(U)) > 1 - \epsilon$.

Since the PCP problem has a solution, let $(w, k = k_1 \cdots k_r)$ be an upper pair that is also a lower pair. Let ϵ be $\frac{1}{2}(\frac{1}{n}\frac{1}{2})^{r+1}$. Then, there exists a two paths σ, σ' whose projection to I is of the form $k_1 \cdots k_r stop$ and, in one of them W has chosen "upper" while in the other one it has chosen "lower". For both σ and σ', the output scheduler for I starts to choose transitions in such a way that a sequence $l_1 \cdots l_{r'}\, stop'$ is output (if $stop'$ is never output, then a state in U cannot be reached and so the scheduler yields a probability less than or equal than $1 - \epsilon$). Then, in both σ, σ' the projection to S is $w l_1 \cdots l_{r'}\, stop'$. So, if the scheduler for W chooses "upper" in σ, then it also chooses "upper" in σ' (and so the scheduler reaches U with probability less than or equal than $1 - \epsilon$) and the same happens in case it chooses "lower".

Therefore, every deterministic scheduler reaches U with probability less than or equal to $1 - \epsilon$, thus contradicting Theorem 4. □

3 Strongly distributed schedulers

Theorem. *Let P be a system such that $A, B \in \mathrm{Atoms}(P)$. Consider the system P' such that $\mathrm{Atoms}(P') = \big(\,\mathrm{Atoms}(P) \setminus \{A, B\}\,\big) \cup \{AB\}$, where AB is the usual cross-product of A and B (as in, for instance, [7, p. 99]). Then, for every strongly distributed scheduler η for P, there exists a strongly distributed scheduler η' for P' yielding the same probability distribution on paths as η.*

Proof. We show that the condition imposed to the interleaving scheduler is sufficient to define an output scheduler for AB. Let σ_{AB} be a local path on AB, and let σ be a global path σ such that $\sigma[AB] = \sigma_{AB}$. Define

$$\Theta_{AB}(\sigma_{AB})(g_A) = \frac{\mathcal{I}(\sigma)(A)}{\mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B)} \Theta_A(\sigma[A])\,.$$

Note that the condition imposed to $\mathcal{I}$ ensures that the particular σ chosen is not relevant. Let $\mathcal{I}'$ be the interleaving scheduler for P_{AB} such that $\mathcal{I}'(\sigma)(AB) = \mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B)$ and $\mathcal{I}'(\sigma)(C) = \mathcal{I}(\sigma)(C)$ for any other atom C. We have to prove that the scheduler η' for P_{AB} obtained from $\mathcal{I}'$ as interleaving scheduler and Θ_{AB} as output scheduler for AB yields the same behaviour as the original scheduler η for P. To see this, note that for a path σ, the probability assigned to a generative transition g_A of A is $p_{\sigma,g_A} = \mathcal{I}(\sigma)(A) \cdot \Theta_A(\sigma[A])(g_A)$. Multiplying and dividing by $\mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B)$ yields

$$p_{\sigma,g_A} = \big(\mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B)\big) \left(\frac{\mathcal{I}(\sigma)(A)}{(\mathcal{I}(\sigma)(A) + \mathcal{I}(\sigma)(B))} \Theta_A(\sigma[A])(g_A) \right)\,,$$

which equals to $\mathcal{I}'(\sigma)(AB) \cdot \Theta_{AB}(\sigma[AB])(g_A)$, that is, the probability of p_{σ,g_A} in η'. The same reasoning allows to conclude a similar equality if atom B is considered instead of A. The input, output, and interleaving schedulers do not change in all other cases. □

Theorem 7. *Let $\mathcal{A} = \{A_1, \cdots, A_n\}$, $\mathcal{B} = \{B_1, \cdots, B_m\}$ be disjoint sets of atoms. Then, if Eqn. (2) holds, then $\frac{\sum_i \mathcal{I}(\sigma)(A_i)}{\sum_i \mathcal{I}(\sigma)(A_i) + \sum_j \mathcal{I}(\sigma)(B_j)} = \frac{\sum_i \mathcal{I}(\sigma')(A_i)}{\sum_i \mathcal{I}(\sigma')(A_i) + \sum_j \mathcal{I}(\sigma')(B_j)}$ holds whenever $\sigma[A] = \sigma'[A]$ for all $A \in \mathcal{A} \cup \mathcal{B}$ and $\sum_i \mathcal{I}(\sigma')(A_i) + \sum_j \mathcal{I}(\sigma')(B_j) \neq 0$.*

Proof. By induction on n. We prove the base case $n = 1$ by induction on m. If $m = 1$, the statement becomes Eqn. (2). For the inductive step, we need a preliminary equality. Note that, if $\mathcal{I}(\sigma)(A) \neq 0$ and $\mathcal{I}(\sigma')(A) \neq 0$ in Eqn. (2), then simple arithmetic gives

$$\frac{\mathcal{I}(\sigma)(B)}{\mathcal{I}(\sigma)(A)} = \frac{\mathcal{I}(\sigma')(B)}{\mathcal{I}(\sigma')(A)} \,. \tag{8}$$

The inductive step is

$$\frac{\mathcal{I}(\sigma)(A_1)}{\mathcal{I}(\sigma)(A_1) + \sum_j \mathcal{I}(\sigma)(B_j)} = \frac{\mathcal{I}(\sigma')(A_1)}{\mathcal{I}(\sigma')(A_1) + \sum_j \mathcal{I}(\sigma')(B_j)} \,.$$

First, we prove the case $\mathcal{I}(\sigma)(A_1) = 0$. In this case, either $\mathcal{I}(\sigma)(B_j) = 0$ for all j (in this case the condition $\mathcal{I}(\sigma)(A_1) + \sum_j \mathcal{I}(\sigma)(B_j) \neq 0$ is false, then the equation is not required to hold) or $\mathcal{I}(\sigma')(A_1) = 0$. To see this, suppose towards the contradiction that $\mathcal{I}(\sigma')(A_1) \neq 0$. Then, by Eqn. (2) it must be

$$\frac{\mathcal{I}(\sigma)(A_1)}{\mathcal{I}(\sigma)(A_1) + \mathcal{I}(\sigma)(B_{j^*})} = \frac{\mathcal{I}(\sigma')(A_1)}{\mathcal{I}(\sigma')(A_1) + \mathcal{I}(\sigma')(B_{j^*})}$$

where j^* is an index such that $\mathcal{I}(\sigma)(B_{j^*}) > 0$ (we don't need $\mathcal{I}(\sigma')(B_{j^*}) \neq 0$, since $\mathcal{I}(\sigma')(A_1) \neq 0$). So, since $\mathcal{I}(\sigma)(A_1) = 0$ then it must be $\mathcal{I}(\sigma')(A_1) = 0$, thus reaching a contradiction. So, the inductive step holds in case $\mathcal{I}(\sigma)(A_i) = 0$.

If $\mathcal{I}(\sigma)(A_1) \neq 0$, then either $\mathcal{I}(\sigma')(A_1) = 0$ and $\mathcal{I}(\sigma')(B_j) = 0$ for all j (and so the condition is not required to hold) or $\mathcal{I}(\sigma')(A_1) \neq 0$, and so we can use Eqn. (8) in the following calculation.

$$\begin{array}{ll} & \frac{\mathcal{I}(\sigma)(A_1)}{\mathcal{I}(\sigma)(A_1)+\sum_j \mathcal{I}(\sigma)(B_j)} \\ = & \{\text{Arithmetics}\} \\ & \left(\frac{\mathcal{I}(\sigma)(B_m)}{\mathcal{I}(\sigma)(A_1)} + \frac{\mathcal{I}(\sigma)(A_1)+\sum_{j=1}^{m-1} \mathcal{I}(\sigma)(B_j)}{\mathcal{I}(\sigma)(A_1)}\right)^{-1} \\ = & \{\text{Equation (8)}\} \\ & \left(\frac{\mathcal{I}(\sigma')(B_m)}{\mathcal{I}(\sigma')(A_1)} + \frac{\mathcal{I}(\sigma)(A_1)+\sum_{j=1}^{m-1} \mathcal{I}(\sigma)(B_j)}{\mathcal{I}(\sigma)(A_1)}\right)^{-1} \\ = & \{\text{Inductive hypothesis}\} \\ & \left(\frac{\mathcal{I}(\sigma')(B_m)}{\mathcal{I}(\sigma')(A_1)} + \frac{\mathcal{I}(\sigma')(A_1)+\sum_{j=1}^{m-1} \mathcal{I}(\sigma')(B_j)}{\mathcal{I}(\sigma')(A_1)}\right)^{-1} \\ = & \{\text{Arithmetics}\} \\ & \frac{\mathcal{I}(\sigma')(A_1)}{\mathcal{I}(\sigma')(A_1)+\sum_j \mathcal{I}(\sigma')(B_j)} \end{array}$$

Then, the statement holds for $n = 1$. For the remaining inductive step, we calculate:

$$\begin{array}{ll} & \frac{\sum_i \mathcal{I}(\sigma)(A_i)}{\sum_i \mathcal{I}(\sigma)(A_i)+\sum_j \mathcal{I}(\sigma)(B_j)} \\ = & \frac{\sum_{i=1}^{n-1} \mathcal{I}(\sigma)(A_i)+\mathcal{I}(\sigma)(A_n)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma)(A_i)+\mathcal{I}(\sigma)(A_n)+\sum_j \mathcal{I}(\sigma)(B_j)} \\ = & \frac{\sum_{i=1}^{n-1} \mathcal{I}(\sigma)(A_i)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma)(A_i)+\mathcal{I}(\sigma)(A_n)+\sum_j \mathcal{I}(\sigma)(B_j)} + \frac{\mathcal{I}(\sigma)(A_n)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma)(A_i)+\mathcal{I}(\sigma)(A_n)+\sum_j \mathcal{I}(\sigma)(B_j)} \\ = & \{\text{Inductive hypothesis for } \{A_i\}_{i=1}^{n-1},\ A_n \cup \{B_j\}_{j=1}^{m}\} \\ = & \frac{\sum_{i=1}^{n-1} \mathcal{I}(\sigma')(A_i)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma')(A_i)+\mathcal{I}(\sigma')(A_n)+\sum_j \mathcal{I}(\sigma')(B_j)} + \frac{\mathcal{I}(\sigma)(A_n)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma)(A_i)+\mathcal{I}(\sigma)(A_n)+\sum_j \mathcal{I}(\sigma)(B_j)} \\ = & \{\text{Base case with } \{A_n\},\ \{B_i\}_{i=1}^{m} \cup \{A_i\}_{i=1}^{n-1}\} \\ = & \frac{\sum_{i=1}^{n-1} \mathcal{I}(\sigma')(A_i)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma')(A_i)+\mathcal{I}(\sigma')(A_n)+\sum_j \mathcal{I}(\sigma')(B_j)} + \frac{\mathcal{I}(\sigma')(A_n)}{\sum_{i=1}^{n-1} \mathcal{I}(\sigma')(A_i)+\mathcal{I}(\sigma')(A_n)+\sum_j \mathcal{I}(\sigma')(B_j)} \end{array}$$

□

4 Undecidability results for strongly distributed schedulers

4.1 Qualitative undecidability

In order to prove the qualitative result, we reduce the supremum acceptance problem for probabilistic finite-state automata (PFA) to the supremum reachability problem for strongly distributed schedulers. The supremum acceptance problem for PFA was proven undecidable in [18]. This reduction is used also in [13].

A PFA is a quintuple (Q, Σ, l, q_i, q_f) where Q is a finite set of *states* with $q_i, q_f \in Q$ being the *initial* and *accepting state* respectively, Σ is the *input alphabet*, and $l : \Sigma \times Q \to (Q \to [0,1])$ is the *transition function* s.t. $l(\alpha, q)$ is a distribution for all $\alpha \in \Sigma$ and $q \in Q$. A word w is an infinite sequence of symbols from Σ. The probability $\Pr(\text{accept}(w))$ of accepting a word is the probability of reaching q_f by starting from q_i and succesively choosing the next state according to l and the next symbol in w.

Given a PFA F, we can construct an atom A_F having only reactive transitions. Such atom A_F is defined as follows: $S_F = Q$, $\mathsf{ActLab}_F = \Sigma$, $G_F(s) = \emptyset$ for all s, $R_F(s,a) = \{l(a,s)\}$ and $\mathsf{init}_F = q_i$.

In our reduction, in addition to A_F we consider atoms A_α, one for each $\alpha \in \Sigma$. The atom A_α is defined as follows: $S_{A_\alpha} = \{\mathsf{init}_\alpha\}$, $\mathsf{ActLab}_F = \{\alpha\}$, $G_F(\mathsf{init}_\alpha) = \{g_\alpha\}$ where $g_\alpha(\mathsf{init}, \alpha) = 1$ and $R_\alpha(\mathsf{init}, \alpha) = \{r_\alpha\}$ where $r_\alpha(\mathsf{init}) = 1$ (although r_α is not used, it is required by the input-enabledness condition).

As in [14] (Proof of Theorem 1), it can be proven that, for the system having atoms A_F, $\{A_\alpha \mid \alpha \in \Sigma\}$, deterministic strongly distributed schedulers have the same power as strongly distributed schedulers.

Each deterministic strongly distributed scheduler defines a word for w. Then, since there is no algorithm to calculate not approximate $\sup_w \Pr(\text{accept}(w))$, there is no algorithm to calculate nor approximate $\sup_{\eta \in \mathsf{DetSDist}(P)} \Pr^\eta(\text{reach}(U))$, where U is the set of global states in which the local state of A_F corresponds to an accepting state in F.

4.2 Quantitative undecidability

We use the same idea as in the case of distributed schedulers. When proving such result, we defined three atoms W, S and I. Here, we reuse the atom W, excepting for a little modification explained later. The atom S is replaced by two atoms S_{Up}, S_{Lo}, each one of them behaves as S, until the point in which S decides, i.e. at the state *guessS*. In this state, S_{Up} has enabled the transition $(1 \xrightarrow{u!} tryUp)$, and S_{Lo} has enabled the transition $(1 \xrightarrow{l!} tryLo)$. In W, the state *endWU* has the following input transitions: $(1 \xrightarrow{u?} good)$ and $(1 \xrightarrow{l?} bad)$. The state *endLU* has the following transitions $(1 \xrightarrow{u?} bad)$ and $(1 \xrightarrow{l?} good)$. Then, the state *good* is reached in the cases in which "the set of atoms $\{S_{Up}, S_{Lo}\}$" guesses correctly "upper" or "lower".

The atom I is replaced by a set of atoms $\{I_i\}_{i=1}^n \cup \{I_{stop}\}$. Each I_i has all the input transitions in I. In addition, in the initial state there is an input transition $(1 \xrightarrow{stop'?} endI_i)$. The atom I_{stop} has all the input transitions in I, and only one output transition $(1 \xrightarrow{stop'!} endI_{stop})$. So, once I_{stop} decides to stop, all the I_i reach the state $endI_i$. Each atom I_i has enabled the output transition $(1 \xrightarrow{i'!} \mathsf{init}_I)$.

Note that the set of atoms in this new reduction can be partitioned into the subsets $\{W\}$, $\{I_i\}_{i=1}^n \cup \{I_{stop}\}$ and $\{S_{Up}, S_{Lo}\}$. At every point of the execution, all the atoms in exactly one of these sets have an output transition enabled. This property can be used to prove that, for this particular system, deterministic strongly distributed schedulers are as powerful as strongly distributed schedulers.

Then, we can repeat the argument in the proof for distributed schedulers in order to prove that the supremum probability of reaching *good* is 1 iff the PCP problem has no solution.

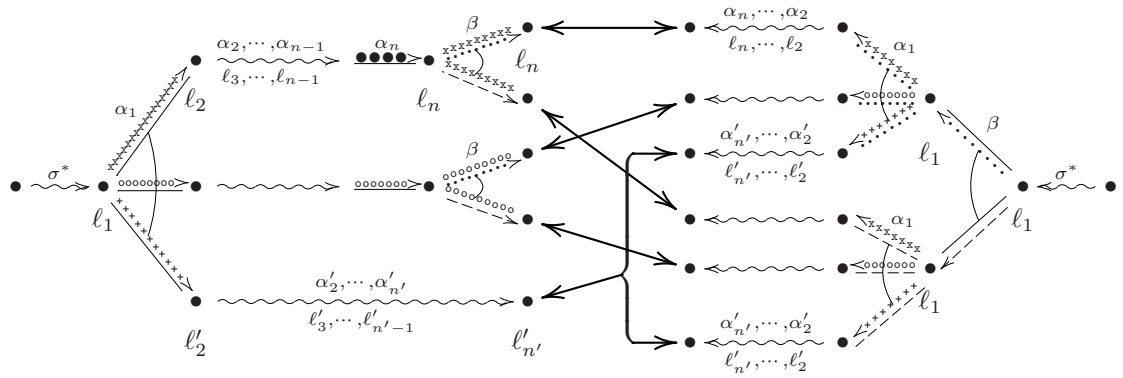


Figure 4: Correspondence between paths in η and $\eta[\sigma^* \leftarrow \beta]$

5 The POR technique

5.1 Condition on the classes $\mathcal{S}$ for the general theorem

In the nonprobabilistic case, the correctness of POR is proven by showing that each execution in the original system can be simulated by an equivalent execution in the reduced system. The execution in the reduced system is obtained by exchanging the order of independent actions.

In the probabilistic case, each execution is a probabilistic tree (corresponding to a scheduler), and so it not obvious what it means to exchange the order of independent actions. The condition we impose to $\mathcal{S}$ in our general theorem is related to the way in which independent actions are moved in a probabilistic tree.

In the following we consider MDPs obtained from Interleaved PIOAs. The "actions" we consider are the actions identifiers in Def. 3. Given a deterministic scheduler η, a path σ^* ending in the state s and $\beta \in \mathsf{ample}(s)$ such that for all $\alpha_i \notin \mathsf{ample}(s)$,

$$\big(\eta(\sigma^* \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n) = \beta' \implies \beta' = \beta \tag{9}$$

we define the scheduler $\eta[\sigma^* \leftarrow \beta]$ in which β is moved right after σ^* (that is $\eta(\sigma^{`*}) = \beta$). Note that Eqn. (9) does not ensure that a path such that $\eta(\sigma^* \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n) = \beta$ exists. In this case, β can also be inserted after σ^*.

Figure 4 illustrates the transformation from η to $\eta[\sigma^* \leftarrow \beta]$. Note that, because of the POR restrictions, the atoms $\mathsf{Inv}(\beta)$ whose state may be changed by β are disjoint of the atoms changed by the α_i. Dashed and dotted lines represent the projections of paths over $\mathsf{Inv}(\beta)$. Lines made up of `x`, `o` and `+` represent the projections of paths over $\mathrm{Atoms}(P) \setminus \mathsf{Inv}(\beta)$. An arrow relating a path in η to path(s) in $\eta[\sigma^* \leftarrow \beta]$ indicate that the (sets of) path(s) in $\eta[\sigma^* \leftarrow \beta]$ has the same probability as the path in η.

Next, we define $\eta[\sigma^* \leftarrow \beta]$. We let $\eta[\sigma^* \leftarrow \beta](\sigma s) = \beta$. For every σ such that σ is not a prefix of σ^* (and $\sigma \neq \sigma^*$), we let $\eta[\sigma^* \leftarrow \beta](\sigma) = \eta(\sigma^*)$. We need several definitions before presenting the case in which σ^* is a prefix of σ.

In the following, we write a state s' as product $(s'^{\beta}, s'^{\neg\beta})$, where $s'^{\beta} \in \prod_{i \in \mathsf{Inv}(\beta)} \mathsf{S}_i$ and $s'^{\neg\beta} \in \prod_{i \notin \mathsf{Inv}(\beta)} \mathsf{S}_i$. In particular, $\mathrm{last}(\sigma^*) = s = (s^{\beta}, s^{\neg\beta})$.

First, we need to relate the paths in the original system to the paths in the reduced system as shown in Fig. 4. For each finite path σ such that σ^* is a prefix of σ and $\Pr^{\eta_{i,S}}(\sigma) > 0$, we define a set of corresponding paths $\mathcal{C}(\sigma)$. This correspondence will be used to define $\eta[\sigma^* \leftarrow \beta]$, as well as to prove that the probabilities of any stuttering invariant language coincide both for the original and the reduced system.

The correspondence is defined according to the actions in σ.

1. If σ is of the form

$$\sigma^* \xrightarrow{\alpha_1} (s_1^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s_n^{\beta}, s_n^{\neg\beta}) \qquad \triangleleft\alpha\triangleright$$

and $\alpha_k \notin \mathsf{ample}(s)$ for all k, then $s_k^{\beta} = s^{\beta}$ for all k, since the actions α_k are independent from β and β is enabled in all the states s_k. So, σ is of the form

$$\sigma^* \xrightarrow{\alpha_1} (s^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s^{\beta}, s_n^{\neg\beta}) \,.$$

For such a σ, we define

$$\mathcal{C}(\sigma) = \{\sigma^* \xrightarrow{\beta} (s'^{\beta}, s^{\neg\beta}) \xrightarrow{\alpha_1} (s'^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s'^{\beta}, s_n^{\neg\beta}) \mid \mathsf{P}(\, s, \beta, (s'^{\beta}, s^{\neg\beta})\,) > 0\} \,.$$

We consider the case in which $\sigma = \sigma^*$ as a corner case of this one. So,

$$\mathcal{C}(\sigma^*) = \{\sigma^* \xrightarrow{\beta} (s'^{\beta}, s^{\neg\beta}) \mid \mathsf{P}(\, s, \beta, (s'^{\beta}, s^{\neg\beta})\,) > 0 \,.$$

In the following, we write $\sigma \sim \triangleleft\alpha\triangleright$ to denote that σ has the form of the path marked with $\triangleleft\alpha\triangleright$ above.

2. If σ is of the form

$$\sigma^* \xrightarrow{\alpha_1} (s_1^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s_n^{\beta}, s_n^{\neg\beta}) \xrightarrow{\beta} (s'^{\beta}, s'^{\neg\beta}) \qquad \triangleleft\alpha\,\beta\triangleright$$

and $\alpha_k \notin \mathsf{ample}(s)$ for all k, then we have $s_k^{\beta} = s^{\beta}$ for all k as before. In addition, $s'^{\neg\beta} = s_n^{\neg\beta}$. So, σ is of the form

$$\sigma^* \xrightarrow{\alpha_1} (s^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s^{\beta}, s_n^{\neg\beta}) \xrightarrow{\beta} (s'^{\beta}, s_n^{\neg\beta}) \,.$$

For such a σ, we define

$$\mathcal{C}(\sigma) = \{\sigma^* \xrightarrow{\beta} (s'^{\beta}, s^{\neg\beta}) \xrightarrow{\alpha_1} (s'^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s'^{\beta}, s_n^{\neg\beta})\} \,.$$

(Note that, in this case, $\mathcal{C}(\sigma)$ is a singleton set.)

In the following, we write $\sigma \sim \triangleleft\alpha\,\beta\triangleright$ to denote that σ has the form of the path marked with $\triangleleft\alpha\,\beta\triangleright$ above.

3. If σ is of the form

$$\sigma^* \xrightarrow{\alpha_1} (s_1^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s_n^{\beta}, s_n^{\neg\beta}) \xrightarrow{\beta} (s'^{\beta}, s'^{\neg\beta}) \xrightarrow{\gamma_1} (s_1'^{\beta}, s_1'^{\neg\beta}) \cdots \xrightarrow{\gamma_{n'}} (s_{n'}'^{\beta}, s_{n'}'^{\neg\beta}) \qquad \triangleleft\alpha\,\beta\,\gamma\triangleright$$

and $\alpha_k \notin \mathsf{ample}(s)$ for all k, we have that σ is of the form

$$\sigma^* \xrightarrow{\alpha_1} (s^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s^{\beta}, s_n^{\neg\beta}) \xrightarrow{\beta} (s'^{\beta}, s_n^{\neg\beta}) \xrightarrow{\gamma_1} (s_1'^{\beta}, s_1'^{\neg\beta}) \cdots \xrightarrow{\gamma_{n'}} (s_{n'}'^{\beta}, s_{n'}'^{\neg\beta}) \,.$$

For such a σ, we define

$$\mathcal{C}(\sigma) = \{\sigma^* \xrightarrow{\beta} (s'^{\beta}, s^{\neg\beta}) \xrightarrow{\alpha_1} (s'^{\beta}, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s'^{\beta}, s_n^{\neg\beta}) \xrightarrow{\gamma_1} (s_1'^{\beta}, s_1'^{\neg\beta}) \cdots \xrightarrow{\gamma_{n'}} (s_{n'}'^{\beta}, s_{n'}'^{\neg\beta})\} \,.$$

(Again, $\mathcal{C}(\sigma)$ is a singleton set.)

In the following, we write $\sigma \sim \triangleleft\alpha\,\beta\,\gamma\triangleright$ to denote that σ has the form of the path marked with $\triangleleft\alpha\,\beta\,\gamma\triangleright$ above.

4. If σ^* is not a prefix of σ (and $\sigma \neq \sigma^*$), we define $\mathcal{C}(\sigma) = \{\sigma\}$.

   In the following, we write $\sigma \sim \triangleleft \neg \sigma^* \triangleright$ to denote that σ^* is not a prefix of σ.

Prior to the definition of $\eta[\sigma^* \leftarrow \beta]$ in terms of $\mathcal{C}$, we need to discuss some properties of $\mathcal{C}$. Note that, if σ' is not of the form $\triangleleft \alpha\, \beta \triangleright$, we have $\mathcal{C}(\sigma') \cap \mathcal{C}(\sigma'') = \emptyset$ whenever $\sigma' \neq \sigma''$. In fact, the following lemma holds by definition of $\mathcal{C}$.

Lemma 6. *$\mathcal{C}(\sigma') \cap \mathcal{C}(\sigma'') \neq \emptyset$ implies $\sigma' \sim \triangleleft \alpha \triangleright$ and $\sigma'' = \sigma \xrightarrow{\beta} (s'^{\beta}, s_n^{\neg \beta})$ for some s'^{β}. (Note that $\sigma'' \sim \triangleleft \alpha\, \beta \triangleright$.) For such σ', σ'', we have $\mathcal{C}(\sigma') \cap \mathcal{C}(\sigma'') = \mathcal{C}(\sigma'')$.*

So far, we have defined $\eta[\sigma^* \leftarrow \beta](\sigma^*) = \beta$ (note that in Fig. 4, the rightmost tree has the action β after σ^*) and $\eta[\sigma^* \leftarrow \beta](\sigma) = \eta(\sigma)$ for all σ such that σ^* is not a prefix of σ. We define $\eta[\sigma^* \leftarrow \beta](\sigma^* \xrightarrow{\beta} s') = \eta(\sigma^*)$ (note that, in Fig. 4, the rightmost tree has the action α_1 after β in all the states). For the other paths σ having σ^* as a prefix, we define $\eta[\sigma^* \leftarrow \beta](\sigma)$ depending on the amount of σ' such that $\sigma \in \mathcal{C}(\sigma')$. According to Lemma 6, we have three possible cases: **(1)** We have just one such a σ'. In this case, $\eta[\sigma^* \leftarrow \beta](\sigma) = \eta(\sigma')$. **(2)** We have two such σ'. Let σ'' be the path of the form $\triangleleft \alpha\, \beta \triangleright$ such that $\sigma \in \mathcal{C}(\sigma'')$. We let $\eta[\sigma^* \leftarrow \beta](\sigma) = \eta(\sigma'')$. As an example, let σ be the path in Fig. 4 whose actions are $\beta, \alpha_1, \cdots, \alpha_n$. For this σ, σ'' is the path whose actions are $\alpha_1, \cdots, \alpha_n, \beta$ (note that the path σ' whose actions are $\alpha_1, \cdots, \alpha_n$ also maps to σ, but $\eta(\sigma') = \beta$, and β was already executed in σ). **(3)** We have no such σ'. In this case, because of the previous cases, $\eta[\sigma^* \leftarrow \beta](\sigma) = 0$. Since we will use this property later, we state is as a lemma.

Lemma 7. *If $\Pr^{\eta[\sigma^* \leftarrow \beta]}([\sigma]) > 0$ then there exists σ' such that $\sigma \in \mathcal{C}(\sigma')$ and $\Pr^{\eta}([\sigma']) > 0$.* [2]

Proof. If σ^* is not a prefix of σ or $\sigma^* = \sigma$, we have $\Pr^{\eta}([\sigma]) = \Pr^{\eta[\sigma^* \leftarrow \beta]}([\sigma])$. So, in this case, we take $\sigma' = \sigma$.

If σ^* is a prefix of σ, we proceed by induction on $\mathrm{len}(\sigma) - \mathrm{len}(\sigma^*)$.

If $\mathrm{len}(\sigma) - \mathrm{len}(\sigma^*) = 1$, we have $\sigma = \sigma^* \xrightarrow{\beta} s$ for some s (because $\Pr^{\eta[\sigma^* \leftarrow \beta]}(\sigma) > 0$ and $\eta[\sigma^* \leftarrow \beta](\sigma^*) = \beta$). In this case, we can take $\sigma' = \sigma^*$.

If $\mathrm{len}(\sigma) - \mathrm{len}(\sigma^*) = n + 1$, let's drop the last state (and the last action) from σ to obtain the path $\sigma{\downarrow_{-1}}$. Let γ and s' be the last action and the last state in σ (resp.). So, $\sigma{\downarrow_{-1}} \xrightarrow{\gamma} s' = \sigma$. Since $\Pr^{\eta[\sigma^* \leftarrow \beta]}(\sigma) > 0$, we have that $\Pr^{\eta[\sigma^* \leftarrow \beta]}(\sigma{\downarrow_{-1}}) > 0$. By inductive hypothesis, there exists at least one σ'_{-1} such that $\sigma{\downarrow_{-1}} \in \mathcal{C}(\sigma'_{-1})$ and $\Pr^{\eta}([\sigma'_{-1}]) > 0$.

If case σ'_{-1} is of the form $\triangleleft \alpha\, \beta\, \gamma \triangleright$, by definition of $\eta[\sigma^* \leftarrow \beta]$ we have $\eta[\sigma^* \leftarrow \beta](\sigma{\downarrow_{-1}}) = \eta(\sigma'_{-1})$ (note that Lemma 6 ensures that σ'_{-1} is uniquely determined, since $\sigma'_{-1} \sim \triangleleft \alpha\, \beta\, \gamma \triangleright$). We know that $\eta[\sigma^* \leftarrow \beta](\sigma{\downarrow_{-1}}) = \gamma$. So, $\eta(\sigma'_{-1}) = \gamma$ and the path $\sigma'_{-1} \xrightarrow{\gamma} s'$ has probability greater than 0 in $\eta_{\sim \triangleleft \alpha\, \beta\, \gamma \triangleright}$. Taking into account the definition of $\mathcal{C}$ for the paths of the form $\triangleleft \alpha\, \beta\, \gamma \triangleright$ and the fact that $\sigma{\downarrow_{-1}} \in \mathcal{C}(\sigma'_{-1})$, we have $\sigma = \sigma{\downarrow_{-1}} \xrightarrow{\gamma} s' \in \mathcal{C}(\sigma'_{-1} \xrightarrow{\gamma} s')$. So, we can take $\sigma' = \sigma'_{-1} \xrightarrow{\gamma} s$.

In case σ'_{-1} is of the form $\triangleleft \alpha\, \beta \triangleright$, we have $\eta[\sigma^* \leftarrow \beta](\sigma{\downarrow_{-1}}) = \eta(\sigma'_{-1})$ (even if there are another path such σ_p such that $\sigma{\downarrow_{-1}} \in \mathcal{C}(\sigma_p)$, the definition of $\eta[\sigma^* \leftarrow \beta]$ uses σ'_{-1}). Again, $\eta(\sigma'_{-1}) = \gamma$ and the path $\sigma'_{-1} \xrightarrow{\gamma} s'$ has probability greater than 0 in η. Note that $\sigma'_{-1} \xrightarrow{\gamma} s' \sim \triangleleft \alpha\, \beta\, \gamma \triangleright$. So, we are in the same case as before and we can take $\sigma' = \sigma'_{-1} \xrightarrow{\gamma} s'$.

In the remaining case, σ'_{-1} is uniquely determined and is of the form $\triangleleft \alpha \triangleright$. Again, we have $\eta[\sigma^* \leftarrow \beta](\sigma{\downarrow_{-1}}) = \eta(\sigma'_{-1})$. So, $\eta(\sigma'_{-1}) = \gamma$ and the path $\sigma'_{-1} \xrightarrow{\gamma} s'$ has probability greater than 0 in η. If $\gamma \notin \mathsf{ample}(s)$, then $\sigma'_{-1} \xrightarrow{\gamma} s'$ is of the form $\triangleleft \alpha \triangleright$ and so $\sigma \in \mathcal{C}(\sigma'_{-1} \xrightarrow{\gamma} s')$, since $\mathcal{C}(\sigma'_{-1} \xrightarrow{\gamma} s')$ is obtained by appending $\xrightarrow{\gamma} s'$ to all the paths in $\mathcal{C}(\sigma'_{-1})$. So, if $\gamma \notin \mathsf{ample}(s)$, we can take

[2] Note that we use $\eta[\sigma^* \leftarrow \beta]$ as a scheduler without having proven that it is indeed a valid scheduler. We do so in order to avoid new definitions for probabilities, etc. However, in our proof we use only the "valid" inputs for the schedulers. An alternative construction could be done by defining the set of all the schedulers complying with the cases above. In this construction, our proof implies that all the schedulers in the set are equivalent, since they differ only in the paths having probability 0. So, any scheduler in the set can be taken to be $\eta[\sigma^* \leftarrow \beta]$.

$\sigma' = \sigma'_{-1} \xrightarrow{\gamma} s'$. If $\gamma \in \mathsf{ample}(s)$, then $\gamma = \beta$ and σ is of the form $\triangleleft\alpha\,\beta\triangleright$. In this case, $\sigma \in \mathcal{C}(\sigma'_{-1})$. So, we can take $\sigma' = \sigma'_{-1}$.

Note that, during the proof, we only evaluate the scheduler $\eta[\sigma^* \leftarrow \beta]$ in the path $\sigma{\downarrow_{-1}}$. The scheduler is well-defined for $\sigma{\downarrow_{-1}}$ because of the existence of σ'_{-1} (which, in turn, is ensured by the inductive hypothesis). □

Before we present the condition on the class $\mathcal{S}$, we need an auxiliar definition. Given a scheduler η and a path σ, we define $\mathrm{post}(\eta, \sigma)$ as the scheduler such that $\mathrm{post}(\eta, \sigma)(\sigma') = \eta(\sigma \cdot \sigma')$, for every σ' such that $\mathrm{last}(\sigma) = \mathrm{first}(\sigma')$, where $\sigma \cdot \sigma'$ denotes the concatenation of σ and σ'. Note that $\mathrm{post}(\eta, \sigma)$ is a scheduler for the system M' that coincides for M excepting for the initial state (the initial state of P' being $\mathrm{last}(\sigma)$).

Definition 4. A class $\mathcal{S}$ of deterministic schedulers is *valid for POR* iff for all $\eta \in \mathcal{S}$, σ^*, β such that $\eta[\sigma^* \leftarrow \beta]$ is defined, we have $\mathrm{post}(\eta[\sigma^* \leftarrow \beta], \sigma^* \xrightarrow{\beta} s) \in \mathcal{S}$ for all s.

The following lemma can be proven using the definition of $\eta[\sigma^* \leftarrow \beta]$.

Lemma 8. *The classes* $\mathsf{DetDist}(P)$ *and* $\mathsf{DetRSDist}(P)$ *are valid for POR.*

5.2 Proof of correctness

Theorem 8. *Let $\mathcal{S}$ be a class of schedulers valid for POR. Given M, let $\hat{M}$ be a system such that For every $\mathsf{LTL}_{\neg\,\mathrm{next}}$ formula ϕ and for every scheduler $\eta \in \mathcal{S}$ there exists a scheduler $\hat{\eta}$ for $\hat{M}$ such that $\Pr^{\eta}(\mathrm{Sat}_\phi) = \Pr^{\hat{\eta}}(\mathrm{Sat}_\phi)$.*

For the proof, we will use the following result.

Lemma 9. *Let $\{S_i\}_{i=0}^{\infty}$ be a sequence of measurable sets such that $S_{i+1} \subseteq S_i$ for all i. Then,*

$$\Pr^{\eta}(\bigcap_{i=0}^{\infty} S_i) = \lim_{n\to\infty} \Pr^{\eta}(S_n)\,.$$

Proof. Since $S_0 = \biguplus_{i=0}^{\infty}(S_i \setminus S_{i+1}) \uplus \bigcap_{i=0}^{\infty} S_i$, we have

$$\Pr^{\eta}(S_0) = \sum_{i=0}^{\infty} \Pr^{\eta}(S_i \setminus S_{i+1}) + \Pr^{\eta}(\bigcap_{i=0}^{\infty} S_i)\,. \tag{10}$$

Then, for every ϵ there exists N_ϵ such that $\sum_{i=N_\epsilon}^{\infty} \Pr^{\eta}(S_i \setminus S_{i+1}) < \epsilon$.

Equation (10), can be generalized to $S_n = \biguplus_{i=n}(S_i \setminus S_{i+1}) \uplus \bigcap_{i=0}^{\infty} S_i$ for all n. Then, for every $n > N_\epsilon$ we have

$$\Pr^{\eta}(S_n) - \Pr^{\eta}(\bigcap_{i=0}^{\infty} S_i) = \sum_{i=n}^{\infty} \Pr^{\eta}(S_i \setminus S_{i+1}) < \epsilon\,.$$

□

To prove the theorem, we use the same argument as in [1]. The proof in [1] relies on the concept of *cylinder*. Given n labels $\ell_i \in 2^{\mathsf{AP}}$ such that $\ell_i \neq \ell_{i+1}$, $Cyl(\ell_1^+, \cdots, \ell_n^+)$ is defined as the set of infinite paths ρ such that ρ has a finite prefix σ such that $\mathsf{trace}(\sigma) = \ell_1 \ldots \ell_1 \cdots \ell_n \ldots \ell_n$ (each label must occur at least once in σ).

As in [1], we show how to construct schedulers η_i such that $\eta_i(\sigma) \in \mathsf{ample}(\mathrm{last}(\sigma))$ for all σ such that $\mathrm{len}(\sigma) \leq i$. In addition, $\Pr^{\eta}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta_i}(Cyl(\ell_1^+, \cdots, \ell_n^+))$, for any cylinder $Cyl(\ell_1^+, \cdots, \ell_n^+)$. We also require $\mathrm{post}(\eta_i, \sigma) \in \mathcal{S}$ for every σ s.t. $\mathrm{len}(\sigma) \leq i + 1$. Once we have constructed these schedulers, we define a scheduler $\eta^*(\sigma) = \eta_{\mathrm{len}(\sigma)}(\sigma)$. Note that $\eta^*(\sigma) \in \mathsf{ample}(\mathrm{last}(\sigma))$ for all σ, because of the corresponding property for the schedulers η_i. Using the same argument as in [1], we show that $\Pr^{\eta}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{*}(Cyl(\ell_1^+, \cdots, \ell_n^+))$ for any cylinder

$Cyl(\ell_1^+, \cdots, \ell_n^+)$. Then, by fundamental results of measure theory, we have that $\Pr^{\eta}(\mathrm{Sat}_\phi) = \Pr^{\eta^*}(\mathrm{Sat}_\phi)$, since $\Pr^{\eta}(\mathbf{L}) = \Pr^{\eta^*}(\mathbf{L})$ for any stuttering insentitive language $\mathbf{L}$.

So, the only difference with respect to the proof in [1] is the construction of the schedulers η_i.

The scheduler η_0 is simply η. The scheduler η_{i+1} is constructed using η_i. In order to construct η_{i+1}, we will construct schedulers $\eta_{i,S}$, where S is a set of paths of length $i+1$. We let $\eta_{i+1,\{\}} = \eta_i$. Given a set S of paths of length $i+1$, a scheduler $\eta_{i,S}$ and a path $\sigma^* \notin S$ s.t. $\mathrm{len}(\sigma^*) = i+1$ we will show how to construct $\eta_{i,S\cup\{\sigma^*\}}$ from $\eta_{i,S}$. Then, we take $\eta_{i+1} = \eta_{i,S_i}$, where S_i is the set of all paths of length $i+1$.

In the following, we define $\eta_{i,S\cup\{\sigma^*\}}$ and prove that

$$\Pr^{\eta_{i,S}}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(Cyl(\ell_1^+, \cdots, \ell_n^+)) \,. \tag{11}$$

If $\eta_{i,S}(\sigma^*) \in \mathsf{ample}(\mathrm{last}(\sigma^*))$, we define $\eta_{i,S\cup\{\sigma^*\}} = \eta_{i,S}$. In this case, Eqn. (11) holds trivially.

Let $s = \mathrm{last}(\sigma^*)$. First, assume that there exists a path $\sigma' = \sigma^* \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n$ such that $\Pr^{\eta_{i,S}}(\sigma') > 0$, $\alpha_k \notin \mathsf{ample}(s)$ for all k and $\eta(\sigma') = \beta$ for some $\beta \in \mathsf{ample}(s)$. Note that, since it holds that $\mathrm{post}(\eta, \sigma^*) \in \mathcal{S}$ and we require **A5***, if such a β exists, then it is unique no matter the particular σ' choosen. If no such σ' exists, we take any $\beta \in \mathsf{ample}(s)$. We define $\eta_{i,S\cup\{\sigma^*\}}(\sigma^*) = \eta[\sigma^* \leftarrow \beta]$.

Lemma 10.

$$\Pr^{\eta_{i,S}}([\sigma]) = \sum_{\sigma' \in \mathcal{C}(\sigma)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma'])$$

Proof. For brevity, we write $\alpha(s, s')$ for $\mathsf{P}(s, \alpha, s')$.

First, we consider the case in which σ is of the form $\triangleleft\alpha\,\beta\,\gamma\triangleright$. We have

$$\begin{array}{ll}
& \Pr^{\eta_{i,S}}([\sigma]) \\
= & \{\sigma \text{ is of the form } \triangleleft\alpha\,\beta\,\gamma\triangleright\} \\
& \Pr^{\eta_{i,S}}([\sigma^* \xrightarrow{\alpha_1} (s^\beta, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s^\beta, s_n^{\neg\beta}) \xrightarrow{\beta} (s'^\beta, s_n^{\neg\beta}) \xrightarrow{\gamma_1} (s_1'^\beta, s_1'^{\neg\beta}) \cdots \xrightarrow{\gamma_n} (s_n'^\beta, s_n'^{\neg\beta})]) \\
= & \{\Pr^{\eta_{i,S}}([\sigma]) > 0,\ \eta_{i,S} \text{ is deterministic, let } s_0^{\neg\beta} = s^{\neg\beta},\ s_0'^{\beta} = s'^{\beta}, \text{ and } s_0'^{\neg\beta} = s_n^{\neg\beta}\} \\
& \Pr^{\eta_{i,S}}(\sigma^*) \cdot \prod_{k=1}^{n} \alpha_k(\,(s^\beta, s_{k-1}^{\neg\beta}), (s^\beta, s_k^{\neg\beta})\,) \cdot \beta(\,(s^\beta, s_n^{\neg\beta}), (s'^\beta, s_n^{\neg\beta})\,) \\
& \qquad \cdot \prod_{k=1}^{n'} \gamma_k(\,(s_{k-1}'^\beta, s_{k-1}^{\neg\beta}), (s_k'^\beta, s_k'^{\neg\beta})\,) \\
= & \{\text{Rearrange product}\} \\
& \Pr^{\eta_{i,S}}(\sigma^*) \cdot \beta(\,(s^\beta, s_n^{\neg\beta}), (s'^\beta, s_n^{\neg\beta})\,) \cdot \prod_{k=1}^{n} \alpha_k(\,(s^\beta, s_{k-1}^{\neg\beta}), (s^\beta, s_k^{\neg\beta})\,) \\
& \qquad \cdot \prod_{k=1}^{n'} \gamma_k(\,(s_{k-1}'^\beta, s_{k-1}^{\neg\beta}), (s_k'^\beta, s_k'^{\neg\beta})\,) \\
= & \{\alpha_k((t^\beta, t^{\neg\beta}), (t^\beta, t'^{\neg\beta})) = \alpha_k((u^\beta, t^{\neg\beta}), (u^\beta, t'^{\neg\beta})) \text{ and similarly for } \beta.\ \text{Let } s_0^{\neg\beta} = s^{\neg\beta}\} \\
& \Pr^{\eta_{i,S}}(\sigma^*) \cdot \beta(\,(s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta})\,) \cdot \prod_{k=1}^{n} \alpha_k(\,(s'^\beta, s_{k-1}^{\neg\beta}), (s'^\beta, s_k^{\neg\beta})\,) \\
& \qquad \cdot \prod_{k=1}^{n'} \gamma_k(\,(s_{k-1}'^\beta, s_{k-1}^{\neg\beta}), (s_k'^\beta, s_k'^{\neg\beta})\,) \\
= & \{\text{Definition of } \mathcal{C} \text{ (recall that, in this case, } \mathcal{C}(\sigma) \text{ is a singleton set)}\} \\
& \textstyle\sum_{\sigma' \in \mathcal{C}(\sigma)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma'])
\end{array}$$

The case in which σ is of the form $\triangleleft\alpha\,\beta\triangleright$ is a particular case of the one above, with $n' = 0$.

Below, we study the case in which σ is of the form $\lhd\alpha\rhd$.

$$
\begin{array}{ll}
& \Pr^{\eta_{i,S}}([\sigma]) \\
= & \Pr^{\eta_{i,S}}([\sigma^* \xrightarrow{\alpha_1} (s^\beta, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s^\beta, s_n^{\neg\beta})]) \\
= & \Pr^{\eta_{i,S}}([\sigma^*]) \ \cdot \ \prod_{k=1}^{n} \alpha_k((s^\beta, s_{k-1}^{\neg\beta}), (s^\beta, s_k^{\neg\beta})) \\
= & \{\sum_{\{s'^\beta \mid \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta}))>0\}} \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta})) \ = \ 1\} \\
& \quad \sum_{\{s'^\beta \mid \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta}))>0\}} \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta})) \ \cdot \Pr^{\eta_{i,S}}([\sigma^*]) \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \cdot \prod_{k=1}^{n} \alpha_k((s^\beta, s_{k-1}^{\neg\beta}), (s^\beta, s_k^{\neg\beta})) \\
= & \quad \sum_{\{s'^\beta \mid \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta}))>0\}} \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta})) \ \cdot \Pr^{\eta_{i,S}}([\sigma^*]) \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \cdot \prod_{k=1}^{n} \alpha_k((s'^\beta, s_{k-1}^{\neg\beta}), (s'^\beta, s_k^{\neg\beta})) \\
= & \sum_{\{s'^\beta \mid \beta((s^\beta, s^{\neg\beta}), (s'^\beta, s^{\neg\beta}))>0\}} \Pr^{\eta_{i,S}}([\sigma^* \xrightarrow{\beta} (s'^\beta, s^{\neg\beta}) \xrightarrow{\alpha_1} (s'^\beta, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s'^\beta, s_n^{\neg\beta})]) \\
= & \sum_{\sigma' \in \mathcal{C}(\sigma)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(\sigma')
\end{array}
$$

□

In the following, we prove that $\Pr^{\eta_{i,S\cup\{\sigma^*\}}}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta_{i,S}}(Cyl(\ell_1^+, \cdots, \ell_n^+))$ for any cylinder $Cyl(\ell_1^+, \cdots, \ell_n^+)$. In the following, we write $\mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_n^+$ to denote that $\mathsf{trace}(\sigma) = \ell_1 \ldots \ell_1 \cdots \ell_n \ldots \ell_n$. Given a cylinder $Cyl(\ell_1^+, \cdots, \ell_n^+)$, for any σ such that $\mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_{n-1}^+$, we define $\mathcal{Z}(\sigma) = \{\sigma \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \xrightarrow{\alpha_3} \cdots \mid \forall_k \mathsf{L}(s_k) = \ell_n\}$. Now, we note that $Cyl(\ell_1^+, \cdots, \ell_n^+)$ can be written as a disjoint union

$$
\begin{array}{rll}
Cyl(\ell_1^+, \cdots, \ell_n^+) = & & \biguplus_{\{\sigma \mid \mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_n^+ \ell' \ \wedge \ \ell_n \neq \ell'\}} [\sigma] \\
& \biguplus & \biguplus_{\{\sigma \mid \mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_{n-1}^+\}} \mathcal{Z}(\sigma)
\end{array}
$$

We denote the set of all the sets in the disjoint union by $\mathcal{U}$. So, $Cyl(\ell_1^+, \cdots, \ell_n^+) = \biguplus_{U \in \mathcal{U}} U$. We have,

$$
\begin{array}{rll}
\Pr^{\eta}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = & & \sum_{\{\sigma \mid \mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_n^+ \ell' \ \wedge \ \ell_n \neq \ell'\}} \Pr^{\eta}([\sigma]) \\
& + & \sum_{\{\sigma \mid \mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_{n-1}^+\}} \Pr^{\eta}(\mathcal{Z}(\sigma)) \ .
\end{array} \tag{12}
$$

We prove that $\Pr^{\eta_{i,S}}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(Cyl(\ell_1^+, \cdots, \ell_n^+))$ by showing that the summands in Eqn. (12) when $\eta = \Pr^{\eta_{i,S}}(Cyl(\ell_1^+, \cdots, \ell_n^+))$ can be rearranged and grouped as to coincide with the summands in Eqn. (12). To this end, we define a mapping $\mathcal{M} : (\mathcal{U} \cup \{[\sigma] \mid \Pr^{\eta_{i,S}}([\sigma]) > 0\}) \rightharpoonup \mathcal{P}(\mathcal{U} \cup \{[\sigma]\})$ such that:

(1) $\mathcal{M}$ is defined for all $U \in \mathcal{U}$ such that $\Pr^{\eta_{i,S}}(U) > 0$ and for every $[\sigma]$ such that $\Pr^{\eta_{i,S}}([\sigma]) > 0$ and

(2) $\Pr^{\eta_{i,S}}(U) \neq 0 \wedge \Pr^{\eta_{i,S}}(U') \neq 0 \wedge U' \in \mathcal{U} \wedge U' \in \mathcal{U} \wedge \wedge \mathcal{M}(U) \cap \mathcal{M}(U') \neq \emptyset \implies U = U'$ and

(3) $\Pr^{\eta_{i,S}}(U) = \sum_{U' \in \mathcal{M}(U)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(U')$ and

(4) for all U' such that $\Pr^{\eta_{i,S\cup\{\sigma^*\}}}(U') > 0$ there exists U such that $\Pr^{\eta_{i,S}}(U) > 0$ and $U' \in \mathcal{M}(U)$.

These properties for $\mathcal{M}$ ensure that the probabilities for $\eta_{i,S}$ and $\eta_{i,S\cup\{\sigma^*\}}$ agree, since the sum in Eqn. (12) converges absolutely and, because of the properties for $\mathcal{M}$ above, we can rearrange the nonzero terms of Eqn. (12) with $\eta = \eta_{i,S\cup\{\sigma^*\}}$ to make them coincide with the nonzero terms of Eqn. (12) with $\eta = \eta_{i,S}$. With respect to the rearrangement, we only evaluate $\mathcal{M}$ in elements in $\mathcal{U}$. However, the definition of $\mathcal{M}([\sigma])$ for every $[\sigma]$ will be useful in other parts of the proof.

The mapping $\mathcal{M}$ is defined as follows:

- $\mathcal{M}([\sigma]) = \{[\sigma'] \mid \sigma' \in \mathcal{C}(\sigma)\}$ for all σ such that $\Pr^{\eta_{i,S}}(U) > 0$ and $\mathsf{trace}(\sigma)$ is of the form $\ell_1^+, \cdots, \ell_n^+ \ell'$ and $\ell_n \neq \ell'$ (Note that σ cannot be of the form $\lhd\alpha\,\beta\rhd$)

- $\mathcal{M}(\mathcal{Z}(\sigma)) = \{\mathcal{Z}(\sigma') \mid \sigma' \in \mathcal{C}(\sigma)\}$ for all σ such that $\Pr^{\eta_{i,S}}(U) > 0$ and $\mathsf{trace}(\sigma)$ is of the form $\ell_1^+, \cdots, \ell_{n-1}^+$.

We prove the properties for $\mathcal{M}$ enounced above.

(1) $\mathcal{M}$ is defined for all $U \in \mathcal{U}$ such that $\Pr^{\eta_{i,S}}(U) > 0$, since every $U \in \mathcal{U}$ is either of the form $[\sigma]$ or of the form $\mathcal{Z}(\sigma)$. If $\Pr^{\eta_{i,S}}(\mathcal{Z}(\sigma)) > 0$, then $\Pr^{\eta_{i,S}}([\sigma]) > 0$. So, the second case in the definition of $\mathcal{M}$ covers all the elements in $\mathcal{U}$ of the form $\mathcal{Z}(\sigma)$.

(2) Suppose, towards a contradiction, that $\mathcal{M}(U) \cap \mathcal{M}(U') \neq \emptyset$, $\Pr^{\eta_{i,S}}(U) \neq 0$, $\Pr^{\eta_{i,S}}(U') \neq 0$, and $U \neq U'$. Then, by definition of $\mathcal{M}$, it must be either ($U = [\sigma]$ and $U' = [\sigma']$) or ($U = \mathcal{Z}(\sigma)$ and $U' = \mathcal{Z}(\sigma')$). In the first case, $\sigma = \sigma'$ by Lemma 6, since neither σ nor σ' can be of the form $\triangleleft\alpha\,\beta\triangleright$ (because $\mathsf{trace}(\sigma) \sim \ell_1^+, \cdots, \ell_n^+\ell'$ with $\ell_n \neq \ell'$). In the second case, it must be $\mathcal{C}(\sigma) \cap \mathcal{C}(\sigma') \neq \emptyset$. By definition of $\mathcal{C}$, it is possible only if $\sigma = \sigma^* \xrightarrow{\alpha_1} (s_1^\beta, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s_n^\beta, s_n^{\neg\beta})$ and $\sigma' = \sigma \xrightarrow{\beta} (s'^\beta, s'^{\neg\beta})$ for some α_k, s_k (Of course, it may be the case that σ is the path that finishes with β). However, since $\Pr^{\eta_{i,S}}(\sigma') > 0$, we have that $\eta_{i,S}(\sigma^* \xrightarrow{\alpha_1} (s_1^\beta, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s_n^\beta, s_n^{\neg\beta})) = \beta$. However, $\mathsf{L}((s_n^\beta, s_n^{\neg\beta})) = \ell_{n-1}$ (see the definition of $\mathcal{U}$ above) and, since β is stutter, the paths of the form $\sigma^* \xrightarrow{\alpha_1} (s_1^\beta, s_1^{\neg\beta}) \cdots \xrightarrow{\alpha_n} (s_n^\beta, s_n^{\neg\beta}) \xrightarrow{\beta} s'$ with $\mathsf{L}(s') = \ell_n$ have probability 0. Then, $\Pr^{\eta_{i,S}}(U) = 0$.

(3) Next, we prove that $\Pr^{\eta_{i,S}}(U) = \sum_{U' \in \mathcal{M}(U)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(U')$.

If $U = [\sigma]$, the claim becomes Lemma 10. So, we need to prove the claim when $U = \mathcal{Z}(\sigma)$.

For all $m > \mathrm{len}(\sigma)$, let $Z_m = \{\sigma \xrightarrow{\alpha_1} s_1 \cdots \xrightarrow{\alpha_{m-\mathrm{len}(\sigma)}} s_{m-\mathrm{len}(\sigma)} \xrightarrow{\alpha_{m-\mathrm{len}(\sigma)+1}} \cdots \mid \forall_{k=1}^{m-\mathrm{len}(\sigma)} \mathsf{L}(s_k) = \ell_n\}$. Then, $U = \bigcap_{m=\mathrm{len}(\sigma)+1}^{\infty} Z_m$, which implies $\Pr^{\eta}(U) = \lim_{m\to\infty} \Pr^{\eta}(Z_m)$ for all η (by virtue of Lemma 9). Let $E_m = \{\sigma_Z \mid \sigma_Z \sim \sigma \xrightarrow{\alpha 1} s_1 \cdots \xrightarrow{\alpha_{m-\mathrm{len}(\sigma)}} s_{m-\mathrm{len}(\sigma)} \wedge \forall_{k=1}^{m-\mathrm{len}(\sigma)} \mathsf{L}(s_k) = \ell_n \wedge \Pr^{\eta_{i,S}}([\sigma_Z]) \neq 0\}$. Then,

$$\Pr^{\eta_{i,S}}(Z_m) = \sum_{\sigma_Z \in E_m} \Pr^{\eta_{i,S}}([\sigma_Z])\ . \tag{13}$$

Similarly, for $\mathcal{M}(U)$ we define $Z'_m = \biguplus_{\sigma' \in \mathcal{C}(\sigma)} \{\sigma' \xrightarrow{\alpha_1} s_1 \cdots \xrightarrow{\alpha_{m-\mathrm{len}(\sigma')}} s_{m-\mathrm{len}(\sigma')} \xrightarrow{\alpha_{m-\mathrm{len}(\sigma')+1}} \cdots \mid \forall_{k=1}^{m-\mathrm{len}(\sigma')} \mathsf{L}(s_k) = \ell_n\}$ for all $m > \mathrm{len}(\sigma)+1$. As again, $\Pr^{\eta}(\biguplus_{U' \in \mathcal{M}(U)}[U']) = \lim_{n\to\infty} \Pr^{\eta}(Z'_m)$ for all η, $E'_m = \{\sigma_{Z'} \mid \sigma_{Z'} \sim \sigma' \xrightarrow{\alpha 1} s_1 \cdots \xrightarrow{\alpha_{m-\mathrm{len}(\sigma')}} s_{m-\mathrm{len}(\sigma')} \wedge \sigma' \in \mathcal{C}(\sigma_Z) \wedge \forall_{k=1}^{m-\mathrm{len}(\sigma')} \mathsf{L}(s_k) = \ell_n \wedge \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}]) \neq 0\}$ (note that all the paths in E'_m have length m) and

$$\Pr^{\eta_{i,S\cup\{\sigma^*\}}}(Z'_m) = \sum_{\sigma_{Z'} \in E'_m} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}])\ . \tag{14}$$

Next, we show that

$$\sum_{\sigma_{Z'} \in E'_{m+1}} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}]) \leq \sum_{\sigma_Z \in E_m} \Pr^{\eta_{i,S}}([\sigma_{Z'}]) \leq \sum_{\sigma_{Z'} \in E'_m} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}])\ . \tag{15}$$

By Eqn. (13) and Eqn. (14), this inequality implies $\Pr^{\eta_{i,S\cup\{\sigma^*\}}}(Z'_{m+1}) \leq \Pr^{\eta_{i,S}}(Z_m) \leq \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(Z'_m)$. Since $U = \bigcap_{m=\mathrm{len}(\sigma)+1}^{\infty} Z_m$ and $\biguplus_{U' \in \mathcal{M}(U)} U' = \bigcap_{m=\mathrm{len}(\sigma)+2}^{\infty} Z'_m$, Lemma 9 implies

$$\Pr^{\eta_{i,S}}(U) = \sum_{U' \in \mathcal{M}(U)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(U')\ ,$$

which is what we want to prove.

First, we consider the inequality $\sum_{\sigma_Z \in E_m} \Pr^{\eta_{i,S}}([\sigma_{Z'}]) \leq \sum_{\sigma_Z \in E'_m} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}])$. To this end, we explore how E_m relates to E'_m. Let's take $\sigma_Z \in E_m$. If σ_Z is of the form $\triangleleft\alpha\,\beta\triangleright$ or of the form $\triangleleft\alpha\,\beta\,\gamma\triangleright$, then $\mathcal{C}(\sigma_Z) \subseteq E'_m$ (recall that, in these cases, $\mathcal{C}(\sigma_Z)$ is a singleton set). If σ_Z is of the form $\triangleleft\alpha\triangleright$, then $\mathcal{C}(\sigma_Z) \subseteq E'_{m+1}$ (since, in this case, $\mathcal{C}$ inserts β in σ_Z) and, in addition, if we drop the last state (and the last action) of a path in $\mathcal{C}(\sigma_Z)$, we obtain a path in E'_m. By dropping the last state of every path in $\bigcup_{\sigma_Z \in E_m \wedge \sigma_Z \sim \triangleleft\alpha\triangleright} \mathcal{C}(\sigma_Z)$ we obtain a set of paths $I \subseteq E'_m$ such that

$\uplus_{\sigma Z \in E_m \wedge \sigma_Z \sim \lhd\alpha\rhd} \uplus_{\sigma_{Z'} \in \mathcal{C}(\sigma_Z)} [\sigma_{Z'}] \subseteq \uplus_{\sigma_{Z'} \in I} [\sigma_{Z'}]$. In addition $I \cap \mathcal{C}(\sigma_Z) = \emptyset$ for all σ_Z of the form $\lhd\alpha\,\beta\rhd$ or $\lhd\alpha\,\beta\,\gamma\rhd$. To see this, suppose that $\mathcal{C}(\sigma_Z) = \{\sigma_{Z'}\}$ and $\sigma_{Z'} \in I$. Since $\sigma_{Z'} \in I$, we know that $\sigma_{Z'}$ is obtained by dropping the last state of a path σ'_α such that $\sigma'_\alpha \in \mathcal{C}(\sigma_\alpha)$ for some σ_α of the form $\lhd\alpha\rhd$. Let $\sigma_\alpha\downarrow_{-1}$ be σ_α without the last state. Since $\sigma'_\alpha \in \mathcal{C}(\sigma_\alpha)$, σ_α is of the form $\lhd\alpha\rhd$ and $\sigma_{Z'} = \sigma'_\alpha\downarrow_{-1}$, we have $\sigma_{Z'} \in \mathcal{C}(\sigma_\alpha\downarrow_{-1})$. In addition, since $\mathcal{C}(\sigma_Z) \cap \mathcal{C}(\sigma_\alpha\downarrow_{-1}) = \{\sigma_{Z'}\}$, it must be $\sigma_Z = \sigma_\alpha\downarrow_{-1} \xrightarrow{\beta} s'$ for some s'. However, $\eta(\sigma_\alpha\downarrow_{-1}) \notin \mathsf{ample}(s)$ (because σ_α is of the form $\lhd\alpha\rhd$). So $\Pr^{\eta_{i,S}}(\sigma_Z) = 0$, which implies $\sigma_Z \notin E_m$.

Note that, since in E_m we cannot have two paths σ_Z and $\sigma_Z \xrightarrow{\beta} s$, Lemma 6 ensures that $\mathcal{C}$ maps distinct paths in E_m to disjoint set of paths. These observations allow the following calculation:

$$
\begin{array}{ll}
 & \sum_{\sigma_Z \in E_m} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
= & \{\text{Split sum}\} \\
 & \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd\neg\sigma^*\rhd} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
 & + \quad \sum_{\sigma_Z \in E_m \wedge (\sigma_Z \sim \lhd\alpha\,\beta\rhd \vee \sigma_Z \sim \lhd\alpha\,\beta\,\gamma\rhd)} \Pr^{\eta_{i,S}}([\sigma_Z]) + \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd\alpha\rhd} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
= & \{\text{Lemma 10}\} \\
 & \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd\neg\sigma^*\rhd} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(\mathcal{C}(\sigma_Z)) \\
 & + \quad \sum_{\sigma_Z \in E_m \wedge (\sigma_Z \sim \lhd\alpha\,\beta\rhd \vee \sigma_Z \sim \lhd\alpha\,\beta\,\gamma\rhd)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\mathcal{C}(\sigma_Z)]) \\
 & + \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd\alpha\rhd} \sum_{\sigma_{Z'} \in \mathcal{C}(\sigma_Z)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}]) \\
\leq & \{\text{Existence of the set } I \text{ (see above)}\} \\
 & \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd\neg\sigma^*\rhd} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}(\mathcal{C}(\sigma_Z)) \\
 & + \quad \sum_{\sigma_{Z'} \in E'_m \,\wedge\, \exists \sigma_Z \bullet \{\sigma_{Z'}\} = \mathcal{C}(\sigma_Z)} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}]) + \sum_{\sigma_{Z'} \in I} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}]) \\
\leq & \{\text{Since } I \cap \mathcal{C}(\sigma_Z) = \emptyset \text{ for all } \sigma_Z \text{ of the form } \lhd\alpha\,\beta\rhd \text{ or } \lhd\alpha\,\beta\,\gamma\rhd \text{ (see above)}\} \\
 & \sum_{\sigma_{Z'} \in E'_m} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}])
\end{array}
$$

Next, we consider the inequation $\sum_{\sigma_Z \in E_m} \Pr^{\eta_{i,S}}([\sigma_Z]) \geq \sum_{\sigma_{Z'} \in E_{m+1}} \Pr^{\eta_{i,S\cup\{\sigma^*\}}}([\sigma_{Z'}])$. For the proof, we need to show that for all $\sigma_{Z'} \in E'_{m+1}$ either **(a)** there exists $\sigma_Z \in E_{m+1}$ such that $\{\sigma_{Z'}\} = \mathcal{C}(\sigma_Z)$ and ($\sigma_Z \sim \lhd\alpha\,\beta\rhd$ or $\sigma_Z \sim \lhd\alpha\,\beta\,\gamma\rhd$) or **(b)** there exists $\sigma_Z \in E_m$ such that $\sigma_{Z'} \in \mathcal{C}(\sigma_Z)$, $\sigma_Z \sim \lhd\alpha\rhd$ and $\eta_{i,S}(\sigma_Z)$. In order to prove this claim, note that $\Pr^{\eta_{i,S\cup\sigma^*}}(\sigma_{Z'}) \neq 0$ (since $\sigma_{Z'} \in E'_m$). So, Lemma 7 ensures the existence of some σ_Z with nonzero probability such that $\sigma_{Z'} \in \mathcal{C}(\sigma_Z)$. Since the changes introduced by $\mathcal{C}$ yield a stuttering equivalent labelling, and $\mathcal{C}$ can insert one action, we have that $\sigma_Z \in E_m \cup E_{m+1}$.

If $\sigma_Z \in E_{m+1}$, and $\sigma_{Z'} \in \mathcal{C}(\sigma_Z)$, we conclude that $\sigma_{Z'} \sim \lhd\alpha\,\beta\rhd$ or $\sigma_Z \sim \lhd\alpha\,\beta\,\gamma\rhd$, and the case **(a)** above holds.

If $\sigma_Z \in E_m$, then $\sigma_Z \sim \lhd\alpha\rhd$. If $\eta_{i,S}(\sigma_Z) = \beta$, we have a path $\sigma'' \in E_{m+1}$ of the form $\sigma_Z \xrightarrow{\beta} s'$ such that $\sigma_{Z'} \in \mathcal{C}(\sigma'')$. So, if $\eta_{i,S}(\sigma_Z) = \beta$, the case **(a)** above holds. If $\eta_{i,S}(\sigma_Z) \neq \beta$, the case **(b)** above holds.

Now, we are able to prove the inequality.

$$
\begin{array}{ll}
& \sum_{\sigma_Z \in E_m} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
\geq & \{\text{Split sum}\} \\
& \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \neg \sigma^* \rhd} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
& + \quad \sum_{\sigma_Z \in E_m \wedge (\sigma_Z \sim \lhd \alpha\, \beta \rhd \vee \sigma_Z \sim \lhd \alpha\, \beta\, \gamma \rhd)} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
& + \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \alpha \rhd \wedge \eta_{i,S}(\sigma_Z) = \beta} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
& + \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \alpha \rhd \wedge \eta_{i,S}(\sigma_Z) \neq \beta} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
\geq & \{\text{Set inclusion}\} \\
& \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \neg \sigma^* \rhd} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}(\mathcal{C}(\sigma_Z)) \\
& + \quad \sum_{\sigma_Z \in E_{m+1} \wedge \sigma_Z \sim \lhd \alpha\, \beta\, \gamma \rhd} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
& + \quad \sum_{\sigma_Z \in E_{m+1} \wedge \sigma_Z \sim \lhd \alpha\, \beta \rhd} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
& + \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \alpha \rhd \wedge \eta_{i,S}(\sigma_Z) \neq \beta} \Pr^{\eta_{i,S}}([\sigma_Z]) \\
= & \{\text{Lemma 10}\} \\
& \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \neg \sigma^* \rhd} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}(\mathcal{C}(\sigma_Z)) \\
& + \quad \sum_{\sigma_Z \in E_{m+1} \wedge (\sigma_Z \sim \lhd \alpha\, \beta \rhd \vee \sigma_Z \sim \lhd \alpha\, \beta\, \gamma \rhd)} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}([\mathcal{C}(\sigma_Z)]) \\
& + \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \alpha \rhd \wedge \eta_{i,S}(\sigma_Z) \neq \beta} \sum_{\sigma_{Z'} \in \mathcal{C}(\sigma_Z)} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}([\sigma_{Z'}]) \\
\geq & \{\text{Rewrite}\} \\
& \quad \sum_{\sigma_Z \in E_m \wedge \sigma_Z \sim \lhd \neg \sigma^* \rhd} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}(\mathcal{C}(\sigma_Z)) \\
& + \quad \sum_{\sigma_{Z'} \in E'_{m+1} \wedge \exists \sigma_Z \,\bullet\, \{\sigma_{Z'}\} = \mathcal{C}(\sigma_Z) \wedge (\sigma_Z \sim \lhd \alpha\, \beta \rhd \vee \sigma_Z \sim \lhd \alpha\, \beta\, \gamma \rhd)} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}([\sigma_{Z'}]) \\
& + \quad \sum_{\sigma_{Z'} \in E'_{m+1} \wedge \exists \sigma_Z \,\bullet\, \sigma_Z \sim \lhd \alpha \rhd \wedge \eta_{i,S}(\sigma_Z) \neq \beta} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}([\sigma_{Z'}]) \\
= & \{\text{See explanation above}\} \\
& \sum_{\sigma_{Z'} \in E'_{m+1}} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}([\sigma_{Z'}])
\end{array}
$$

So, we have proven Eqn. (15). Above, we explained that Eqn. (15) implies

$$\Pr^{\eta_{i,S}}(U) = \sum_{U' \in \mathcal{M}(U)} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}(U') \,.$$

It remains to prove the property **(4)** for $\mathcal{M}$.

(4) If $U' = [\sigma']$, the claim becomes Lemma 7.

With respect to the case in which $U' = \mathcal{Z}(\sigma')$, since $\Pr^{\eta_{i,S \cup \{\sigma^*\}}}(\sigma') > 0$, there exists σ such that $\sigma' \in \mathcal{C}(\sigma)$ and $\Pr^{\eta_{i,S}}([\sigma]) > 0$. Then, $U' \in \mathcal{M}(\mathcal{Z}(\sigma))$. Because of the property **(3)** for $\mathcal{M}$ proven above, we have the inequality

$$0 < \Pr^{\eta_{i,S \cup \{\sigma^*\}}}(U') \leq \sum_{U'' \in \mathcal{C}(\mathcal{Z}(\sigma))} \Pr^{\eta_{i,S \cup \{\sigma^*\}}}(U'') = \Pr^{\eta_{i,S}}(\mathcal{Z}(\sigma)) \,.$$

So, we can take $U = \mathcal{Z}(\sigma)$.

We have proven that $\Pr^{\eta_{i,S}}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta_{i,S \cup \{\sigma^*\}}}$ This is the inductive step that allows to conclude that, for all i, there exists a scheduler η_i such that **(1)** $\Pr^{\eta_i}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta}(Cyl(\ell_1^+, \cdots, \ell_n^+))$ and **(2)** $\eta_i(\sigma) \in \mathsf{ample}(\mathrm{last}(\sigma))$ for all σ such that $\mathrm{len}(\sigma) \leq i$.

As pointed out in [1], the properties for η_i we proved are not sufficient to conclude that $\Pr^{\eta^*}(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta}(Cyl(\ell_1^+, \cdots, \ell_n^+))$. Here, we simply recall the arguments in [1]. Consider the atoms in Fig. 5 and the formula $\phi = \mathtt{true}\,\mathsf{U}\,\mathtt{smile}$, where the atomic proposition `smile` holds iff the atom A is in the "smiling" state. If $\eta(s_{\mathsf{init}}) = \alpha$ (recall that init is the initial state) and $\mathsf{ample}(s_{\mathrm{Init}}) = \{\beta\}$, then the scheduler η^* chooses β for all the paths, and so $\Pr^{\eta}(\mathrm{Sat}_\phi) = 1$, while $\Pr^{\eta^*}(\mathrm{Sat}_\phi) = 0$. However, the ample set $\{\beta\}$ violates condition (**A4**), since init and α form an end component and β is enabled in init. Note that, in this case, the path $\mathsf{init} \xrightarrow{\beta} \mathsf{init} \xrightarrow{\beta} \cdots$ has positive probability in η^*.

We explain the case in which (**A4**) holds using Fig. 6. For the atoms in the figure, $\{\beta\}$ is a valid ample set for init. Suppose that $\alpha \in \mathsf{ample}(s')$, where s' is the state reached when β does not lead to init. Again, consider a scheduler such that $\eta(\mathsf{init}) = \alpha$. So, for the path σ such that

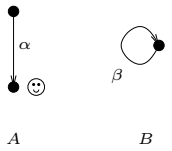


Figure 5: Example showing the need for (**A4**).

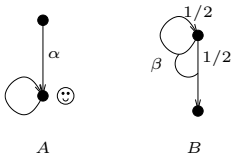


Figure 6: Another example showing the need for (**A4**)

$\Pr^\eta(\sigma) = 1$ we have the paths $\sigma_1^* = \mathsf{init} \xrightarrow{\beta} s' \xrightarrow{\alpha} \mathtt{smile}$, $\sigma_2^* = \mathsf{init} \xrightarrow{\beta} \mathsf{init} \xrightarrow{\beta} s' \xrightarrow{\alpha} \mathtt{smile}, \cdots$ and $\sum_{i=1}^{\infty} \Pr^{\eta^*}([\sigma_i^*]) = \Pr^\eta(\sigma)$. In addition, $\Pr^{\eta^*}(\mathsf{init} \xrightarrow{\beta} \mathsf{init} \xrightarrow{\beta} \cdots) = 0$.

In general, (**A4**) ensures that all the infinite paths in η^* "ending" in an end component have corresponding paths in η. It is a well-known result that the set of paths that do not "end" in an end component has probability 0 (see [1]). This ensures that, if (**A4**) holds, we have $\Pr^\eta(Cyl(\ell_1^+, \cdots, \ell_n^+)) = \Pr^{\eta^*}(Cyl(\ell_1^+, \cdots, \ell_n^+))$ for any cylinder $Cyl(\ell_1^+, \cdots, \ell_n^+)$, which in turn implies

$$\Pr^\eta(\mathrm{Sat}_\phi) = \Pr^{\eta^*}(\mathrm{Sat}_\phi)$$

for any $\mathsf{LTL}_{\neg\,\mathrm{next}}$ formula ϕ.

5.3 Theorem 2: Distributed schedulers

Let $\mathsf{Gen}(A)$ be the set of all actions generated by atom A, that is, all actions of the form $(g_A, \cdots)$. For every atom A such that $\beta, \beta' \in \mathsf{ample}(s)$ and $\beta, \beta' \in \mathsf{Gen}(A)$, all the actions α_i, α_i' as in **A5***, we know that the atoms involved in β, β' are neither involved in α_i nor in α_i' (otherwise, α_i –or α_i'– should be dependent on $\mathsf{ample}(s)$ for some i). So, $\sigma s \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n[B] = \sigma s \xrightarrow{\alpha_1'} s_1' \xrightarrow{\alpha_2'} s_2' \cdots \xrightarrow{\alpha_{n'}'} s_{n'}'[B]$ for all $B \in \mathsf{Inv}(\delta)$.

So, if $\eta \in \mathsf{DetDist}(P)$ we have $\eta(\sigma s \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \cdots \xrightarrow{\alpha_n} s_n) = \beta$ implies $\eta(\sigma s \xrightarrow{\alpha_1'} s_1' \xrightarrow{\alpha_2'} s_2' \cdots \xrightarrow{\alpha_{n'}'} s_{n'}') = \beta$.

Then, in case $\mathcal{S} = \mathsf{DetDist}(P)$, **A5**$'$ implies **A5***.

So, Theorem 4, Lemma 8 and Theorem 8 imply Theorem 2.

5.4 Theorem 3: Strongly distributed schedulers

Let A (A', resp.) be the atom that generates β (β', resp.) in **A5***. Suppose that the scheduler η in **A5*** is reinforced strongly distributed. Then A and A' are not involved in actions α_i or α_i'. So, the portions visible to A and A' of the path in which β is chosen agree with the portions of the path in which β' is chosen. Since the scheduler is reinforced strongly distributed and deterministic, $A = A'$. In addition, since the input/output schedulers in η are evaluated in the same projection in both paths, the same argument as in the case of distributed schedulers allows to conclude that $\beta = \beta'$.

Since $\mathsf{SDist}(P) \subseteq \mathsf{RSDist}(P)$, Theorem 3 is implied by Theorem 5, Lemma 8 and Theorem 8.

5.5 The notion of independence in previous works

We presented our results using a structural notion of independence. In [1], independence is defined as follows:

Two actions α, β are independent iff for all states $s \in \mathsf{S}$ with $\{\alpha, \beta\} \subseteq \mathsf{Actions}(s)$ we have: **(1)** $\mathsf{P}(s, \alpha, t) > 0$ implies $\beta \in \mathsf{Actions}(t)$, **(2)** $\mathsf{P}(s, \beta, u) > 0$ implies $\alpha \in \mathsf{Actions}(u)$, **(3)** for all states $w \in S$: $\sum_{t \in S} \mathsf{P}(s, \alpha, t) \cdot \mathsf{P}(t, \beta, w) = \sum_{u \in S} \mathsf{P}(s, \beta, u) \cdot \mathsf{P}(u, \alpha, w)$.

Here, to avoid confusion, we say that actions α and β complying such condition are weakly independent.

Intuitively, if $\beta \in \mathsf{ample}(s)$, suppose that an action α occurs before β in some path σ. Then, if α and β are weakly independent, it means that the probabilistic behaviour is not modified if we exchange the order of α and β. However, if these actions are weakly independent but not independent, α may give information to the atom B that outputs β, and so it may be the case that B does not execute β in another path starting from s. In this case, β cannot be moved right after s. So, we must prevent B to choose another action β. This is the fact that motivates **A5** in in [1, 10].

Let **A3′** be the restriction **A3** substituting "weakly dependent" for "dependent" (so, **A1**, **A2**, **A3′**, **A4** are the same restrictions as in [1, 10]). For a given s, if there exists a path $s \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_n} s_n \xrightarrow{\gamma} s_{n+1}$, where some of the α_i is not independent of $\mathsf{ample}(s)$ and γ is probabilistic (note that, by **A3′**, they are required to be weakly independent), in this case we must require the same restrictions as in [1, 10]. Otherwise, we can require the restrictions in Theorem 2 or Theorem 3 (according, of course, to the class of schedulers under consideration).

So, this criterion allows more reductions than the ones in [1, 10].

5.6 The need for A5′ under distributed schedulers

The introductory example of Fig. 1 clearly shows that **A5** is unnecesarily strong if we only want to preserve the probabilities of properties under the class of distributed schedulers. In the following, we discuss the need of **A5′** for POR under distributed schedulers.

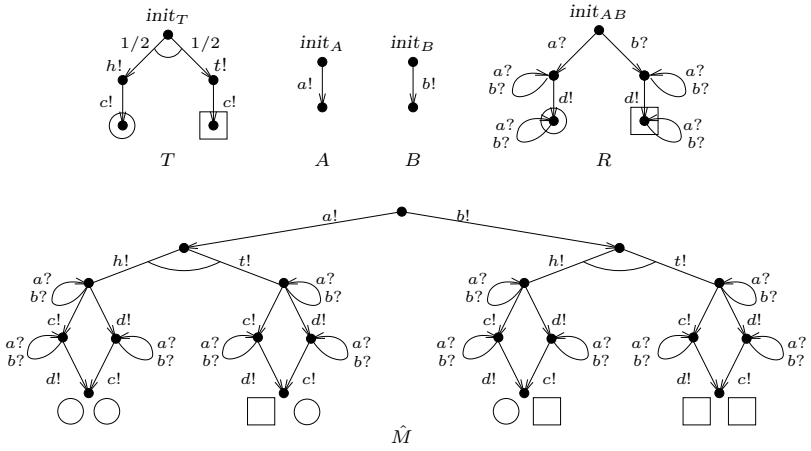


Figure 7: Distributed schedulers vs strongly distributed schedulers.

Consider atoms T, A, B and R in Fig. 7. Intuitively, atom R "remembers", the order in which $a!$ and $b!$ were executed. We aim to maximize the probability to reach a state in which either T and R are both in a circle, or T and R are both in a square (in the following, we say that they reach "an agreement"). That is, we would like to know the maximum probability that R guesses the outcome in T by only inspecting atoms A and B.

Notice that it is possible to find a distributed scheduler such that the composed system reaches an agreement with probability 1. Such scheduler lets T execute first and then selects atom A or

B according the outcome of T. This is the same situation that motivated the introduction of strongly distributed schedulers (see explanation of Fig. 3) since atom T is completely independent of all other components.

Condition **A5′** only admits two possible ample sets at the initial state: (1) the set of all actions enabled at initial state, and (2) the singleton containing action $(\frac{1}{2} \xrightarrow{h!} + \frac{1}{2} \xrightarrow{t!})$ of T. Both of them yield maximum probability of 1 in the reduced system since they contain the scheduler described above.

Of course, this maximum probability value is still unrealistic if we we consider strongly distributed schedulers. Notice that the proposed scheduler, though distributed, it is not strongly distributed. So a POR under the class of strongly distributed schedulers may allow to cut away such beheviour. Therefore, it drops **A5′** and allows also the ample set $\{a, b\}$ at the initial state. Notice that this choice is not possible under **A5′** since a and b are generated by two different atoms (and besides they are *not* independent). A possible reduction with the choice of this ample set is depicted in Fig. 7. Precisely this reduction yields a maximum probability of $\frac{1}{2}$ of reaching an agreement.

6 Experimental results

6.1 Dining philosophers

We translated the dining philosophers model in [3] to the PRISM language. We used a program in order to obtain a model for N philosophers given a single philosopher. Another program generates a reduced version. The reduced version is obtained by adding extra conditions to the guards. Such conditions allow only the actions in the ample set. The calculation of the ample sets is not done by the program, since the program simply add the conditions according to ample sets we deduced.

An example of the deductions we used to discover ample sets is the following: suppose that the philosophers are $A_1 \cdots A_N$, and suppose this is the order in which they are arranged (that is: A_1 is at the left of A_2, A_2 is at left of A_3 and A_N is at the left of A_1). Suppose that philosopher A_i has its left fork and philosopher A_j has its right fork, with $i < j$. The only actions of the philosophers in the set $\{A_1 \cdots A_{i-1} A_{j+1} A_N\}$ that are dependent on the actions of the philosophers in the set $\{A_i \cdots A_j\}$ are the actions in which A_{i-1} or A_{j+1} takes or leaves its right fork (in the case of A_{i-1}) or its left fork (in the case of A_{j+1}). However, since these forks are being held by A_i and A_j, these actions cannot be executed, and so the set $\{A_i \cdots A_j\}$ complies with restriction **A3**.

The ample described above can be used in several states. Note that this ample set is not possible using the current techniques, since, given that **A5** is hard to calculate, the ample sets have either a single action or all the actions when reduced according to [1, 10] (see [2]).

In `cs.famaf.unc.edu.ar/~sgiro/diningPhil.tar.gz` the following files are available:

- `preprocBaier`: program to generate the full model. It takes as arguments the name of the model for a single philosopher and the amount of philosophers
- `preprocRedBaier`: program to generate the reduced model. It takes the same arguments as `preprocBaier`
- `philModelBaier.nm`: model for a single philosopher
- `stopGenEnsure.ml`: program used by `preprocBaier` and `preprocRedBaier`. It must be compiled using `ocamlc`

6.2 Anonymous fair service

A server must serve two clients in a fair fashion regardless of the rates at which they ask for service. In addition, the clients cannot be identified, so the server cannot simply count how much times it has served each of the clients. The protocol to solve the problem is the following: the server keeps track of the order in which requests were received. At most two requests may be pending,

since we assume that clients cannot perform requests while waiting [3]. So, once two requests were received, a coin is tossed in order to decide which of the requests is replied: in case the coin lands heads, the first request is replied. Otherwise, the server replies the second request. Then, the coin is tossed again.

A model is available at `cs.famaf.unc.edu.ar/~sgiro/fair_server.tar.gz`. The file also contains a reduced model `fair_server_red.nm`. The command `prism filename fair_server.pctl` can be used to check the probabilities p_m.

References

[1] C. Baier, M. Größer, and F. Ciesinski. Partial order reduction for probabilistic systems. In *QEST '04*, pages 230–239, Washington, DC, USA, 2004. IEEE CS.

[2] C. Baier, M. Größer, and F. Ciesinski. Quantitative analysis of distributed randomized protocols. In *Proc. of FMICS'05*, pages 2–7. ACM, 2005.

[3] Christel Baier, Frank Ciesinski, and Marcus Größer. Probmela and verification of markov decision processes. *SIGMETRICS Perform. Eval. Rev.*, 32(4):22–27, 2005.

[4] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proc. of FSTTCS 95*, LNCS 1026, pages 288–299. Springer, 1995.

[5] A. Cassandra. The POMDP page. `www.pomdp.org`.

[6] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *J. Cryptology*, 1(1):65–75, 1988.

[7] L. Cheung. *Reconciling Nondeterministic and Probabilistic Choices.* PhD thesis, Radboud Universiteit Nijmegen, 2006.

[8] F. Ciesinski and C. Baier. LiQuor: A tool for qualitative and quantitative linear time analysis of reactive systems. In *QEST'06*, pages 131–132. IEEE CS, 2006.

[9] E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking.* MIT Press, 2000.

[10] P. R. D'Argenio and P. Niebert. Partial order reduction on concurrent probabilistic programs. In *QEST '04*, pages 240–249, Washington, DC, USA, 2004. IEEE CS.

[11] L. de Alfaro. The verification of probabilistic systems under memoryless partial-information policies is hard. In *PROBMIV'99. TR CSR-99-8*, pages 19–32. University of Birmingham, 1999.

[12] L. de Alfaro, T. A. Henzinger, and R. Jhala. Compositional methods for probabilistic systems. In *CONCUR'01*, LNCS 2154, pages 351–365. Springer, 2001.

[13] S. Giro and P. R. D'Argenio. Quantitative model checking revisited: neither decidable nor approximable. In *FORMATS'07*, LNCS 4763, pages 179–194. Springer, 2007.

[14] S. Giro and Pedro R. D'Argenio. On the verification of probabilistic i/o automata with unspecified rates. To appear. Available at `cs.famaf.unc.edu.ar/~sgiro/GD09-sac.pdf`.

[15] R.J. van Glabbeek, S.A. Smolka, and B. Steffen. Reactive, generative, and stratified models of probabilistic processes. *Information and Computation*, 121:59–80, 1995.

[16] P. Godefroid. *Partial-Order Methods for the Verification of Concurrent Systems - An Approach to the State-Explosion Problem.* LNCS 1032. Springer, 1996.

[3] Otherwise, it is impossible to guarantee fairness, since one of the entities may perform requests at an arbitrarily high rate.

[17] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In *Proc. of TACAS'06*, LNCS 3920, pages 441–444. Springer, 2006.

[18] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artif. Intell.*, 147(1-2):5–34, 2003.

[19] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, Laboratory for Computer Science, MIT, 1995.

[20] Stavros Tripakis. Undecidable problems of decentralized observation and control. In *Proc. of the 40th IEEE Conference on Decision and Control*, volume 5, pages 4104–4109, 2001.

[21] Noel Vaillant. `probability.net`. Probability tutorials on line. Tutorial 2.