Distributed Markov Chains (corrected version)

Ratul Saha[1], Javier Esparza[2], Sumit Kumar Jha[3⋆], Madhavan Mukund[4⋆⋆], and P. S. Thiagarajan[1⋆⋆⋆]

[1] National University of Singapore, Singapore, {ratul,thiagu}@comp.nus.edu.sg
[2] Technische Universität München, esparza@in.tum.de
[3] University of Central Florida, USA, jha@eecs.ucf.edu
[4] Chennai Mathematical Institute, India, madhavan@cmi.ac.in

Abstract. The formal verification of large probabilistic models is challenging. Exploiting the concurrency that is often present is one way to address this problem. Here we study a class of communicating probabilistic agents in which the synchronizations determine the probability distribution for the next moves of the participating agents. The key property of this class is that the synchronizations are deterministic, in the sense that any two simultaneously enabled synchronizations involve disjoint sets of agents. As a result, such a network of agents can be viewed as a succinct and distributed presentation of a large global Markov chain. A rich class of Markov chains can be represented this way.

We use partial-order notions to define an interleaved semantics that can be used to efficiently verify properties of the global Markov chain represented by the network. To demonstrate this, we develop a statistical model checking (SMC) procedure and use it to verify two large networks of probabilistic agents.

1 Introduction

1.1 NOTE

This is a corrected version of the technical report that was cited in the VMCAI'2015 paper by the same authors with the title "Distributed Markov chains". Unfortunately we recently discovered a bug in the translation of DMCs into deterministic cyclic negotiations (DCNs) model. As a consequence, the main result claimed in section 8 *of the conference paper that soundness of DMCs with final states can be determined in polynomial time needs to be withdrawn. In section*

⋆ Sumit Kumar Jha acknowledges support from the National Science Foundation under projects CCF-1438989 and CCF-1422257, Air Force Research Lab under contract #CA0116UCF2013, and the Oak Ridge National Laboratory under contract #4000126570.

⋆⋆ Partially supported by a grant from Infosys Foundation.

⋆⋆⋆ P. S. Thiagarajan acknowledges support from the Singapore Ministry of Education grant T1 251RES1115.

8 *of this report we sketch briefly the root cause of this gap between DMCs and DCNs.*

We present here a class of distributed probabilistic systems called distributed Markov chains (DMCs). A DMC is a network of probabilistic transition systems that synchronize on common actions. The information that the agents gain through a synchronization determines the probability distribution for their next moves. Internal actions correspond to synchronizations involving only one agent. The synchronizations are deterministic in the sense that, at any global state, if two synchronizations are enabled then they will involve disjoint set of agents. We capture this syntactically by requiring that at a local state of an agent, the synchronizations that the agent is willing to engage in will all involve the same set of partners. In many distributed probabilistic systems, the communication protocols are naturally deterministic in this sense, or can be designed to be so. As our two case studies in Section 7 show, the determinacy restriction is less limiting than may appear at first sight while permitting a considerable degree of concurrency.

We define an interleaved semantics where one synchronization action is executed at a time, followed by a probabilistic move by the participating agents. Except in the trivial case where there is no concurrency, the resulting transition system will *not* be a Markov chain. Hence, defining a probability measure over interleaved runs, called *trajectories*, is a technical challenge. We address this by noting that there is a natural independence relation on local actions—two actions are independent if they involve disjoint sets of agents. Using this relation, we partition the trajectories in the usual way into equivalence classes that correspond to partially ordered executions. We then use the maximal equivalence classes to form a trajectory space that is a counterpart to the path space of a Markov chain.

To endow this trajectory space with a probability measure, we exploit the fact that, due to determinacy of synchronizations, any two actions enabled at a global state will be independent. Hence, by executing all the enabled actions simultaneously, followed by probabilistic moves by all the agents involved, one obtains a finite state Markov chain that captures the global behavior of the DMC under this "maximally parallel" execution semantics.

Using Mazurkiewicz trace theory [7], we then embed the trajectory space into the path space of this Markov chain and use this embedding to induce a probability measure over the trajectory space. Consequently, the global behavior of this Markov chain can be verified efficiently using the interleaved semantics.

We then demonstrate that the DMC model possesses a good of modeling power while considerably easing the task of analyzing the global behavior of the network. We formulate a statistical model checking (SMC) procedure for DMCs in which the specifications consist of Boolean combinations of local bounded linear temporal logic (BLTL) [3] formulas. We then use the sequential probability ratio test (SPRT) based SMC technique [19, 20] to analyze the global behavior of a DMC.

Our two case studies show that one can easily construct DMC models of a variety of distributed probabilistic systems [6]. Both the systems we study exhibit a considerable degree of concurrency. Further, the performance and scalability of our interleaved semantics based verification techniques is significantly better than the SMC procedure of PLASMA [5].

To summarize, our main contributions are: (i) establishing that deterministic synchronizations are a fruitful restriction for distributed stochastic systems, (ii) showing that the space of partially ordered runs of such systems can be endowed with a probability measure due to the clean combination of concurrent and stochastic dynamics, (iii) constructing an SMC procedure in this distributed stochastic setting.

Related work Our work is in line with partial order based methods for Markov Decision Processes (MDPs) [11] where, typically, a partial commutation structure is imposed on the actions of a *global* MDP. For instance, in [4], partial order reduction is used to identify "spurious" nondeterminism arising out of the interleaving of concurrent actions, in order to determine when the underlying behavior corresponds to a Markov chain. In contrast, in a DMC, deterministic communication ensures that local behaviors always generate a global Markov chain. The independence of actions is directly given by the local state spaces of the components. This also makes it easier to model how components influence each other through communications.

The interplay between concurrency and stochasticity has also been explored in the setting of event structures [1,18]. In these approaches, the global behaviour — which is not a Markov chain — is endowed with a probability measure. Further, probabilistic verification problems are not formulated and studied. Markov nets, studied in [2] can be easily modeled as DMCs. However, in [2], the focus is on working out a probabilistic event structure semantics rather than on developing a model checking procedure based on the interleaved semantics, as we do here.

Our model is formulated as a sub-class of probabilistic asynchronous automata [13], where we require synchronizations to be deterministic. This restriction allows us to develop a probability measure over the (infinite) trajectory space, which in turn paves the way for carrying out formal verification based on probabilistic temporal logic specifications. In contrast, the work reported in [13] is language-theoretic, with the goal of generalizing Zielonka's theorem [21] to a probabilistic setting. Moreover, in the model of [13], conflicting actions may be enabled at a global state and it is difficult to see how one can formulate a σ-algebra over the runs with a well-defined probability measure.

2 The Distributed Markov Chain (DMC) Model

We fix n agents $\{1, 2, \ldots, n\}$ and set $[n] = \{1, 2, \ldots, n\}$. For convenience, we denote various $[n]$-indexed sets of the form $\{X_i\}_{i\in[n]}$ as just $\{X_i\}$. We begin with some notation for distributed state spaces.

Definition 1. *For $i \in [n]$, let S_i be a finite set of local states, where $\{S_i\}$ are pairwise disjoint.*

- *$S = \bigcup_i S_i$ is the set of* local states.
- *For nonempty $u \subseteq [n]$, $\boldsymbol{S}_u = \prod_{i \in u} S_i$ is the set of u-states.*
- *$\boldsymbol{S}_{[n]}$ is the set of* global states*, typically denoted $\boldsymbol{S}$.*
- *For a state $\boldsymbol{v} \in \boldsymbol{S}_u$ and $w \subseteq u$, $\boldsymbol{v}_w$ denotes the projection of $\boldsymbol{v}$ to $\boldsymbol{S}_w$.*
- *For $u = \{i\}$, we write $\boldsymbol{S}_i$ and $\boldsymbol{v}_i$ rather than $\boldsymbol{S}_{\{i\}}$ and $\boldsymbol{v}_{\{i\}}$, respectively.*

Our model is a restricted version of probabilistic asynchronous automata [13].

Definition 2. *A* probabilistic asynchronous system *is a structure $(\{S_i\}, \{s_i^{in}\}, A, loc, en, \{\pi^a\}_{a \in A})$ where:*

- *S_i is a finite set of local states for each i and $\{S_i\}$ is pairwise disjoint.*
- *$s_i^{in} \in S_i$ is the initial state of agent i.*
- *A is a set of synchronization actions.*
- *$loc : A \to 2^{[n]} \setminus \emptyset$ specifies the agents that participate in each action a.*
  - *For $a \in A$, we write $\boldsymbol{S}_a$ instead of $\boldsymbol{S}_{loc(a)}$ and call it the set of a-states.*
- *For each $a \in A$, $en_a \subseteq \boldsymbol{S}_a$ is the subset of a-states where a is enabled.*
- *With each $a \in A$, we associate a probabilistic transition function $\pi^a : en_a \to (\boldsymbol{S}_a \to [0,1])$ such that, for every $\boldsymbol{v} \in en_a$, $\sum_{\boldsymbol{u} \in \boldsymbol{S}_a} \pi^a(\boldsymbol{v})(\boldsymbol{u}) = 1$.*

The action a represents a synchronization between the agents in $loc(a)$ and it is enabled at the global state $\mathbf{s}$ if $\mathbf{s}_a \in en_a$. When a occurs at $\mathbf{s}$, only the components in $loc(a)$ are involved in the move to the new global state $\mathbf{s}'$; the new a-state $\mathbf{s}'_a$ is chosen probabilistically according to the distribution $\pi^a(\mathbf{s}_a)$. On the other hand, for every $j \notin loc(a)$, $\mathbf{s}_j = \mathbf{s}'_j$.

We would like to lift the probabilities associated with individual moves to a probability measure over the runs of the system. This is difficult to achieve because of the combination of nondeterminism, concurrency and probability in the model. This motivates us to restrict the nondeterminism in the model.

For an agent i and a local state $s \in S_i$, we define the set of actions that i can participate in at s to be $act(s) = \{a \mid i \in loc(a), s = \mathbf{v}_i \text{ for some } \mathbf{v} \in en_a\}$. Using this notion we define the DMC model as follows.

Definition 3. *A* distributed Markov chain (DMC) *is a probabilistic asynchronous system $\mathcal{D} = (\{S_i\}, \{s_i^{in}\}, A, loc, en, \{\pi^a\}_{a \in A})$ in which (i) for each local state $s \in S$, if $a, b \in act(s)$ then $loc(a) = loc(b)$, and (ii) if $a, b \in A$, $a \neq b$ and $loc(a) = loc(b)$, then $en_a \cap en_b = \emptyset$.*

By (i), the set of partners that an agent can synchronize with is fixed deterministically by its current local state. Typically an agent will be willing to engage in a *set* of actions at a local state. But by (ii), at most one of these actions will be enabled in any global state.

Events Events will play a crucial role in defining the dynamics of a DMC. Let $\mathcal{D}$ be a DMC. An *event* of $\mathcal{D}$ is a triple $e = (\mathbf{v}, a, \mathbf{v}')$ where $\mathbf{v}, \mathbf{v}' \in \mathbf{S}_a$, $\mathbf{v} \in en_a$ and $\pi^a(\mathbf{v})(\mathbf{v}') > 0$. We define $loc((\mathbf{v}, a, \mathbf{v}'))$ to be $loc(a)$.

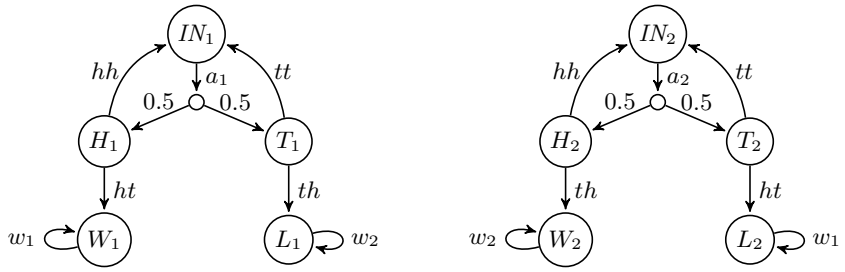

Fig. 1. The DMC model for the two players coin toss game

Suppose $e = (\mathbf{v}, a, \mathbf{v}')$ is an event and $p = \pi^a(\mathbf{v})(\mathbf{v}')$. Then e represents an occurrence of the synchronization action a followed by a joint move by the agents in $loc(a)$ from $\mathbf{v}$ to $\mathbf{v}'$ with probability p. Again, components outside $loc(e)$ are unaffected by this move.

Let Σ denote the set of events of $\mathcal{D}$ and $e, e', \ldots$ range over Σ. With the event $e = (\mathbf{v}, a, \mathbf{v}')$ we associate the probability $p_e = \pi^a(\mathbf{v})(\mathbf{v}')$.

The interleaved semantics We now associate a global transition system with $\mathcal{D}$ based on event occurrences.

Recall that $\mathbf{S}$ is the set of global states. The event $e = (\mathbf{v}, a, \mathbf{v}')$ is *enabled* at $\mathbf{s} \in \mathbf{S}$ iff $\mathbf{v} = \mathbf{s}_a \in en_a$. The transition system of $\mathcal{D}$ is $TS = (\mathbf{S}, \Sigma, \rightarrow, \mathbf{s}^{in})$, where $\mathbf{s}^{in}$ is the global initial state with $\mathbf{s}^{in}_i = s^{in}_i$ for each i. The transition relation $\rightarrow \subseteq \mathbf{S} \times (\Sigma \times (0,1]) \times \mathbf{S}$ is given by $\mathbf{s} \xrightarrow{e,p_e} \mathbf{s}'$ iff $e = (\mathbf{v}, a, \mathbf{v}')$ is enabled at $\mathbf{s}$, $\mathbf{s}'_a = \mathbf{v}'$ and $\mathbf{s}_j = \mathbf{s}'_j$ for every $j \notin loc(e)$.

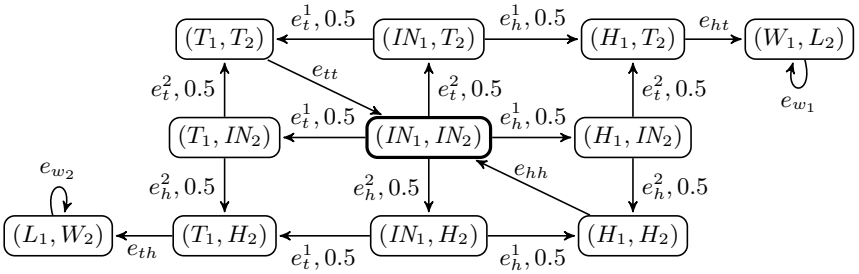

Fig. 2. The transition system of a DMC for the two player coin toss game

In Fig. 1 we show a DMC describing a simple two player game. Each player tosses an unbiased coin. If the tosses have the same outcome, the players toss again. If the outcomes are different, then the player who tossed heads wins. In this 2-component system, $S_i = \{IN_i, T_i, H_i, L_i, W_i\}$ for $i = 1, 2$, where T_i/H_i denote that a tail/head was tossed, respectively, and L_i/W_i denote local losing/winning states, respectively. Agent 1, for instance, has an internal action a_1 with $loc(a_1) = \{1\}$, $en_{a_1} = \{IN_1\}$ and $\pi^{a_1}(IN_1)(T_1) = 0.5 = \pi^{a_1}(IN_1)(H_1)$. Thus, $e^1_h = (\{IN_1\}, a_1, \{H_1\})$ and $e^1_t = (\{IN_1\}, a_1, \{T_1\})$ are both events that are

enabled at (IN_1, IN_2). On the other hand, tt is an action with $loc(tt) = \{1, 2\}$, $en_{tt} = \{(T_1, T_2)\}$. There will be an event $e_{tt} = (\{T_1, T_2\}, tt, \{IN_1, IN_2\})$ with $\pi^{tt}((T_1, T_2))((IN_1, IN_2)) = 1$. To aid readability, such an action with a unique event (with probability 1) as its only outcome is shown without any probability value. In this simple example, all the actions except a_1 and a_2 are of this type.

The trace alphabet (Σ, I) The independence relation $I \subseteq \Sigma \times \Sigma$ given by $e\ I\ e'$ iff $loc(e) \cap loc(e') = \emptyset$. Clearly I is irreflexive and symmetric and hence (Σ, I) is a Mazurkiewicz trace alphabet [7].

3 The trajectory space

Let TS be the transition system associated with a DMC $\mathcal{D}$. To reason about the probabilistic behaviour of $\mathcal{D}$ using TS, one must build a σ-algebra over the paths of this transition system and endow it with a probability measure. The major difficulty is that, due to the mix of concurrency and stochasticity, TS is not a Markov chain (except in trivial cases where there is no concurrency). In Fig. 2, for instance, the sum of the probabilities of the transitions originating from the state (IN_1, IN_2) is 2. To get around this, we first filter out concurrency by working with equivalence classes of paths.

We shall refer to paths in TS as *trajectories*. A finite *trajectory* of TS from $\mathbf{s} \in \mathbf{S}$ is a sequence of the form $\mathbf{s}_0 e_0 \mathbf{s}_1 \ldots \mathbf{s}_{k-1} e_{k-1} \mathbf{s}_k$ such that $\mathbf{s}_0 = \mathbf{s}$ and, for $0 \leq \ell < k$, $\mathbf{s}_\ell \xrightarrow{e_\ell, p_{e_\ell}} \mathbf{s}_{\ell+1}$. Infinite trajectories are defined as usual.

For the trajectory $\rho = \mathbf{s}_0 e_0 \mathbf{s}_1 \ldots \mathbf{s}_{k-1} e_{k-1} \mathbf{s}_k$, we define $ev(\rho)$ to be the event sequence $e_0 e_1 \ldots e_{k-1}$. Again, this notation is extended to infinite trajectories in the natural way. Due to concurrency, one can have infinite trajectories that are not maximal, so we proceed as follows.

Let $\Sigma_i = \{e \mid i \in loc(e)\}$. Suppose ξ is an event sequence (finite or infinite). Then $proj_i(\xi)$ is the sequence obtained by erasing from ξ all events that are not in Σ_i. This leads to the equivalence relation $\approx$ over event sequences given by $\xi \approx \xi'$ iff $proj_i(\xi) = proj_i(\xi')$ for every i. We let $[\xi]$ denote the $\approx$-equivalence class containing ξ and call it a *(Mazurkiewicz) trace.* [5] The partial order relation $\sqsubseteq$ over traces is defined as $[\xi] \sqsubseteq [\xi']$ iff $proj_i(\xi)$ is a prefix of $proj_i(\xi')$ for every i. Finally the trace $[\xi]$ is said to be maximal if for every ξ', $[\xi] \sqsubseteq [\xi']$ implies $[\xi] = [\xi']$. The trajectory ρ is *maximal* iff $[ev(\rho)]$ is a maximal trace. In the transition system of Fig. 2, $(IN_1, IN_2)e_h^1(H_1, IN_2)e_T^2(H_1, T_2)e_{ht}((W_1, L_2)e_{w_1})^\omega$ is a maximal infinite trajectory. In fact, in this example all the infinite trajectories are maximal.

The σ-algebra of trajectories We denote by $Trj_{\mathbf{s}}$ the set of maximal trajectories from $\mathbf{s}$. Two trajectories can correspond to interleavings of the same partially ordered execution of events. Hence, one must work with equivalence

[5] For infinite sequences, it is technically more convenient to define traces using equivalence of projections rather than permutation of independent actions.

classes of maximal trajectories to construct a probability measure. The equivalence relation $\simeq$ over $Trj_{\mathbf{s}}$ that we need is defined as $\rho \simeq \rho'$ if $ev(\rho) \approx ev(\rho')$. As usual $[\rho]$ will denote the equivalence class containing the trajectory ρ.

Let ρ be finite trajectory from $\mathbf{s}$. Then $\uparrow\rho$ is the subset of $Trj_{\mathbf{s}}$ satisfying $\rho' \in \uparrow\rho$ iff ρ is a prefix of ρ'. We now define $BC(\rho)$, the basic trj-cylinder at $\mathbf{s}$ generated by ρ, to be the least subset of $Trj_{\mathbf{s}}$ that contains $\uparrow\rho$ and satisfies the closure property that if $\rho' \in BC(\rho)$ and $\rho' \simeq \rho''$ then $\rho'' \in BC(\rho)$. In other words, $BC(\rho) = \{[\rho'] \mid \rho' \in Trj_{\mathbf{s}}, [ev(\rho)] \sqsubseteq [ev(\rho')]\}$.

It is worth noting that we could have $BC(\rho) \cap BC(\rho') \neq \emptyset$ without having $\rho \simeq \rho'$. For instance, in Fig. 2, let $\rho = (IN_1, IN_2)e_h^1(H_1, IN_2)$ and $\rho' = (IN_1, IN_2)e_t^2(IN_1, T_2)$. Then $BC(\rho)$ and $BC(\rho')$ will have common maximal trajectories of the form $(IN_1, IN_2)e_h^1(H_1, IN_2)e_t^2(H_1, T_2)\ldots$.

We now define $\widehat{SA}(\mathbf{s})$ to be the least σ-algebra that contains the basic trj-cylinders at $\mathbf{s}$ and is closed under countable unions and complementation (relative to $Trj_{\mathbf{s}}$).

To construct the probability measure $\widehat{P} : \widehat{SA}(\mathbf{s}) \to [0,1]$ we are after, a natural idea would be to assign a probability to each basic trj-cylinder as follows. Let $BC(\rho)$ be a basic trj-cylinder with $\rho = \mathbf{s}_0 e_0 \mathbf{s}_1 \ldots \mathbf{s}_{k-1} e_{k-1} \mathbf{s}_k$. Then $\widehat{P}(BC(\rho)) = p_0 \cdot p_1 \cdot \ldots \cdot p_{k-1}$, where $p_\ell = p_{e_\ell}$, for $0 \leq \ell < k$. This is inspired by the Markov chain case in which the probability of a basic cylinder is defined to be the product of the probabilities of the events encountered along the common finite prefix of the basic cylinder. However, showing directly that this extends uniquely to a probability measure over $\widehat{SA}_{\mathbf{s}}$ is very difficult.

We get around this by associating a Markov chain $\mathcal{M}$ with $\mathcal{D}$ and then embed $\widehat{SA}_{\mathbf{s}}$ into $SA_{\mathbf{s}}$, the σ-algebra generated by the infinite paths in $\mathcal{M}$ starting from $\mathbf{s}$. The standard probability measure over $SA_{\mathbf{s}}$ will then induce a probability measure over $\widehat{SA}_{\mathbf{s}}$.

4 The Markov chain semantics

We associate a Markov chain with a DMC using a "maximal parallelism" based semantics. A *nonempty* set of events $u \subseteq \Sigma$ is a *step* at $\mathbf{s}$ if each $e \in u$ is enabled at $\mathbf{s}$ and, for every distinct pair of events $e, e' \in u$, $e\ I\ e'$. We say u is a *maximal* step at $\mathbf{s}$ if u is a step at $\mathbf{s}$ and $u \cup \{e\}$ is not a step at $\mathbf{s}$ for any $e \notin u$. In Fig. 2, $\{e_h^1, e_h^2\}$, $\{e_h^1, e_t^2\}$, $\{e_t^1, e_h^2\}$ and $\{e_t^1, e_t^2\}$ are maximal steps at the initial state (IN_1, IN_2).

Let u be a maximal step at $\mathbf{s}$. Then $\mathbf{s}'$ is the u-successor of $\mathbf{s}$ if the following conditions are satisfied: (i) For each $e \in u$, if $e = (\mathbf{v}, a, \mathbf{v}')$ and $i \in loc(e)$ then $\mathbf{s}'_i = \mathbf{v}'_i$, and (ii) $\mathbf{s}_j = \mathbf{s}'_j$ if $j \notin loc(u)$, where $loc(u) = \bigcup_{e \in u} loc(e)$.

Suppose u is a maximal step at $\mathbf{s}$ and $i \in loc(u)$. Then, because events in a step are independent, it follows that there exists a unique $e \in u$ such that $i \in loc(e)$, so the u-successor of $\mathbf{s}$ is unique. We say $\mathbf{s}'$ is a *successor* of $\mathbf{s}$ if there exists a maximal step u at $\mathbf{s}$ such that $\mathbf{s}'$ is the u-successor of $\mathbf{s}$. From the definition of a DMC, it is easy to see that if $\mathbf{s}'$ is a successor of $\mathbf{s}$ then there

exists a unique maximal step u at $\mathbf{s}$ such that $\mathbf{s}'$ is the u-successor of $\mathbf{s}$. Finally, we say that $\mathbf{s}$ is a *deadlock* if no event is enabled at $\mathbf{s}$.

Definition 4. *The Markov chain $\mathcal{M} : \boldsymbol{S} \times \boldsymbol{S} \to [0,1]$ generated by $\mathcal{D}$ is given by:*

- *If $\boldsymbol{s} \in \boldsymbol{S}$ is a deadlock then $\mathcal{M}(\boldsymbol{s}, \boldsymbol{s}) = 1$ and $\mathcal{M}(\boldsymbol{s}, \boldsymbol{s}') = 0$ for $\boldsymbol{s} \neq \boldsymbol{s}'$.*
- *Suppose $\boldsymbol{s} \in \boldsymbol{S}$ is not a deadlock. Then $\mathcal{M}(\boldsymbol{s}, \boldsymbol{s}') = p$ if there exists a maximal step u at $\boldsymbol{s}$ such that $\boldsymbol{s}'$ is the u-successor of $\boldsymbol{s}$ and $p = \prod_{e \in u} p_e$.*
- *If $\boldsymbol{s}$ is not a deadlock and $\boldsymbol{s}'$ is not a successor of $\boldsymbol{s}$ then $\mathcal{M}(\boldsymbol{s}, \boldsymbol{s}') = 0$.*

It follows that $\mathcal{M}(\mathbf{s}, \mathbf{s}') \in [0,1]$ for every $\mathbf{s}, \mathbf{s}' \in \mathbf{S}$. In addition, if u and u' are two maximal steps at $\mathbf{s}$ then $loc(u) = loc(u')$ and $|u| = |u'|$. The initial state of $\mathcal{M}$ is $\mathbf{s}^{in} = (s_1^{in}, s_2^{in}, \ldots, s_n^{in})$.

Lemma 5. *$\mathcal{M}$ is a finite state Markov chain.*

Proof. Clearly $\mathcal{M}(\mathbf{s}, \mathbf{s}') \in [0,1]$ for every $\mathbf{s}, \mathbf{s}' \in \mathbf{S}$. We need to show that

$$\sum_{\mathbf{s}' \in \mathbf{S}} \mathcal{M}(\mathbf{s}, \mathbf{s}') = 1 \; \forall \, \mathbf{s} \in \mathbf{S}.$$

If $\mathbf{s}$ is a deadlock the result follows at once. So assume that $\mathbf{s}$ is not a deadlock.

Let U be a maximal step at $\mathbf{s}$. Let $A_{\mathbf{s}}$ be the set of actions that are enabled at $\mathbf{s}$. Since $\mathbf{s}$ is not a deadlock, $A_{\mathbf{s}}$ is non-empty.

If $(\mathbf{v}, a, \mathbf{v}') \in U$ then $a \in A_{\mathbf{s}}$. On the other hand if $a \in A_{\mathbf{s}}$ then there must exist an event of the form $(\mathbf{v}, a, \mathbf{v}')$ in U since U is a maximal step. For the same reason if $e_1 = (\mathbf{v}, a, \mathbf{v}'), e_2 = (\mathbf{u}, b, \mathbf{u}') \in u$ then $loc(e_1) \cap loc(e_2) \neq \emptyset$ iff $e_1 = e_2$. Thus $|U| = |A_{\mathbf{s}}|$ and for each $a \in A_{\mathbf{s}}$ there exists a unique event in U of the form $(\mathbf{v}, a, \mathbf{v}')$.

In what follows, for the event $(\mathbf{v}, a, \mathbf{v}')$, let ${}^\bullet e = \mathbf{v}$, $act(e) = a$ and $e^\bullet = \mathbf{v}'$. For $a \in A_{\mathbf{s}}$ let $E_a = \{e \mid e \in \Sigma \,,\, act(e) = a\}$. Clearly $\mathbf{s}_a = {}^\bullet e$ for each $e \in E_a$ and $\sum_{e \in E_a} \pi^a(\mathbf{s}_a)(e^\bullet) = 1$ by definition.

The required result now follows from the fact that U is a maximal step at $\mathbf{s}$ iff $|U \cap E_a| = 1$ for each $a \in A_{\mathbf{s}}$. □

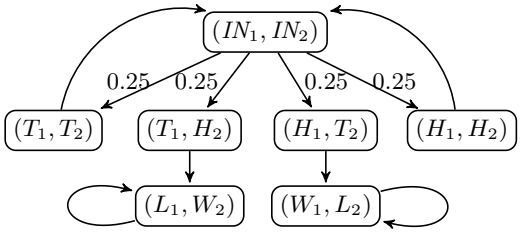

Fig. 3. Markov chain for the DMC in Fig. 2

In Fig. 3 we show the Markov chain of the DMC whose transition system was shown in Fig. 2. Again, unlabelled transitions have probability 1.

Suppose u is a maximal step at $\mathbf{s}$ with $|u| = m$ and $|S_i| = k$ for each $i \in loc(u)$. In $\mathcal{M}$ there will be, in general, k^m transitions at $\mathbf{s}$. In contrast there will be at most $k \cdot m$ transitions at $\mathbf{s}$ in TS. Hence—assuming that we do not explicitly construct $\mathbf{S}$—there can be substantial computational gains if one can verify the properties of $\mathcal{D}$ by working with TS instead of $\mathcal{M}$. This will become clearer when we look at some larger examples in Section 7.

The path space of $\mathcal{M}$ Let $\mathcal{M}$ be the Markov chain associated with a DMC $\mathcal{D}$. The path space and a probability measure over this space is obtained in the usual way. A finite path in $\mathcal{M}$ from $\mathbf{s}$ is a sequence $\tau = \mathbf{s}_0\mathbf{s}_1 \dots \mathbf{s}_m$ such that $\mathbf{s}_0 = \mathbf{s}$ and $\mathcal{M}(\mathbf{s}_\ell, \mathbf{s}_{\ell+1}) > 0$, for $0 \leq \ell < m$. The notion of an infinite path starting from $\mathbf{s}$ is defined as usual. $Path_{\mathbf{s}}$ and $Path^{fin}_{\mathbf{s}}$ denote the set of infinite and finite paths starting from $\mathbf{s}$, respectively.

For $\tau \in Paths^{fin}_{\mathbf{s}}$, $\uparrow\tau \subseteq Paths_{\mathbf{s}}$ is the set of infinite paths that have τ as a prefix. $\Upsilon \subseteq Path_{\mathbf{s}}$ is a basic cylinder at $\mathbf{s}$ if $\Upsilon = \uparrow\tau$ for some $\tau \in Paths^{fin}_{\mathbf{s}}$. The σ-algebra over $Path_{\mathbf{s}}$, denoted $SA(\mathbf{s})$, is the least family that contains the basic cylinders at $\mathbf{s}$ and is closed under countable unions and complementation (relative to $Path_{\mathbf{s}}$). $P_{\mathbf{s}} : SA(\mathbf{s}) \to [0, 1]$ is the usual probability measure that assigns to each basic cylinder $\uparrow\tau$, with $\tau = \mathbf{s}_0\mathbf{s}_1 \dots \mathbf{s}_m$, the probability $p = p_0 \cdot p_1 \cdots p_{m-1}$, where $\mathcal{M}(\mathbf{s}_\ell, \mathbf{s}_{\ell+1}) = p_\ell$, for $0 \leq \ell < m$.

5 The probability measure for the trajectory space

To construct a probability measure over the trajectory space we shall associate infinite paths in $\mathcal{M}$ with maximal trajectories in TS. The Foata normal form from Mazurkiewicz trace theory will help achieve this. Let $\xi \in \Sigma^\star$. A standard fact is that $[\xi]$ can be canonically represented as a "step" sequence of the form $u_1u_2 \dots u_k$. More precisely, the Foata normal form of the finite trace $[\xi]$, denoted $FN([\xi])$, is defined as follows [7].

- $FN([\epsilon]) = \epsilon$.
- Suppose $\xi = \xi' e$ and $FN([\xi']) = u_1u_2 \dots u_k$. If there exists $e' \in u_k$ such that $(e', e) \notin I$ then $FN([\xi]) = u_1u_2 \dots u_k\{e\}$. If not, let ℓ be the least integer in $\{1, 2, \dots, k\}$ such that $e \; I \; e'$ for every $e' \in \bigcup_{\ell \leq m \leq k} u_m$. Then $FN([\xi]) = u_1 \dots u_{\ell-1}(u_\ell \cup \{e\})u_{\ell+1} \dots u_m$.

For the example shown in Fig. 2, $FN(e^1_h \; e^2_t \; e_{ht} \; e_{w_1} \; e_{w_1}) = \{e^1_h, e^2_t\} \; \{e_{ht}\} \; \{e_{w_1}\} \; \{e_{w_1}\}$. This notion is extended to infinite traces in the obvious way. Note that $\xi \approx \xi'$ iff $FN(\xi) = FN(\xi')$.

Conversely, we can extract a (maximal) step sequence from a path in $\mathcal{M}$. Suppose $\mathbf{s}_0\mathbf{s}_1 \dots$ is a path in $Paths_{\mathbf{s}}$. There exists a unique sequence $u_1 \, u_2 \dots$ such that u_ℓ is a maximal step at $\mathbf{s}_{\ell-1}$ and $\mathbf{s}_\ell$ is the u_ℓ-successor of $\mathbf{s}_{\ell-1}$ for every $\ell > 0$. We let $st(\tau) = u_1 \, u_2 \dots$ and call it the step sequence induced by τ.

This leads to the map $tp : Trj_{\mathbf{s}} \to Paths_{\mathbf{s}}$ given by $tp(\rho) = \tau$ iff $FN(ev(\rho)) = st(\tau)$.

Lemma 6. *tp is well defined.*

Proof. Let $\rho \in Trj_{\mathbf{s}}$ such that $\xi = ev(\rho)$ and $FN([\xi]) = u_1 u_2 \cdots$. $FN([\xi])$ is an infinite sequence since we have assumed $\mathcal{M}$ is deadlock-free. Since ρ is a maximal trajectory, by definition, $[\xi]$ is a maximal event trace.

We construct a path $\tau = \mathbf{s}_0 \mathbf{s}_1 \ldots \mathbf{s}_k \cdots \in Trj_{\mathbf{s}}$ such that $\mathbf{s}_l \xrightarrow{u_l} \mathbf{s}_{l+1}$ for $l \geq 0$, which implies $FN([ev(\rho)]) = st(\tau)$. We construct τ inductively as follows:

- u_1 is a maximal step at $\mathbf{s}$. Hence, $\mathbf{s}_0 = \mathbf{s}$ and there exists s_1 such that $\mathbf{s}_0 \xrightarrow{u_1} \mathbf{s}_1$.
  All events $e \in u_1$ are pairwise independent by the definition of Foata normal form. Also, u_1 is maximal because if not, then there exists an event $e \notin u_1$ which is enabled at $\mathbf{s}$ and is pairwise independent with all events in u_1. However, since $[\xi]$ is a maximal event (and hence Mazurkiewicz) trace, there exists u_t for some $t > 1$ such that $e \in u_t$ which contradicts that $u_1 u_2 \cdots$ is a Foata normal form. Finally, there exists at least one event in u_1 that is enabled at $\mathbf{s}$ since $\rho \in Trj_{\mathbf{s}}$. Hence, by the definition of DMC, all events in u_1 are enabled at $\mathbf{s}$ since they are pairwise independent concluding u_1 is a maximal step at $\mathbf{s}$.
- Let us assume we constructed $\mathbf{s}_0 \mathbf{s}_1 \cdots \mathbf{s}_{k-1}$ such that $\mathbf{s}_l \xrightarrow{u_l} \mathbf{s}_{l+1}$ for $0 \leq l < k-1$. By similar argument given above, u_{k-1} is a maximal step enabled at s_{k-1} and hence there exists s_k such that $\mathbf{s}_{k-1} \xrightarrow{u_{k-1}} \mathbf{s}_k$, thus completing the proof. □

As usual, for $X \subseteq Trj_{\mathbf{s}}$ we define $tp(X) = \{tp(\rho) \mid \rho \in X\}$. It turns out that tp maps each basic cylinder in the trajectory space to a finite union of basic cylinders in the path space. As a result, tp maps every measurable set of trajectories to a measurable set of paths. Consequently, one can define the probability of a measurable set of trajectories X to be the probability of the measurable set of paths $tp(X)$.

To understand how tp acts on the basic cylinder $BC(\rho)$, let $FN(ev(\rho)) = u_1 u_2 \ldots u_k$. We associate with ρ the set of finite paths $paths(\rho) = \{\pi \mid st(\pi) = U_1 U_2 \ldots U_k$ and $u_\ell \subseteq U_\ell$, for $1 \leq \ell \leq k\}$. In other words $\pi \in paths(\rho)$ if it extends each step in $FN(ev(\rho))$ to a maximal step. Then, tp maps $BC(\rho)$ to the (finite) union of the basic cylinders generated by the finite paths in $paths(\rho)$. These observations and their main consequence, namely, the construction of a probability measure over the trajectory space, can be summarized as the following Lemma 7, 8 and 9:

Lemma 7. *Let $\rho = \mathbf{s}_0 \mathbf{s}_1 \cdots \mathbf{s}_k$ be a finite trajectory at $\mathbf{s}$. We assume $B = BC(\rho)$ to be a basic trj-cylinder from $\mathbf{s}$ and $FN([ev(\rho)]) = u_1 u_2 \cdots u_k$.*

(i) Suppose $\tau \in Path_{\mathbf{s}}$. Then $\tau \in tp(BC(\rho))$ iff $u_l \subseteq st(\tau)_l$ for $1 \leq l \leq k$. We define $st(\tau)_l$ to be the maximal step appearing in position l of the sequence $st(\tau)$.

(ii) $tp(B)$ is a finite union of basic cylinder sets in $SA_{\mathbf{s}}$ and hence is a member of $SA_{\mathbf{s}}$. Furthermore $P_{\mathbf{s}}(tp(B)) = \prod_{1 \leq l \leq k} p_l$ where $p_l = \prod_{e \in u_l} p_e$ for $1 \leq l \leq k$.

Proof. (i) We need to show that $tp(BC(\rho)) = paths(\rho)$.
Suppose $\rho' \in BC(\rho)$. Then, by definition of basic cylinders, there exists $\rho'' \in BC(\rho)$ such that $\rho' \simeq \rho''$ and ρ is a finite prefix of ρ''. Hence

$$\begin{aligned}
&\rho' \simeq \rho'' \\
&\implies ev(\rho') \approx ev(\rho'') && \text{(definition of } \simeq) \\
&\implies FN([ev(\rho')]) = FN([ev(\rho'')]) && \text{(uniqueness of } FN) \\
&\implies tp(\rho') = tp(\rho'') && \text{(definition of } tp)
\end{aligned}$$

Since $FN([ev(\rho)]) = u_1 u_2 \cdots u_k$ and ρ is a finite prefix of ρ'', $tp(\rho'') \in paths(\rho)$ and hence $tp(\rho') \in paths(\rho)$. Thus $tp(BC(\rho)) \subseteq paths(\rho)$.
To prove the converse, let $\tau \in paths(\rho)$ with $st(\tau) = U_1 U_2 \cdots$. We need to prove $\tau \in tp(BC(\rho))$. By definition of tp and BC, it is sufficent to prove that there exists maximal trajectory ρ' such that $\rho' \in BC(\rho)$ and $FN([ev(\rho')]) = st(\tau)$.
We proceed by induction on k. Suppose $FN(ev(\rho)) = u_1$ with $st(\tau)_1 = U_1$ and $u_1 \subseteq U_1$. Let $U_1 \backslash u_1 = \{e_1, e_2, \ldots, e_m\}$. Then it is easy to see that there exists a trajectory of the form $\rho' = \rho\rho_1\rho_2 \in Trj_{\mathbf{s}}$ such that $ev(\rho_1) = e_1 e_2 \ldots e_m$. Clearly $\rho' \in BC(\rho)$ such that $FN(ev(\rho')) = U_1 = st(\tau)$ and this proves the basis step. The induction step follows from the induction hypothesis applied at the U-successor of $\mathbf{s}$ for each U defined as above.

(ii) We again proceed by induction on k. Suppose $k = 1$. Let us assume $\{U_1, U_2, \ldots, U_m\}$ to be the set of maximal steps at $\mathbf{s}$ such that $u_l \subseteq U_l$ for $1 \leq l \leq m$. Then, from the previous part, it follows that $tp(BC(\rho)) = \bigcup_{1 \leq l \leq m} \uparrow(\mathbf{s}\mathbf{s}_l)$ where $\mathbf{s}_l$ is the U_l-successor of $\mathbf{s}$ for $1 \leq l \leq m$. This establishes the basis step. The induction step now follows by appealing to the induction hypothesis at $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m$. This proves that $tp(B)$ is a member of $SA_{\mathbf{s}}$. Then using the previous part of the lemma and inducing on k, it is easy to show that $P_{\mathbf{s}}(tp(B)) = \prod_{1 \leq l \leq k} p_l$ where $p_l = \prod_{e \in u_l} p_e$ for $1 \leq l \leq k$.

□

Lemma 8. *Let $B \in \widehat{SA}_{\mathbf{s}}$.*

(i) B is $\simeq$-closed.
(ii) $tp(Trj_{\mathbf{s}} - B) = Path_{\mathbf{s}} - tp(B)$.
(iii) $tp(B) \in SA_{\mathbf{s}}$.

Proof. (i) For $\rho \in B$ and $\rho' \simeq \rho$, by definition of $B \in \widehat{SA}_{\mathbf{s}}$ it follows that $\rho' \in B$. Hence B is $\simeq$-closed.

(ii) Let $\rho \in Trj_{\mathbf{s}} - B$. If $tp \in tp(B)$ then there exists $\rho' \in B$ such that $tp(\rho') = tp(\rho)$. But this will imply that $FN([ev(\rho)]) = FN([ev(\rho')])$ which in turn would imply that $ev(\rho) \approx ev(\rho')$. But this implies that $\rho \simeq \rho'$ leading to the contradiction that $\rho \in B$ since B is $\simeq$-closed. This shows that $tp(Trj_{\mathbf{s}} - B) \subseteq Path_{\mathbf{s}} - tp(B)$.
Now assume $\tau \in Path_{\mathbf{s}} - tp(B)$. It is easy to show that there exists a $\rho \in Trj_{\mathbf{s}}$ such that $tp(\rho) = \tau$. Clearly $\rho \in Trj_{\mathbf{s}} - B$. Hence $Path_{\mathbf{s}} - tp(B) \subseteq tp(Trj_{\mathbf{s}} - B)$ which concludes the proof.

(iii) If B is a basic cylinder at $\mathbf{s}$, then the result follows from part *(iii)* of Lemma 7. By definition, $\widehat{SA}_{\mathbf{s}}$ is closed under complementation and countable union. Since $Path_{\mathbf{s}} - tp(B) \in SA_{\mathbf{s}}$, from the previous part we now have $tp(Trj_{\mathbf{s}} - B) \in SA_{\mathbf{s}}$. Finally, if $\{B_k\}$ is a countable collection such that $B_k \in \widehat{SA}_{\mathbf{s}}$ and $tp(B_k) \in SA_{\mathbf{s}}$ for each k, $tp(\bigcup_k B_k) = \bigcup_k tp(B_k) \in SA_{\mathbf{s}}$.
Hence for any $B \in \widehat{SA}_{\mathbf{s}}$, $tp(B) \in SA_{\mathbf{s}}$. □

Lemma 9. *$\widehat{P}_{\mathbf{s}}$ is well defined.*

Proof. Since tp is well-defined and $P_{\mathbf{s}}$ is a probability measure, $\widehat{P}_{\mathbf{s}}$ is a well-defined function. To prove that $\widehat{P}_{\mathbf{s}}$ is a well-defined measure on $\widehat{SA}_{\mathbf{s}}$, we prove the following:

- $\widehat{P}_{\mathbf{s}}(\widehat{SA}_{\mathbf{s}}) = P_{\mathbf{s}}(tp(\widehat{SA}_{\mathbf{s}})) = P_{\mathbf{s}}(SA_{\mathbf{s}}) = 1$, since $P_{\mathbf{s}}$ is a probability measure.
- Let $\{B_k\}$ be a countable collection of pairwise disjoint sets in $\widehat{SA}_{\mathbf{s}}$. Clearly $\{tp(B_k)\}$ is a countable collection of pairwise disjoint sets in $SA_{\mathbf{s}}$. We now have:

$$\begin{aligned}
\widehat{P}_{\mathbf{s}}(\bigcup_k B_k) &= P_{\mathbf{s}}(tp(\bigcup_k B_k)) && \text{(definition of } \widehat{P}_{\mathbf{s}}) \\
&= P_{\mathbf{s}}(\bigcup_k tp(B_k)) && (\textstyle\bigcup_k \text{ is countable union}) \\
&= \sum_k P_{\mathbf{s}}(tp(B_k)) && \text{(part } \textit{(iii)} \text{ of Lemma 8)} \\
&= \sum_k \widehat{P}_{\mathbf{s}}(B_k) && \text{(definition of } \widehat{P}_{\mathbf{s}})
\end{aligned}$$

Hence, $\widehat{P}_{\mathbf{s}}$ is a probability measure. □

Note that while a finite path in $\mathcal{M}$ always induces a maximal step sequence, a finite trajectory, in general, does not have this structure. Some components can get ahead of others by an arbitrary amount. The lemmas above states that, despite this, any finite trajectory defines a finite set of of basic cylinders whose overall probability can be easily computed, by taking the product of the probabilities of the events encountered along the trajectory. This helps considerably when verifying the properties of $\mathcal{M}$. In particular, local reachability properties can be checked by exercising only those components that are relevant.

Going back to our running example, let $\rho_t = (IN_1, IN_2)e_t^1(T_1, IN_2)$, and $X_t = \uparrow\rho_t$. Let $\rho'_t = (IN_1, IN_2)e_t^2(IN_1, T_2)$, and $X'_t = \uparrow\rho_t$. Assume ρ_h, X_h, ρ'_h and X'_h are defined similarly. Then $\widehat{P}(X_t) = \widehat{P}(X'_h) = 0.5$, while $\widehat{P}(X_h \cup X_t) = 1$. On the other hand, due to the fact that e_h^1 and e_h^2 are independent, we have $\widehat{P}(X_h \cup X'_h) = 0.75$.

6 A statistical model checking procedure for DMCs

To bring out the applicability of the DMC formalism and its interleaved semantics, we formulate a statistical model checking procedure. The specification logic $PBLTL^{\otimes}$ (product $PBLTL$) is a simple generalization of probabilistic bounded linear time temporal logic ($PBLTL$) [14] that captures Boolean combinations of local properties of the components. The logic can express interesting global reachability properties as well since the Boolean connectives can capture -in a limited fashion- the way the components influence each other.

We assume a collection of pairwise disjoint sets of atomic propositions $\{AP_i\}$. As a first step, the formulas of $BLTL^{\otimes}$ are given as follows.

(i) $ap \in AP_i$ is a $BLTL^{\otimes}$ formula and $type(ap) = \{i\}$.
(ii) If φ and φ' are $BLTL^{\otimes}$ formulas with $type(\varphi) = type(\varphi') = \{i\}$ then so is $\varphi \mathbf{U}_i^t \varphi'$ where t is a positive integer. Further, $type(\varphi \mathbf{U}_i^t \varphi') = \{i\}$. As usual, $F^t\varphi$ abbreviates $(true\mathbf{U}^t\varphi)$ and $G^t\varphi$ is defined as $\neg F^t \neg\varphi$.
(iii) If φ and φ' are $BLTL^{\otimes}$ formulas then so are $\neg\varphi$ and $\varphi \vee \varphi'$ with $type(\neg\varphi) = type(\varphi)$ and $type(\varphi \vee \varphi') = type(\varphi) \cup type(\varphi')$.

The formulas of $PBLTL^{\otimes}$ are given by:

(i) Suppose φ is a $BLTL^{\otimes}$ formula and γ a rational number in the open interval $(0, 1)$. Then $Pr_{\geq\gamma}(\varphi)$ is a $PBLTL^{\otimes}$ formula.
(ii) If ψ and ψ' are $PBLTL^{\otimes}$ formulas then so are $\neg\psi$ and $\psi \vee \psi'$.

To define the semantics, we project each trajectory to its components. For $\mathbf{s} \in \mathbf{S}$ and $i \in [n]$ we define $Proj_i : Trj_{\mathbf{s}}^{fin} \to S_i^+$ inductively.

(i) $Proj_i(\mathbf{s}) = \mathbf{s}_i$.
(ii) Suppose $\rho = \mathbf{s}_0 e_o \mathbf{s}_1 \ldots \mathbf{s}_m e_m \mathbf{s}_{m+1}$ is in $Trj_{\mathbf{s}}^{fin}$ and $\rho' = \mathbf{s}_0 e_0 \mathbf{s}_1 \ldots \mathbf{s}_m$. If $i \in loc(e_m)$ then $Proj_i(\rho) = Proj_i(\rho')(\mathbf{s}_{m+1})_i$. Otherwise $Proj_i(\rho) = Proj_i(\rho')$.

We lift $Proj_i$ to infinite trajectories in the obvious way—note that $Proj_i(\rho)$ can be a finite sequence for the infinite trajectory ρ. We assume a set of local valuation functions $\{V_i\}$, where $V_i : S_i \to 2^{AP_i}$. Let φ be a $BLTL^{\otimes}$ formula with $type(\varphi) = \{i\}$. We begin by interpreting such formulas over sequences generated by the alphabet S_i. For $\varrho \in S_i^+ \cup S_i^\omega$, the satisfaction relation $\varrho, k \models_i \varphi$, with $0 \leq k \leq |\varrho|$, is defined as follows.

(i) $\varrho, k \models_i ap$ for $ap \in AP_i$ iff $ap \in V_i(\varrho(k)(i))$, where $\varrho(k)(i)$ is the S_i-state at position k of the sequence ϱ.
(ii) $\neg$ and $\vee$ are interpreted in the usual way.
(iii) $\varrho, k \models_i \varphi_1 \mathbf{U}_i^t \varphi_2$ iff there exists ℓ such that $k \leq \ell \leq max(k + t, |\varrho|)$ with $\varrho, \ell \models_i \varphi_2$, and $\varrho, m \models_i \varphi_1$, for $k \leq m < \ell$.

As usual, $\varrho \models_i \varphi$ iff $\varrho, 0 \models_i \varphi$. Next, suppose φ is a $BLTL^{\otimes}$ formula and $\rho \in Path_{\mathbf{s}}$. Then the relation $\rho \models_{\mathbf{s}} \varphi$ is defined as follows.

(i) If $type(\varphi) = \{i\}$ then $\rho \models_{\mathbf{s}} \varphi$ iff $Proj_i(\rho) \models_i \varphi$.
(ii) Again, $\neg$ and $\vee$ are interpreted in the standard way.

Given a formula φ in $BLTL^{\otimes}$ and a global state $\mathbf{s}$, we define $Trj_{\mathbf{s}}(\varphi)$ to be the set of trajectories $\{\rho \in Trj_{\mathbf{s}} \mid \rho \models_s \varphi\}$.

Lemma 10. *For every formula φ, $Trj_s(\varphi)$ is a member of $\widehat{SA}(\boldsymbol{s})$.*

Proof. Let $Path_{\mathbf{s}}(\varphi)$ be the set of paths in the Markov chain $\mathcal{M}$ that satisfies the formula φ. It is easy to see that $Path_{\mathbf{s}}(\varphi)$ is a member of $SA_{\mathbf{s}}$. We then use Lemma 8 to obtain the result. □

The semantics of $PBLTL^{\otimes}$ is now given by the relation $\mathcal{D} \models^{trj}_{\mathbf{s}} \psi$, defined as:

(i) Suppose $\psi = Pr_{\geq\gamma}(\varphi)$. Then $\mathcal{D} \models^{trj}_{\mathbf{s}} \psi$ iff $\widehat{P}(Path_{\mathbf{s}}(\varphi)) \geq \gamma$.
(ii) Again, the interpretations of $\neg$ and $\vee$ are the standard ones.

For the example in Fig. 1, one can assert $Pr_{\geq 0.99}((F^7(L_1) \wedge F^7(W_2)) \vee (F^7(W_1) \wedge F^7(L_2)))$. Here, the local states also serve as the atomic propositions. Hence, the formula says that with probability ≥ 0.99, a winner will be decided within 7 rounds.

We write $\mathcal{D} \models^{trj} \psi$ for $\mathcal{D} \models^{trj}_{\mathbf{s}^{in}} \psi$. The model checking problem is to determine whether $\mathcal{D} \models^{trj} \psi$. We shall adapt the SMC procedure developed in [20] to solve this problem approximately.

6.1 The statistical model checking procedure

We note that in the Markov chain setting, given a BLTL formula and a path in the chain, there is a bound k that depends only on the formula such that we can decide whether the path is a model of the formula by examining just a prefix of the path of length k [14]. By the same reasoning, for a $BLTL^{\otimes}$ formula φ, we can compute a vector of bounds $(k_1, k_2, \ldots, k_n)$ that depends only on φ such that for any trajectory ρ starting from $\mathbf{s}^{in}$, we only need to examine a finite prefix ρ' of ρ that satisfies $|Proj_i(\rho')| \geq k_i$, for $1 \leq i \leq n$. The complication in our setting is that it is not guaranteed that one can generate such a prefix with bounded effort. This is due to the mix of concurrency and stochasticity in DMCs. More precisely, at a global state $\mathbf{s}$, one may need to advance the history of the agent i but this may require first executing an event $e = (\mathbf{v}, a, \mathbf{v}')$ at $\mathbf{s}$ that does not involve the agent i. However, there could also be another event $e' = (\mathbf{v}, a, \mathbf{u})$ enabled at $\mathbf{s}$. Since one must randomly choose one of the enabled events according to the underlying probabilities, one may repeatedly fail to steer the computation towards a global state in which the history of i can be advanced. A second complication is that starting from the current global state it may be impossible to reach a global state at which some event involving i is enabled.

To cope with this, we maintain a count vector $(c_1, c_2, \ldots, c_n)$ that records how many times each component has moved along the trajectory ρ that has been generated so far. A simple reachability analysis will reveal whether a component

is *dead* in the current global state — that is, starting from the current state, there is no possibility of reaching a state in which an event involving this agent can be executed. If this is the case for the agent i or the c_i is already the required bound then remove it from the current set of active agents. If the current set of active agents is not empty, we execute, one by one, all the enabled actions — using a fixed linear order over the set of actions — followed by one move by each of the participating agents, according to the underlying probabilities. Recall that action a is enabled at $\mathbf{s}$ iff $\mathbf{s}_a \in en_a$. Due to the determinacy of synchronizations, the global state thus reached will depend only on the probabilistic moves chosen by the participating agents. We then update the count vector to $(c'_1, c'_2, \ldots, c'_n)$ and mark the new dead components. We show in Theorem 11 that, continuing in this manner, with probability 1 we will eventually generate a finite trajectory $\widehat{\rho}$ and reach a global state $\mathbf{s}$ with no active agents. We then check if $\widehat{\rho}$ satisfies φ and update the score associated with the statistical test described as follows.

Theorem 11. *With probability* 1*, the method of generating a finite trajectory described in section 6.1 will terminate.*

Proof. The core task is to check whether $\mathcal{D} \models^{trj}_{\mathbf{s}} \widehat{P}_{\geq\gamma}(\varphi)$ where φ is a $BLTL^{\otimes}$ formula. Let $(k_1, k_2, \ldots, k_n)$ be the bound vector obtained from φ as follows. If $i \notin type(\varphi)$ then $k_i = 0$. If $i \in type(\varphi)$ then k_i is the maximum of the bounds associated (in the usual way) with the i-type formulas appearing in φ.

Let ρ be a finite trajectory such that the corresponding count vector is $(c_1, c_2, \ldots, c_n)$. In other words the agent i has executed c_i events so far along ρ. Let $\mathbf{s}^{in} \xrightarrow{\rho} \mathbf{s}$. Then, ρ is *terminal* if for every i either $c_i \geq k_i$ or there exists i such that $c_i < k_i$ and the component i is dead at $\mathbf{s}$. It is easy to show that one can effectively determine if a finite trajectory is terminal. Let TER be the set of terminal finite trajectories and $\Upsilon = \cup_{\rho \in TER} BC(\rho)$. Clearly TER is a countable set and hence Υ is in $\widehat{SA}$. It is now enough to prove that $\widehat{P}(\Upsilon) = 1$.

It will be more convenient to carry out the proof in the Markov chain setting. We first blow up $\mathcal{M}$ into the larger $\widehat{\mathcal{M}}$ as follows. Let $\widehat{\mathbf{S}} = \mathbf{S} \times (C_1 \times C_2 \times \cdots \times C_n)$ such that $C_i = \{0, 1, \ldots, k_i\} \cup \{\omega_i\}$ for every i. Here, ω_i denotes "large" and satisfies $k_i < \omega_i$ and $\omega_i + 1 = \omega_i$ for every i.

We define $\widehat{\mathcal{M}}$ via: $\widehat{\mathcal{M}}((\mathbf{s}, \mathbf{c})(\mathbf{s}', \mathbf{c}')) = p$ iff $\mathcal{M}(\mathbf{s}, \mathbf{s}') = p$. Morever, if $\mathbf{s}'$ is the u-successor of $\mathbf{s}$ then for each i the following conditions hold. Suppose $i \in loc(u)$. Then $\mathbf{c}'(i) = \mathbf{c} + 1$ if $\mathbf{c}(i) < k_i$ and $\mathbf{c}'(i) = \omega_i$ if $\mathbf{c}(i) = k_i$ or $\mathbf{c}(i) = \omega_i$. On the other hand, $\mathbf{c}'(i) = \mathbf{c}(i)$ if $i \notin loc(u)$.

It is easy to verify that $\widehat{\mathcal{M}}$ is a finite state Markov chain. The initial state is $(\mathbf{s}^{in}, \mathbf{0})$, where $\mathbf{0}(i) = 0$ for each i. In what follows we will often suppress the mention of $(\mathbf{s}^{in}, \mathbf{0})$.

Let τ be a finite path in $\widehat{\mathcal{M}}$ that ends at $(\mathbf{s}, \mathbf{c})$. We say $(\mathbf{s}, \mathbf{c})$ is *final* iff $k_i \leq \mathbf{c}_i$ for every i or there exists i such that $\mathbf{c}(i) < k_i$ and i is dead at $(\mathbf{s}, \mathbf{c})$. Then τ is *terminal* iff $(\mathbf{s}, \mathbf{c})$ is final. Similar to TER, we now define $\widehat{TER}$ to be the set of terminal finite paths in $\widehat{\mathcal{M}}$. From the definitions it follows that $tp(TER) = \widehat{TER}$ and hence it suffices to prove that $P(\cup_{\tau \in \widehat{TER}} BC(\tau)) = 1$. Here, P denotes the

usual probability measure over $SA_{(\mathbf{s}^{in},\mathbf{0})}$ in $\widehat{\mathcal{M}}$. We also note that $\widehat{TER}$ is a countable set and hence $\bigcup_{\tau\in\widehat{TER}} BC(\tau)$ will be a member of $SA_{(\mathbf{s}^{in},\mathbf{0})}$.

Next, let $(\mathbf{s},\mathbf{c}) \in \widehat{\mathcal{S}}$. Let $\widehat{\mathbf{W}}$ be the set of final states. Now as shown in [3], $P(\bigcup_{\tau\in\widehat{TER}} BC(\tau)) = 1$ follows if we can show that $(\mathbf{s}^{in},\mathbf{0}) \in \widehat{\mathbf{S}} - Pre^*(\widehat{\mathbf{S}} - Pre^*(\widehat{\mathbf{W}}))$ where $Pre^*(\widehat{\mathbf{x}})$ for $\widehat{\mathbf{S}} \subseteq \widehat{\mathbf{S}}$ is the least subset of $\widehat{\mathbf{S}}$ that contains $\widehat{\mathbf{X}}$ and satisfies: If $\widehat{\mathbf{s}} \in \widehat{\mathbf{X}}$ and $\widehat{\mathcal{M}}(\widehat{\mathbf{s}'},\widehat{\mathbf{s}}) > 0$ then $\widehat{\mathbf{s}'} \in Pre^*(\widehat{\mathbf{X}})$.

We now claim that $Pre^*(\widehat{\mathbf{W}}) = \widehat{\mathbf{S}}$. It is easy to see that proving this is sufficient since this leads to $\widehat{\mathbf{S}} - Pre^*(\widehat{\mathbf{S}} - Pre^*(\widehat{\mathbf{W}})) = \widehat{\mathbf{S}}$ and trivially $(\mathbf{s}^{in},\mathbf{0}) \in \widehat{\mathbf{S}}$. We recall $\widehat{\mathbf{Y}} \subseteq \widehat{\mathbf{S}}$ is a strongly connected component of $\mathcal{M}$ iff there is a path from $\widehat{\mathbf{s}}$ to $\widehat{\mathbf{u}}$ for every $\widehat{\mathbf{s}},\widehat{\mathbf{u}} \in \widehat{\mathbf{Y}}$. Let $\{SC_1, SC_2, \ldots, SC_r\}$ be the set of strongly connected components (SCCs) of $\widehat{\mathbf{M}}$. The relation $\preceq$ over SCCs is given by: $SC \preceq SC'$ iff there exists a state $\widehat{\mathbf{s}} \in SC$, a state $\widehat{\mathbf{u}} \in SC'$ and a path from $\widehat{\mathbf{s}}$ to $\widehat{\mathbf{u}}$ in $\widehat{\mathbf{M}}$. Clearly, $\preceq$ is a partial order relation and the maximal elements under *preceq* will be called the bottom strongly connected components (BSCCs).

Now let $\widehat{\mathbf{Y}}$ be a BSCC and $(\mathbf{s},\mathbf{c}),(\mathbf{s}',\mathbf{c}') \in \widehat{\mathbf{Y}}$. Then it must be the case that $\mathbf{c} = \mathbf{c}'$. Next, suppose $\mathbf{c}(i) < k_i$ for some i. Then i must be dead at $(\mathbf{s},\mathbf{c})$ which implies that i is dead at every state at $\widehat{\mathbf{Y}}$. Consequently, every state in $\widehat{\mathbf{Y}}$ is final. By definition of BSCCs, we now have $Pre^*(\widehat{\mathbf{W}}) = \widehat{\mathbf{S}}$, thus concluding the proof. □

Statistical test The parameters for the test are δ, α, β, where δ is the size of the indifference region and (α,β) is the strength of the test, with α bounding the Type I errors (false positives) and β bounding the Type II errors (false negatives). These parameters are to be chosen by the user. We generate finite i.i.d. sample trajectories sequentially. We associate a Bernoulli random variable x_ℓ with the sample ρ_ℓ and set $x_\ell = 1$ if $\rho_\ell \in Trj_{\mathbf{s}^{in}}(\varphi)$ and set $x_\ell = 0$ otherwise. We let $c_m = \sum_\ell x_\ell$ and compute the score $SPRT$ via

$$SPRT = \frac{(\gamma^-)^{c_m}(1-\gamma^-)^{n-c_m}}{(\gamma^+)^{c_m}(1-\gamma^+)^{n-c_m}}$$

Here $\gamma^+ = \gamma + \delta$ and $\gamma^- = \gamma - \delta$. If $SPRT \leq \frac{\beta}{1-\alpha}$, we declare $\mathcal{D} \models^{trj} \widehat{P}_{\geq r}\varphi$. If $SPRT \geq \frac{1-\beta}{\alpha}$, we declare $\mathcal{D} \not\models^{trj} \widehat{P}_{\geq\gamma}\varphi$. Otherwise, we draw one more sample and repeat.

This test is then extended to handle formulas of the form $\neg\psi$ and $\psi_1 \vee \psi_2$ in the usual way [14]. It is easy to establish the correctness of this statistical model checking procedure.

7 Experimental results

We have tested our SMC procedure on two probabilistic distributed algorithms: (i) a leader election protocol for a unidirectional ring of anonymous processes by Itai and Rodeh [10,12] and (ii) a randomized solution to the dining philosophers problem [17]. Both these algorithms -for large instances- exhibit a considerable

degree of concurrency since any two agents that do not share a neighbor can execute independently. We focused on approximately verifying termination properties of these algorithms to bring out the scalability and the performance features of our SMC technique. We also compared our results with the corresponding ones obtained using the tool PLASMA [5].

In the leader election protocol, each process randomly chooses an identity from $\{1, 2, \ldots, N\}$, and passes it on to its neighbor. If a process receives an identity lower than its own, the message is dropped. If the identity is higher than its own, the process drops out of the election and forwards the message. Finally, if the identity is the same as its own, the process forwards the message, noting the identity clash. If an identity clash is recorded, all processes with the highest identity choose a fresh identity and start another round.

Since the initial choice of identity for the N processes can be done concurrently, in the global Markov chain, there will be N^N possible moves. However, correspondingly in the interleaved semantics, there will be N^2 transitions from the initial state.

We have built a DMC model of this system in which each process and channel is an agent. Messages are transferred between processes and channels via synchronizations while ensuring this is done using a deterministic protocol. For simplicity, all channels in our implementation have capacity 1. We can easily construct higher capacity channels by cascading channels of capacity 1 while staying within the DMC formalism.

The challenge in modeling the dining philosophers problem as a DMC is to represent the forks between philosophers, which are typically modeled as shared variables. We use a deterministic round robin protocol to simulate these shared variables. The same technique can be used for a variety of other randomized distributed algorithms presented as case studies for PLASMA.

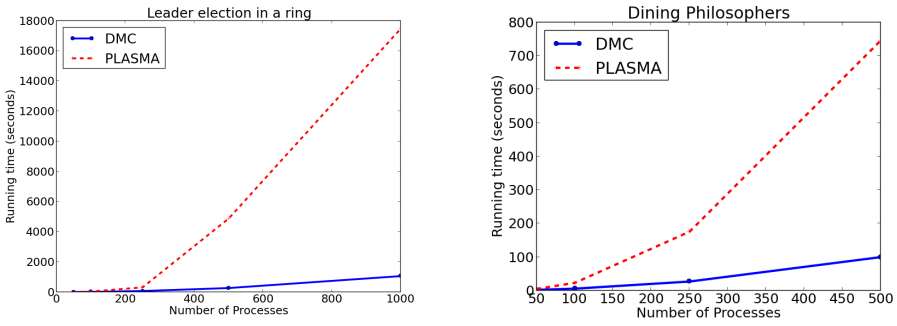

Fig. 4. Comparison of simulation times in DMC and PLASMA

We ran our trajectories based SPRT procedure written in Python programming language on a Linux server (Intel Xeon 2.30 GHz, 16 core, 72GB RAM). For the first example, we verified that a leader is elected with probability above 0.99 within K rounds, for a ring of N processes for various values of N up to

1000. For each N, it turned out that $K = N$ is a good choice for K as explained below.

The termination property was specified as follows. Let L_i denote the boolean variable which evaluate to true iff in the current global state, node i has been elected as the leader. Then for N processes with $K = N$, the specification is:

$$Pr_{\geq 0.99}(\bigvee_{i=1}^{n}[F^N(\neg L_1)\wedge\cdots\wedge F^N(\neg L_{i-1})\wedge F^N(L_i)\wedge F^N(\neg L_{i+1})\wedge\cdots\wedge F^N(\neg L_n)])$$

For the dining philosophers, we verified that with probability above 0.95 every philosopher eventually eats, up to $N = 500$ philosophers. In both the experiments, we set the bound on both the Type I and Type II errors to be 0.01 and the indifference region to be 0.01. We tested the same properties with the same statistical parameters using the PLASMA model. PLASMA supports parallel execution and multithreading. Since our DMC implementation is currently sequential, we restricted PLASMA to single-threaded sequential execution for a meaningful comparison.

In Fig. 4, we have compared the running time for SPRT model-checking in the DMC model with that for PLASMA. The x-axis is the number of processes in the system and the y-axis is the running time, in seconds. We have not been able to determine from the literature how PLASMA translates the model into a DTMC. Consequently we could only compare the simulation times at the system level while treating the PLASMA tool as a black box. In the case of PLASMA, the specifications use time bounds which we took it to imply the number of time steps for which the model is simulated. We found that for $N = 1000$, PLASMA verifies the termination property to be true if the time bound is set to be $10,000$. Further, increasing the bound does not cause the simulation times to change. Hence we fixed the time bound to be $10,000$ for all choices of N in the PLASMA setting. In the DMC setting we found that setting $K = N$ caused our implementation to verify the termination property to be true. Again, increasing this to larger number of rounds does not change the simulation times. For this reason we fixed $K = N$ for each N.

The experiments show that as the model size increases, the running time increase for the DMC approach is almost linear whereas for PLASMA it is more rapid. The results also show the significant performance and scalability advantages of using the interleaved semantics approach based on DMC models. We expect further improvements to be easily achieved via a parallel implementation.

8 DMCs with global final states

At first sight, DMCs appear to be closely related to the model of deterministic cyclic negotiations (DCNs) [9]. However, in the DCNs, the control flow is not influenced in anyway by the internal states of the agents. This issue was not dealt with carefully enough in the translation proposed in the conference paper and in the accompanying technical report since its correctness seemed obvious!

The trouble is, in a DMC, the control flow is strongly influenced by the internal states of the agents and this leads to a significant gap between the

expressive powers of the two formalisms. This gap is best brought out in net theoretic terms. A DCN can be easily represented as the subclass of Petri nets known as extended free choice nets (EFC nets). In such nets, if a place is a common input place of two transitions then these transitions have the same set of input places. In other words it can not be the case that two transition t_1 and t_2 have as their input places, say, $\{p_1, p\}$ and $\{p_2, p\}$ respectively with $p_1 \neq p_2$ since p is a common input place.

On the other hand, the natural net representation of a DMC will *not* in general be an EFC net. For instance, for the coin toss example, its net representation is shown in Figure 5, which is obviously not an EFC net.

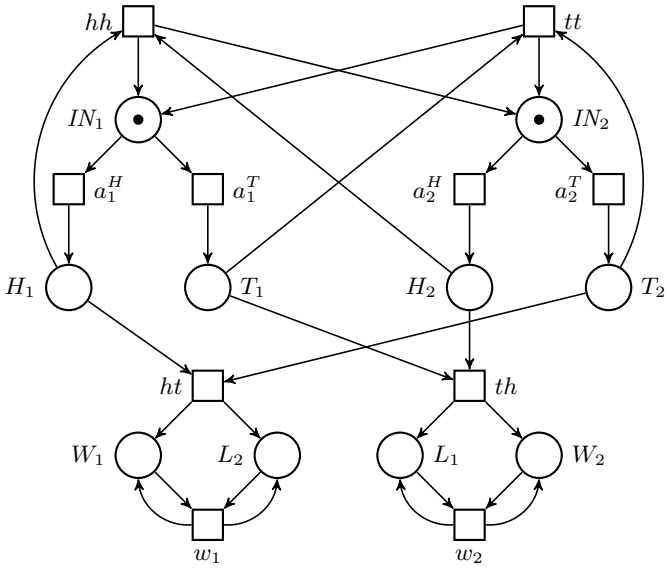

Fig. 5. Net representation for the coin toss example

We believe that there are natural subclasses of DMCs that can be represented as DCNs and hence will admit a polynomial time procedure for checking their soundness (when they are accompanied by final states). However the full class of DMCs does not admit a representation as DCNs and it is not clear whether such DMCs accompanied by final states admit a polynomial time algorithm for checking soundness (i.e every state reachable from the initial state can reach the final state).

9 Conclusion

We have formulated a distributed probabilistic system model called DMCs. Our model achieves a clean mix of concurrency and probabilistic dynamics by restricting synchronizations to be deterministic. Our key technical contribution is

the construction of a probability measure over the σ-algebra generated by the (interleaved) trajectories of a DMC. This opens the door to using partial order reduction techniques to efficiently verify the dynamic properties of a DMC. As a first step we have developed a SPRT based statistical model checking procedure for the logic $PBLTL^{\otimes}$. Our experiments suggest that our method can handle systems of significant sizes.

The main partial order concept we have used is to group trajectories into equivalence classes. One can also explore how ample sets [15] and related notions can be used to model check properties specified in logics such as PCTL [3]. Another possibility is to see if the notion of finite unfoldings from Petri net theory can be applied in the setting of DMCs [8, 16].

In our two case studies, the specification has a global character in that it mentions every agent in the system. In many specifications, only a few agents will be mentioned. If the system is loosely coupled, we can check whether the required property is fulfilled without having to exercise all the agents. This will lead to additional computational gains.

In many of the benchmark examples in [6], the probabilistic moves are local. On the other hand, DMCs allow synchronous probabilistic moves where the probability distribution is influenced by information obtained through communication. It will be interesting to exploit this feature to model and analyze applications arising in embedded control systems.

We currently allow agents to gain complete information about the state of the agents they synchronize with. In practice, only a part of this state may/should be exposed.

References

1. Samy Abbes and Albert Benveniste. True-concurrency probabilistic models: Branching cells and distributed probabilities for event structures. *Info. and Comp.*, 204(2):231–274, 2006.
2. Samy Abbes and Albert Benveniste. True-concurrency probabilistic models: Markov nets and a law of large numbers. *Theor. Comput. Sci.*, 390(2-3):129170, 2008.
3. Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking.* The MIT Press, 2008.
4. Jonathan Bogdoll, Luis Mara Ferrer Fioriti, Arnd Hartmanns, and Holger Hermanns. Partial order methods for statistical model checking and simulation. In *Formal Techniques for Distributed Systems*, Lecture Notes in Computer Science, pages 59–74. 2011.
5. Benoît Boyer, Kevin Corre, Axel Legay, and Sean Sedwards. Plasma-lab: A flexible, distributable statistical model checking library. In *Proc. QEST 2013*, volume 8054 of *Lecture Notes in Computer Science*, pages 160–164, 2013.
6. PRISM: case studies. `http://www.prismmodelchecker.org/casestudies/`.
7. V. Diekert and G. Rozenberg. The book of traces. World Scientific, 1995.
8. J. Esparza and K. Heljanko. *Unfoldings: A Partial-Order Approach to Model Checking.* Springer Publishing Company, 1 edition, 2008.

9. Javier Esparza and Jörg Desel. On negotiation as concurrency primitive II: Deterministic cyclic negotiations. In *FoSSaCS*, pages 258–273, 2014.
10. Wan Fokkink. Variations on Itai-Rodeh leader election for anonymous rings and their analysis in PRISM. *J. UCS*, 12, 2006.
11. Marcus Groesser and Christel Baier. Partial order reduction for Markov decision processes: A survey. In *Formal Methods for Components and Objects*, pages 408–427. 2006.
12. Alon Itai and Michael Rodeh. Symmetry breaking in distributed networks. *Info. and Comp.*, 88(1):60–87, September 1990.
13. S. Jesi, G. Pighizzini, and N. Sabadini. Probabilistic asynchronous automata. *Math. Systems Theory*, 29(1):5–31, February 1996.
14. Sumit K. Jha, Edmund M. Clarke, Christopher J. Langmead, Axel Legay, Andr Platzer, and Paolo Zuliani. A Bayesian approach to model checking biological systems. In *Computational Methods in Systems Biology*, pages 218–234. 2009.
15. Edmund M. Clarke Jr, Orna Grumberg, and Doron A. Peled. *Model Checking*. The MIT Press, Cambridge, Mass, 1999.
16. K. L. McMillan and D. K. Probst. A technique of state space search based on unfolding. *Form Method Syst Des*, 6(1):45–65, January 1995.
17. Amir Pnueli and Lenore D. Zuck. Verification of multiprocess probabilistic protocols. *Distributed Computing*, 1(1):53–72, 1986.
18. Daniele Varacca, Hagen Völzer, and Glynn Winskel. Probabilistic event structures and domains. In *CONCUR 2004 - Concurrency Theory*, pages 481–496. 2004.
19. A. Wald. Sequential tests of statistical hypotheses. *Ann. Math. Statist.*, pages 117–186.
20. Hakan Lorens Samir Younes. *Verification and Planning for Stochastic Processes with Asynchronous Events*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2004.
21. Wieslaw Zielonka. Notes on finite asynchronous automata. *ITA*, 21(2):99–135, 1987.