Causally Deterministic Markov Decision Processes

S. Akshay
Indian Institute of Technology Bombay, Mumbai, India

Tobias Meggendorfer
Lancaster University Leipzig, Germany

P. S. Thiagarajan
The University of North Carolina at Chapel Hill, NC, USA
Chennai Mathematical Institute, India

Abstract

Probabilistic systems are often modeled using *factored* versions of Markov decision processes (MDPs), where the states are composed out of the local states of components and each transition involves only a small subset of the components. Concurrency arises naturally in such systems. Our goal is to exploit concurrency when analyzing factored MDPs (FMDPs). To do so, we first formulate FMDPs in a way that aids this goal and port several notions from concurrency theory to the probabilistic setting of MDPs. In particular, we provide a concurrent semantics for FMDPs based on the classical notion of event structures, thereby cleanly separating causality, concurrency, and conflicts that arise from stochastic choices. We further identify the subclass of *causally deterministic* FMDPs (CMDPs), where non-determinism arises solely due to concurrency. Using our event structure semantics, we show that in CMDPs, *local* reachability properties can be computed using a "greedy" strategy. Finally, we implement our ideas in a prototype and apply it to four models, confirming the potential for substantial improvements over state-of-the-art methods.

2012 ACM Subject Classification Theory of computation → Concurrency

Keywords and phrases MDPs, distribution, causal determinism

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2024.6

Supplementary Material *Software (Artifact)*: `https://zenodo.org/records/12657579` [23]

Funding *S. Akshay*: Partially supported by Google India Research Award 2023 and SBI Foundation Hub for Data and Analytics.

1 Introduction

Factored versions of systems often constitute an important subclass. Two typical, well known examples – among very many – are Petri nets (and related models of concurrency) [26] and dynamic Bayesian networks [17]. A common key feature is that a state of the system is a *vector* of local component states. Further, a transition only involves a small subset of the components and hence can be specified succinctly; so much so, the size of the induced global system will often be exponential in the size of the factored presentation. This allows to model *large* systems without having to enumerate the set of global states and transitions explicitly.

Here, we explore this idea in the probabilistic setting of Markov decision processes (MDPs). Our starting point is a variant of factored MDPs (FMDPs). These are made up of several individual components, and a vector of local states constitutes the global state. Moreover, each action is associated with a fixed set of components named its *locations*. The availability of an action at a global state only depends on the local states of its locations and the stochastic changes that take place when an action occurs only involve the states of its locations. The resulting transition relation can be easily converted into the usual presentation of factored MDPs in the literature [4, 14]. Notably, our version of FMDPs includes models

specified in the established PRISM language [18] and JANI [5]. When handling systems with a large number of components, a key challenge is to analyze the global behavior in terms of the factored presentation instead of first explicitly constructing the global behavior. In the case of factored MDPs, this is particularly difficult due to the complex interplay between non-deterministic, stochastic and concurrent features of the dynamics.

As a first step toward addressing this challenge, we focus on the analysis of a subclass of FMDPs, called *causally deterministic* FMDPs (CMDPs). The defining feature of CMDPs is that any two actions that are available at a global state will have disjoint sets of locations. As a result, the two actions will be *causally independent*: executing one of them will not affect the availability of the other or its outcomes. Consequently, CMDPs admit a powerful partial order based analysis technique for verifying certain "robust" probabilistic properties. In the current paper, we focus on local reachability properties.

As a key tool to analyzing CMDPs, we identify the notion of *complete* strategies, which can be explained as follows. In a CMDP, the role of a strategy is to resolve the non-determinism that arises in the dynamics due to causally independent actions. This means a strategy linearizes a partially ordered set of action occurrences. Hence, if an action a is enabled at a state s, and is not chosen along a finite sequence of moves leading to the state s', then a will still be available at s'. Accordingly, a complete strategy is defined to be one in which the set of trajectories along which an available action is ignored forever has probability measure 0. Based on this notion, our main technical results for CMDPs are that (i) complete strategies suffice to obtain the optimal (maximal) probability of a local reachability property and (ii) *all* complete strategies will yield the same maximal probability value. Consequently we can choose a greedy complete strategy which avoids visiting many "useless" states. As the experimental results in Sec. 6 show, for CMDPs, our method vastly outperforms established, highly optimized tools such as Storm [8].

We establish these properties by exploiting fundamental objects drawn from concurrency theory, namely Mazurkiewicz traces [9] and prime event structures [24]. In particular, we develop an event structure semantics for *all* FMDPs. Since they arise in the context of FMDPs, the events in the event structure will have probability values assigned to them in a natural manner. We then use these probabilistic events to show that all complete strategies yield the same maximal probability values for local reachability properties. We view the present work as a first step towards developing partial-order reductions for FMDPs in general. Specifically, via the event structure semantics, based on Mazurkiewicz traces, a variety of techniques such as finite prefixes of event structures [11], and partial order reduction notions such as ample sets [13] and stubborn sets [15] can be brought to bear when analyzing FMDPs.

To summarize, our contributions are:

1. A novel class of factored MDPs, called CMDPs, in which the non-determinism between actions arises solely due to their causal independence.
2. An event structure semantics for FMDPs that cleanly separates causality, concurrency, and (stochastic) conflicts arising in the global behavior of an FMDP.
3. The identification of complete strategies for CMDPs which have the crucial properties; (i) they suffice to attain the optimal probability values for local reachability properties and (ii) all of them yield the same optimal value.
4. A prototypical implementation of (i) a syntactical over-approximation for checking that the input MDP is a CMDP and (ii) a greedy complete strategy accompanied by an experimental evaluation on four models. Comparison with existing state-of-the-art tools, e.g., Storm [8], shows a vast performance improvement for the evaluated models, highlighting the potential benefits of our approach.

Structure. In the rest of this section we review related work. We then present basic material concerning Markov chains and Markov decision processes. In Sec. 3, we introduce our class of FMDPs and the subclass of causally deterministic FMDPs (CMDPs). In the subsequent section we construct the event structure representation of our FMDPs which then leads to the main results developed in Sec. 5. The greedy strategy, its implementation, and the experimental results are presented in Sec. 6. The paper concludes with Sec. 7.

Related Work. Factored MDPs have been long studied in the literature [4, 14], where the transition relation is usually presented using a two layer dynamic Bayesian network. With an eye toward learning applications, a reward function is also included. Our formulation of FMDPs is geared towards capturing the distributed dynamics of FMDPs and hence is based on the notion of locations. Further, reward functions play no role in the present setting.

In the verification setting, several works have considered compositional methods to reason about large MDPs that are "factorized" via compositional operations. While some approaches use bisimulation based equivalences [12], others use abstractions [16], and yet others use a category-theoretical view of MDPs [31]. In a sense, these represent an approach which is orthogonal to ours, which is grounded in FMDPs and focused on solving quantitative behavioral properties. There have also been works adapting partial-order reduction techniques to the probabilistic setting using ample sets [13] and stubborn sets [15]. Variants of these approaches are incorporated in state-of-the-art tools such as Storm [8] and PRISM [18]. However, these works deal with MDPs viewed as monolithic objects presented in terms of global states and transitions. Thus, it will be difficult – if not impossible – to deal with the large MDPs that are presented succinctly as FMDPs. Furthermore, the focus in these works is on model checking linear time and branching time (probabilistic) properties using a semantically defined notion of commutability of actions along an execution sequence. These techniques do not enable one to compute optimal values of local reachability properties that we achieve using the event structure semantics. It will however be interesting to explore these methods in the context of CMDPs and, more generally, FMDPs.

Similarly, [7] exploit a model consisting of purely probabilistic components, however they use these components only to obtain a compact symbolic representation of the global MDP; in the end, they still work with the entire global MDP. In contrast, our analysis method directly works with the factored representation of the global MDP.

Several studies start with event structures, adjoin probabilities to events and study the resulting objects from a theoretical standpoint [1, 30]. However, in these studies probabilities are introduced in an ad-hoc manner and no attempt is made to establish a verification framework for an associated system model. In sharp contrast, the probabilities attached to the events in our event structures arise naturally from the associated MDPs. Furthermore, our use of event structures is firmly grounded in a verification framework for CMDPs.

Generalized stochastic Petri nets (GSPN) [20], despite being based on Petri nets, do not exploit concurrency and instead focus on their interleaved global behaviors in terms of (continuous time) Markov chains. A variant called Markov decision Petri nets is proposed in [3] as a high level modeling formalism. Their global behaviors are captured by MDPs and analyzed using symbolic representations. Here again concurrency essentially plays no role.

Finally, distributed Markov chains (DMCs) studied in [28, 29] have a similar flavour to CMDPs. DMCs consist of a network of probabilistic transition systems that synchronize on common actions with a sufficiently strong *syntactic* restriction ensuring that if two actions are enabled at a global state then they must involve disjoint sets of components. In addition,

they focus on statistical model checking of properties specified in a variant of bounded linear temporal logic. In contrast, CMDPs are a natural *behavioral* subclass of MDPs and our focus is determining the *exact* maximal probability of (unbounded) local reachability properties.

2 Preliminaries

A *Markov chain (MC)* (e.g., [2]), is a tuple $\mathsf{M} = (S, \hat{s}, P)$, where S is a (countable) set of *states*, $\hat{s} \in S$ is the *initial state*, and $P : S \to \mathcal{D}(S)$ is a *transition function* that for each state s yields a probability distribution over successor states, where $\mathcal{D}(S)$ is the set of distributions over S. A *Markov decision process (MDP)* (e.g., [25]) is a tuple $\mathcal{M} = (S, \hat{s}, Act, \mathsf{P})$, where S is a (finite) set of states, $\hat{s} \in S$ is the initial state, Act a finite set of actions, overloaded as $Act(s) \subseteq Act$ specifying available actions at a state s, and $\mathsf{P} : S \times Act \to \mathcal{D}(S)$ yielding a distribution over successors for each $s \in S$ and $a \in Act(s)$. For simplicity, we write $P(s, s')$ instead of $P(s)(s')$ for a MC and $\mathsf{P}(s, a, s')$ instead of $\mathsf{P}(s, a)(s')$ for an MDP.

Paths. An infinite path in an MC M is an infinite sequence $\varrho = s_1 s_2 \ldots$ where $s_1 = \hat{s}$ and $P(s_i, s_{i+1}) > 0$ for all i. A finite path ρ is a finite prefix of an infinite path. A Markov chain $\mathsf{M} = (S, \hat{s}, P)$ naturally induces a unique probability measure Pr_{M} over the σ-algebra generated by the cylinder sets induced by the finite paths [2, Sec. 10.1]. Similarly, for an MDP $\mathcal{M}$, an infinite path is a sequence $\varrho = s_1 a_1 s_2 a_2 \ldots$ such that $s_1 = \hat{s}$ and for all i we have $a_i \in Act(s_i)$ and $\mathsf{P}(s_i, a_i, s_{i+1}) > 0$. A finite path is a finite prefix of an infinite path *ending in a state*. We write $\mathsf{FPaths}_{\mathcal{M}}$ to denote the set of finite paths in $\mathcal{M}$. Moreover, $|\rho| = k$ denotes the length of a path (setting it to ∞ for infinite paths) and we define it to be the number of actions (transitions) that appear in the path. For $i \leq |\rho|$ we write ρ_i to refer to the i-th state in a path. Finally, $\mathrm{last}(\rho) = \rho_{|\rho|}$ denotes the last state in a finite path.

Strategies. Intuitively, in every state s, an action a from $Act(s)$ is chosen and the system advances to a successor state s' according to the probability distribution given by $\mathsf{P}(s, a)$. Starting from the initial state $\hat{s}$ and repeating this process indefinitely yields an infinite path. The way actions are chosen along an infinite path is captured by *strategies*. Specifically, a strategy is function mapping each finite path to one of the actions, say a, available in the last state, say s, of the path. This leads to new states chosen according to the distribution $\mathsf{P}(s, a)$. We let Π refer to the set of all strategies. To support our technical constructions arising later, our strategies are thus deterministic but not necessarily memoryless. A strategy is *memoryless* (or *positional*) if it only depends on the current state, i.e. $\pi(\rho) = \pi(\rho')$ whenever $\mathrm{last}(\rho) = \mathrm{last}(\rho')$. As usual, a strategy π induces the Markov chain $\mathcal{M}^\pi = (\mathsf{FPaths}_{\mathcal{M}}, \hat{s}, P^\pi)$, where for $\rho \in \mathsf{FPaths}_{\mathcal{M}}$ with $s = \mathrm{last}(\rho)$ and $a = \pi(\rho) \in Act(s_n)$ we set $P^\pi(\rho, \rho a s') = \mathsf{P}(s, a, s')$. We write $\mathsf{Pr}^\pi_{\mathcal{M}, \hat{s}} = \mathsf{Pr}_{\mathcal{M}^\pi, \hat{s}}$ for the induced probability measure.

Reachability. Fix an MDP $\mathcal{M} = (S, \hat{s}, Act, \mathsf{P})$. Then, *(unbounded) reachability* for a set of target states $T \subseteq S$ is the set of all (infinite) paths which eventually visit one of the target states, i.e. $\Diamond T = \{\varrho \mid \exists i.\ \varrho_i \in T\}$, which is measurable [2, Sec. 10.1.1]. For a strategy π, the probability of reaching T according to π is the probability assigned to this set of infinite paths $\Diamond T$ in $\mathcal{M}^\pi$, i.e. $\mathsf{Pr}^\pi_{\mathcal{M}}[\Diamond T]$. However, different strategies will in general yield different probabilities and one is often interested in the *maximum* of these probabilities. In other words, the goal is to determine $\sup_{\pi \in \Pi} \mathsf{Pr}^\pi_{\mathcal{M}}[\Diamond T]$ (also called the *value*). For MDPs, a well-known result states that it suffices to consider memoryless deterministic strategies for this maximization (see, e.g., [2, Lem. 10.102]).

3 Factored MDPs and Causal Determinacy

Often, an MDP comprises interacting components (or agents, processes). In particular, many modelling formalisms used in practice, e.g. the PRISM language [18] or JANI [5], define MDPs in this manner. Consequently, a state of the MDP will consist of a tuple of local states of the component processes. Further, an action will often involve only a fixed subset of the components leading to a stochastic transformation of the states of these components while the states of the other components are left untouched. We propose to use *factored* MDPs to study such systems.

Accordingly, let $Proc$ denote a finite set of components. Each component $p \in Proc$ has a set of *local states* denoted as S_p. This gives rise to the set of *global states* $\mathbf{S} = \prod_{p \in Proc} S_p$. To capture the idea that an action involves only a fixed subset of components, we use the *location map* $\mathrm{loc} : Act \to 2^{Proc} \setminus \emptyset$ to specify for each action a the set of components that participate in a. For convenience, we also identify a global state $\mathbf{s}$ with the map $\mathbf{s} : Proc \to \bigcup_{p \in Proc} S_p$ such that $\mathbf{s}(p) \in S_p$ for $p \in Proc$. Then, for a set of components $P \subseteq Proc$, we let $\mathbf{s}[P]$ denote the map $\mathbf{s}$ restricted to P. In other words, $\mathbf{s}[P] \in \prod_{p \in P} S_p$ and $\mathbf{s}[P](p) = \mathbf{s}(p)$ for $p \in P$. For $a \in Act$, we often write $\mathbf{s}[a]$ instead of $\mathbf{s}[\mathrm{loc}(a)]$ and call it the a-state induced by $\mathbf{s}$. This leads to $\mathbf{S}[a] = \{\mathbf{s}[a] \mid \mathbf{s} \in \mathbf{S}\}$, the set of a-states. With these notations at hand, we introduce factored MDPs (FMDPs).

▶ **Definition 1.** *A* factored MDP $\mathcal{M}$ *is a tuple* $(\{S_p\}_{p \in Proc}, \{\hat{s}_p\}_{p \in Proc}, Act, \mathrm{loc}, \{\mathsf{P}_a\}_{a \in Act})$ *where (i) Proc is a finite, non-empty set of components, (ii)* S_p *is a finite, non-empty set of states for each* $p \in Proc$*, (iii)* $\hat{s}_p \in S_p$ *is the initial state of component p, inducing the global initial state* $\hat{\mathbf{s}}$ *with* $\hat{\mathbf{s}}(p) = \hat{s}_p$ *for each* $p \in Proc$*, (iv) Act is a finite, non-empty set of actions, (v)* $\mathrm{loc} : Act \to 2^{Proc} \setminus \{\emptyset\}$ *is the locations map, and (vi) for each* $a \in Act$*,* $\mathsf{P}_a : \mathbf{S}[a] \to \mathcal{D}(\mathbf{S}[a])$ *is a (partial) transition function.*

Similar to MDPs, we write $\mathsf{P}_a(\mathbf{u}, \mathbf{v})$ instead of $\mathsf{P}_a(\mathbf{u})(\mathbf{v})$. The FMDP $\mathcal{M}$ induces an MDP called its global MDP defined as follows.

▶ **Definition 2.** *Let* $\mathcal{M}$ *be an FMDP as above. Then its* global MDP *is given by* $\widehat{\mathcal{M}} = (\mathbf{S}, \hat{\mathbf{s}}, Act, \mathsf{P})$ *where (i)* $a \in Act(\mathbf{s})$ *iff* $\mathsf{P}_a(\mathbf{s}[a])$ *is defined, and (ii) for every* $\mathbf{s}' \in \mathbf{S}$ *and* $a \in Act(\mathbf{s})$*, we have* $\mathsf{P}(\mathbf{s}, a, \mathbf{s}') = v > 0$ *iff* $\mathsf{P}_a(\mathbf{s}[a], \mathbf{s}'[a]) = v$ *and* $\mathbf{s}[Proc \setminus \mathrm{loc}(a)] = \mathbf{s}'[Proc \setminus \mathrm{loc}(a)]$.

We can immediately verify that $\widehat{\mathcal{M}}$ is indeed an MDP. Moreover, $\widehat{\mathcal{M}}$ has two important properties, namely: (F1) The availability of an action a at a state $\mathbf{s}$ depends only on $\mathbf{s}[a]$. Further, when an action a at occurs at a state $\mathbf{s}$, the changes it produces involve only the components in $\mathrm{loc}(a)$; the local states of components in $Proc \setminus \mathrm{loc}(a)$ remain unchanged. (F2) When action a occurs at a global state $\mathbf{s}$, the changes it produces (to the states of participating components) depends only on the a-state $\mathbf{s}[a]$. In particular, suppose $\mathbf{s}_1$ and $\mathbf{s}_2$ are global states and $a \in Act$ is an action where $\mathbf{s}_1[a] = \mathbf{s}_2[a]$. Then, if $\mathsf{P}(\mathbf{s}_1, a, \mathbf{s}_1') = v > 0$ there exists a unique global state $\mathbf{s}_2'$ such that $\mathsf{P}(\mathbf{s}_2, a, \mathbf{s}_2') = v$ and $\mathbf{s}_1'[a] = \mathbf{s}_2'[a]$.

Before presenting an illustrative example, we briefly remark on this defining way of defining an FMDP and how it relates to established notions.

▶ Remark 3. Traditionally, FMDPs are defined using a transition relation represented by a two-layer dynamic Bayesian network [4, 14]. We have chosen to use a slightly different definition, aligned with concurrency theory, so that the distributed nature of the dynamics can be clearly brought out, as we shall see below. However, our theory is neutral to *how* the dynamics of the individual components are represented as long as the global transitions

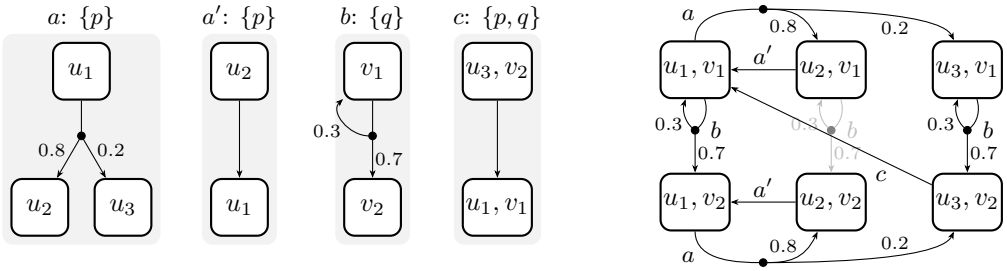


Figure 1 This figure illustrates a two-component FMDP where $Proc = \{p, q\}$, $S_p = \{u_1, u_2, u_3\}$, $S_q = \{v_1, v_2\}$, and $Act = \{a, a', b, c\}$. On the left, for each action a both $\mathrm{loc}(a)$ and P_a are depicted. On the right, the induced global MDP is shown. The middle b action is greyed out solely for readability, it is not special in any way. We omit the probability label if it is 1.

are factored in terms of the components participating in the actions. In particular, once the properties (F1) and (F2) stated above are satisfied by the resulting global MDP, our theory is applicable to any model which exhibits such behaviour, e.g. the DBN-based definitions.

▶ **Example 4.** In Fig. 1, an example of an FMDP (on the left) and its induced global MDP (on the right) is shown. To explain the relation between FMDP and global MDP, we write $\langle u_1, u_2\rangle$ and similar to denote global states and c-states as tuples of local states, as the correspondence with the local states of the components is clear. The a-transition from $\langle u_1\rangle$ to $\langle u_2\rangle$ in the FMDP implies a is available at the global state $\langle u_1, v_1\rangle$ since $\langle u_1, v_1\rangle[a] = \langle u_1\rangle$ and $P_a(\langle u_1\rangle)$ is defined in the FMDP. Further, $\mathsf{P}(\langle u_1, v_1\rangle, a, \langle u_2, v_1\rangle) = 0.8$ as $\mathsf{P}_a(\langle u_1\rangle, \langle u_2\rangle) = 0.8$. In particular, this transition does not modify the state of the q-component since $\mathrm{loc}(a) = \{p\}$. The other transitions shown in the global MDP can be inferred using similar reasoning.

By slight abuse of notation, in the following we write $\mathbf{S}$ to denote the set of *reachable* states, defined in the obvious way. We also identify the FMDP with its induced global MDP and freely go back and forth between the two notions and the associated notations. Finally, for simplicity we assume that the FMDPs we deal with are free of *deadlocks*, i.e. if $\mathbf{s} \in \mathbf{S}$ then $Act(\mathbf{s}) \neq \emptyset$. (Since our focus is on reachability, this can be ensured by adding a new component d with a single state s_d, a new action a_d with $\mathrm{loc}(a_d) = \{d\}$, and $\mathsf{P}_{a_d}(\langle s_d\rangle, a_d, \langle s_d\rangle) = 1$.)

3.1 Local Reachability

Let $\mathcal{M}$ be an FMDP. Then, a *local* reachability problem is specified by $T \subseteq S_p$ for some component p. Let $\mathbf{T} = \{\mathbf{s} \mid \mathbf{s}(p) \in T\}$ the corresponding global reachability set. The goal is to determine the probability $\sup_{\pi\in\Pi}\mathsf{Pr}^{\pi}_{\mathcal{M}}[\Diamond\mathbf{T}]$.

Local reachability for an FMDP can be solved by ignoring its factored nature and instead treat it as a "global" reachability problem on the induced global MDP. In this case, classical approaches as employed by PRISM [18] and Storm [8] can be used. This problem is in PTIME [2, Cor. 10.107], but *in the size of the global MDP*, which can easily be exponential in the size of the FMDP. Our goal is to mitigate this state-space explosion by exploiting the partially ordered nature of the dynamics of the model.

3.2 Causal Determinacy and Complete Strategies

As a first step, we shall tackle the state explosion problem by considering the subclass of FMDPs in which the sole source of non-determinism is from the *causal independence* of actions. This idea can be captured through a natural restriction.

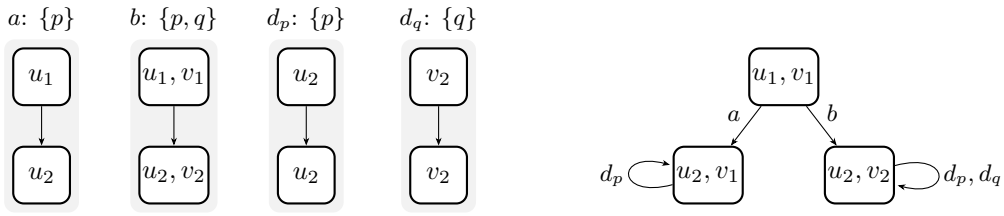


Figure 2 This figure illustrates an FMDP which is not CD. We have $Proc = \{p, q\}$, $S_p = \{u_1, u_2\}$, $S_q = \{v_1, v_2\}$, and $Act = \{a, b, d_p, d_q\}$. On the left, we depict the transition function P_a for all $a \in Act$ and to the right the induced global MDP.

▶ **Definition 5.** *An FMDP $\mathcal{M}$ is* causally deterministic *(CD) if for every (reachable) state* $\mathbf{s}$ *and* $a, b \in Act(\mathbf{s})$ *with* $a \neq b$ *we have* $\mathrm{loc}(a) \cap \mathrm{loc}(b) = \emptyset$. *We call such an FMDP a* CMDP.

▶ **Example 6.** The FMDP shown in Fig. 1 is causally deterministic: In any global state, the available set of actions is $\{a, b\}$, $\{a', b\}$ or $\{c\}$. In contrast, the FMDP of Fig. 2 is not CD. In $\langle u_1, v_1\rangle$ both a and b are available, but $\mathrm{loc}(a) \cap \mathrm{loc}(b) = \{p\} \neq \emptyset$. And indeed, it is relevant whether we choose a or b. For example, a leads to a state in which b is not enabled anymore, and, in particular, v_2 is not reachable, while b reaches v_2 with probability 1.

▶ Remark 7. Causal determinacy is intrinsically a concurrency based notion. If $a, b \in Act(\mathbf{s})$ with $a \neq b$ then a and b can occur independent of each other at $\mathbf{s}$. In fact, suppose $\mathbf{s}_0 a_1 \mathbf{s}_1 \cdots \mathbf{s}_{n-1} a_n \mathbf{s}_n$ is a finite path, $b \in Act(\mathbf{s}_0)$ and $a_i \neq b$ for $1 \leq i \leq n$. Then $\mathrm{loc}(a_i) \cap \mathrm{loc}(b) = \emptyset$ for all $1 \leq i \leq n$ and $b \in Act(\mathbf{s}_n)$. This follows from the fact that a CMDP is an FMDP and hence $\mathbf{s}_0[b] = \mathbf{s}_n[b]$. This basic feature of a CMDP leads to a partial-order based technique using which one can often efficiently verify many behavioral properties that are "robust" with respect to interleavings of partially ordered behaviors, such as local reachability. Due to space considerations, we will not pause to formalize the notion of robust properties since it is not needed to establish our results.

▶ Remark 8. Deciding whether the global MDP induced by an FMDP (encoded in a standard manner) is CD is in PSPACE. The idea is to convert the probabilistic transitions to non-deterministic ones, and reduce the CD property to a reachability property of the resulting 1-safe Petri net, known to be in PSPACE [10]. However, given our main goals, establishing this result in detail would be a digression and hence we do not do so. That said, for practical purposes, we later discuss a simple, sufficient syntactic condition allowing us to over-approximate CD in our case studies.

As a central tool to exploit causal determinacy, we introduce *complete* strategies.

▶ **Definition 9.** *Let ρ be an infinite path in a Markov chain $\mathcal{M}^\pi$ induced by a strategy π on a CMDP $\mathcal{M}$. Then, ρ is a* complete path *iff for every $i \geq 0$, if $a \in Act(\rho_i)$ then there exists $j \geq i$ such that $\pi(\rho_i \rho_{i+1} \dots \rho_j) = a$. In other words, if a is available at ρ_i then it is eventually chosen by the strategy along the path (where it will remain available due to CD).*

Let Υ be the set of complete paths in $\mathcal{M}^\pi$. A strategy π is complete *iff $\mathsf{Pr}^\pi_{\mathcal{M}}[\Upsilon] = 1$.*

Thus, incomplete paths may be present in $\mathcal{M}^\pi$, but the collection of such paths has measure 0 and does not contribute to the reachability probabilities of interest. Note that Υ is measurable as it can be written as countable intersections and unions of cylinder sets.

First, we show here that it suffices to consider only complete strategies for local reachability. Later we will show that *all* complete strategies will yield the same, maximal probability value. Consequently, we can freely choose a "greedy" strategy with which the maximal probabilities can be computed in an efficient manner.

▶ **Lemma 10.** *There exists a deterministic, complete strategy* $\pi \in \Pi$ *which achieves the optimal value, i.e.* $\mathsf{Pr}^{\pi}_{\mathcal{M}}[\Diamond \mathbf{T}] = \sup_{\pi' \in \Pi} \mathsf{Pr}^{\pi'}_{\mathcal{M}}[\Diamond \mathbf{T}]$.

Proof Sketch. We delegate the (rather routine) proof to App. B. For a sketch, we show that any (optimal, memoryless) strategy can be extended to a complete strategy without reducing the reachability probability it achieves. Intuitively, the modified strategy waits until the original strategy has visited all the states it will ever visit (thus any goal it might reach is already reached), which happens with probability 1, and then switches to a "complete" mode in which it plays all the actions that have not been played since they became available. ◀

In the next section, we develop the event structure semantics for FMDPs. Using this, we show in Sec. 5 that any two complete strategies achieve the *same* value.

4 An Event Structure Semantics for FMDPs

4.1 Mazurkiewicz Trace Languages

We first associate a Mazurkiewicz trace language with an FMDP. Then, using a standard construction, we obtain the event structure representation. We recall from [21] a Mazurkiewicz trace alphabet is a pair (Σ, I) where Σ is a finite non-empty alphabet and $I \subseteq \Sigma \times \Sigma$ is an irreflexive and symmetric relation called the *independence* relation. When describing the executions of a distributed system, Σ is the set of actions and $a\ I\ b$ asserts that the actions a and b are "causally" independent. In other words, they can be executed in any order when they are both enabled. We define $D = (\Sigma \times \Sigma) \setminus I$ to be the *dependency* relation. The relation I induces in a natural way the equivalence relation $\approx_I\ \Sigma^* \times \Sigma^*$. It is the least equivalence satisfying $\sigma ab \sigma' \approx_I \sigma ba \sigma'$ for $a\ I\ b$. For $\sigma \in \Sigma^*$, $[\sigma]$ denotes the $\approx_I$-equivalence class containing σ, often called a *Mazurkiewicz trace*. It corresponds to the set of all possible interleavings of a unique partially ordered set of actions. A Mazurkiewicz trace language is a subset of $\{[\sigma] \mid \sigma \in \Sigma^*\}$, i.e. a set of Mazurkiewicz traces. For convenience, we abbreviate Mazurkiewicz traces (Mazurkiewicz trace languages) as traces (trace languages).

4.2 The Mazurkiewicz Trace Language of an FMDP

To define the trace language of an FMDP we start with $\mathcal{M}$-events.

▶ **Definition 11.** *Let* $\mathcal{M} = (\{S_p\}_{p \in Proc}, \{\hat{s}_p\}_{p \in Proc}, Act, \mathrm{loc}, \{\mathsf{P}_a\}_{a \in Act})$ *be an FMDP. Then* $\alpha = (\mathbf{u}, a, \mathbf{v})$ *is an* $\mathcal{M}$*-event if* $\mathsf{P}_a(\mathbf{u}, \mathbf{v}) > 0$. *We define the probability of* α *as* $Pr(\alpha) = \mathsf{P}_a(\mathbf{u}, a, \mathbf{v})$. *Furthermore, we set* $act(\alpha) = a$ *and* $\mathrm{loc}(\alpha) = \mathrm{loc}(a)$.

The $\mathcal{M}$-event $\alpha = (\mathbf{u}, a, \mathbf{v})$ comprises the a-state that must hold at a state $\mathbf{s}$ for it to occur (i.e. $\mathbf{s}[a] = \mathbf{u}$). It also reports the a-state that is chosen with probability $Pr(\alpha)$ resulting in the global state $\mathbf{s}'$ (i.e. $\mathbf{s}'[a] = \mathbf{v}$ and $\mathbf{s}[Proc \setminus \mathrm{loc}(a)] = \mathbf{s}'[Proc \setminus \mathrm{loc}(a)]$). For instance, $\alpha = (\langle u_1 \rangle, a, \langle u_2 \rangle)$ is an $\mathcal{M}$-event in the FMDP shown in Fig. 1 with $Pr(\alpha) = 0.8$.

$\mathcal{M}$-events naturally give rise to the transition relation $\longrightarrow_{\mathcal{M}}$ over $\mathbf{S}$, defined as follows. Suppose $\alpha = (\mathbf{u}, a, \mathbf{v})$, and $\mathbf{s}, \mathbf{s}' \in \mathbf{S}$. Then $\mathbf{s} \xrightarrow{\alpha}_{\mathcal{M}} \mathbf{s}'$ if $\mathbf{s}[a] = \mathbf{u}$, $\mathbf{s}'[a] = \mathbf{v}$, and $\mathbf{s}[Proc \setminus \mathrm{loc}(a)] = \mathbf{s}'[Proc \setminus \mathrm{loc}(a)]$. As usual, we write $\longrightarrow$ instead of $\longrightarrow_{\mathcal{M}}$. We now define an $\mathcal{M}$-path to be a sequence $\mathbf{s}_0 \alpha_1 \mathbf{s}_1 \alpha_2 \cdots \mathbf{s}_{n-1} \alpha_n \mathbf{s}_n$ such that (i) $\mathbf{s}_0 = \widehat{\mathbf{s}}$ and (ii) $\mathbf{s}_{i-1} \xrightarrow{\alpha_i} \mathbf{s}_i$ for $1 \leq i \leq n$. In essence, $\mathcal{M}$-paths correspond to finite paths in the global MDP. Since we only deal with $\mathcal{M}$-paths from now on, we say "path" instead of $\mathcal{M}$-path henceforth.

Let $\Sigma_{\mathcal{M}}$ denote the set of $\mathcal{M}$-events. In the following, we instead write Σ, as $\mathcal{M}$ will be clear from the context. Moreover, we set $\Sigma_p = \{\alpha \mid \alpha \in \Sigma, p \in \mathrm{loc}(\alpha)\}$ for each component p. We define the independence relation $I \subseteq \Sigma \times \Sigma$ as $I = \{(\alpha, \beta) \mid \mathrm{loc}(\alpha) \cap \mathrm{loc}(\beta) = \emptyset\}$. Clearly, I is irreflexive and symmetric, and hence (Σ, I) is a trace alphabet. Next, for $\Sigma' \subseteq \Sigma$ let $prj_{\Sigma'} : \Sigma^* \to \Sigma'^*$ be the projection which from sequences in Σ^* erases all appearances of letters that are not in Σ'. We abbreviate prj_{Σ_p} as prj_p. This leads to the equivalence relation $\approx_I$ over Σ^* given by $\sigma \approx_I \sigma'$ iff for every $p \in Proc$ we have $prj_p(\sigma) = prj_p(\sigma')$. Effectively, two traces are equivalent if no single component can differentiate between them. We define $\approx_I$ in this way instead of using the usual partial commutative relation, as it extends smoothly to infinite $\mathcal{M}$-event sequences. For convenience, we write from now on $\approx$ instead of $\approx_I$.

Let $\sigma = \alpha_1\alpha_2 \cdots \alpha_n \in \Sigma^*$. Then σ is an *$\mathcal{M}$-event sequence* of $\mathcal{M}$ if there exists states $\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_n$ such that $\mathbf{s}_0\alpha_1\mathbf{s}_1 \cdots \mathbf{s}_{n-1}\alpha_n\mathbf{s}_n$ is an $\mathcal{M}$-path. We let $L^{seq}_{\mathcal{M}}$ denote the set of $\mathcal{M}$-event sequences of $\mathcal{M}$. This leads to the trace language of $\mathcal{M}$ given by $L_{\mathcal{M}} = \{[\sigma] \mid \sigma \in L^{seq}_{\mathcal{M}}\}$.

We next introduce some terminology to aid in the construction of the event structure representation of $\mathcal{M}$. These notions are generic to the theory of Mazurkiewicz trace languages. However, for convenience, we introduce them in the context of $L_{\mathcal{M}}$. First, $\sqsubseteq \subseteq L_{\mathcal{M}} \times L_{\mathcal{M}}$ is given by $[\sigma] \sqsubseteq [\sigma']$ iff $prj_p(\sigma)$ is a prefix of $prj_p(\sigma')$ for every $p \in Proc$. Clearly, $\sqsubseteq$ is a well-defined partial ordering relation. Next, suppose $[\sigma], [\sigma'] \in L_{\mathcal{M}}$. Then $[\sigma] \uparrow [\sigma']$ iff there exists $[\sigma''] \in L_{\mathcal{M}}$ such that $[\sigma] \sqsubseteq [\sigma'']$ and $[\sigma'] \sqsubseteq [\sigma'']$. Finally, $[\sigma] \in L_{\mathcal{M}}$ is a *prime trace* iff there exists an $\mathcal{M}$-event α such that $last(\sigma') = \alpha$ for every $\sigma' \in [\sigma]$ where $last(\tau)$ is the last letter of the non-null sequence τ.

There is a rich theory of Mazurkiewicz trace languages available, see e.g. [9]. Here we only use basic facts of the theory which we state below. The proofs are standard and can be assembled from [9, 27] and hence we omit them.

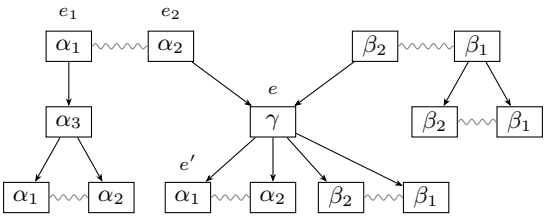
▶ **Proposition 12.** *It holds that (i) if $\sigma \approx \sigma'$ then $|\sigma| = |\sigma'|$, and (ii) $[\sigma] \uparrow [\sigma']$ iff there exist sequences σ'', σ_1 and σ'_1 such that (a) $\sigma \approx \sigma''\sigma_1$ and $\sigma' \approx \sigma''\sigma'_1$ and (b) $a\ I\ b$ for every letter a that appears in σ_1 and every letter b that appears in σ'_1.*

4.3 The Event Structure Representation of FMDPs

We begin by recalling from [24] that a prime *event structure* is a tuple $ES = (E, \leq, \#)$ where (i) E is a countable set of events, (ii) $\leq \subseteq E \times E$ is a partial ordering relation called the *causality relation*, and (iii) $\# \subseteq E \times E$ is an irreflexive and symmetric relation called the *conflict relation*. It is required that if $e \# e'$ and $e' \leq e''$ then $e \# e''$. Usually, a prime event structure is accompanied by a labelling function that relates a system to its event structure representation. In our case, there will be two such functions.

▶ **Definition 13.** *Let $\mathcal{M} = (\{S_p\}_{p \in Proc}, \{\hat{s}_p\}_{p \in Proc}, Act, \mathrm{loc}, \{\mathsf{P}_a\}_{a \in Act})$ be an FMDP. Its event structure is a tuple $ES_{\mathcal{M}} = (E, \leq, \#, \lambda, \mu)$ where $(E, \leq, \#)$ is a prime event structure where (i) $E = \{[\sigma] \in L_{\mathcal{M}} \mid [\sigma]$ is a prime trace$\}$, (ii) $\leq$ is $\sqsubseteq$ restricted to $E \times E$, (iii) $[\sigma] \# [\sigma']$ iff it is* not *the case that $[\sigma] \uparrow [\sigma']$, (iv) $\lambda : E \to \Sigma$ is the labelling function satisfying $\lambda([\sigma]) = last(\sigma)$, and (v) $\mu : E \to [0, 1]$ assigns to $e = [\alpha_1\alpha_2 \cdots \alpha_n] \in E$ the probability $\mu(e) = \prod_{1 \leq j \leq n} Pr(\alpha_j)$ (i.e. the probability of a prime trace is the product of the probabilities of the $\mathcal{M}$-events encountered along a sequence in the prime trace).*

In what follows we often write $\leq$ instead of $\sqsubseteq$ when viewing events as elements of E and not as traces. The "states" of an event structure are called *configurations* and the dynamics of $ES_{\mathcal{M}}$ is captured via a transition relation over its configurations.



- $\alpha_1 = (\{u_1\}, a, \{u_2\})$
- $\alpha_2 = (\{u_1\}, a, \{u_3\})$
- $\alpha_3 = (\{u_2\}, a', \{u_1\})$
- $\beta_1 = (\{v_1\}, b, \{v_1\})$
- $\beta_2 = (\{v_1\}, b, \{v_2\})$
- $\gamma = (\{u_3, v_2\}, c, \{u_1, v_1\})$
- $e = [\alpha_2\beta_2\gamma] = \{\alpha_2\beta_2\gamma, \beta_2\alpha_2\gamma\}$

Figure 3 This figure illustrates the initial fragment of the event structure representation for the FMDP depicted in Fig. 1.

▶ **Definition 14.** *For $c \subseteq E$, define $\downarrow c = \{y \mid \exists x \in c \text{ s.t. } y \leq x\}$. Then $c \subseteq E$ is a* configuration *iff $c = \downarrow c$ and $(c \times c) \cap \# = \emptyset$.*

We define $C_{\mathcal{M}}$ to be the set of *finite* configurations of $ES_{\mathcal{M}}$ and note that $\emptyset$ is a configuration. Let $c, c' \in C_{\mathcal{M}}$ and $\alpha \in \Sigma$. Then $c \xrightarrow{\alpha}_{ES} c'$ iff there exists $e \in E \setminus c$ such that $c \cup \{e\} = c'$ and $\lambda(e) = \alpha$. This basically says that an event e which is not in the configuration c can be added to it to obtain a larger configuration provided the past of e (under $<$) is contained in c. For simplicity, we write $\downarrow e$ instead of $\downarrow\{e\}$ for $e \in E$. Clearly, $\downarrow e$ is a configuration for every e in E.

In Fig. 3 we show the initial fragment of the event structure representation of the FMDP in Fig. 1. In order to minimize clutter, we have named the $\mathcal{M}$-events as α_1, α_2, etc. We note that $Pr(\alpha_1) = 0.8$, $Pr(\alpha_2) = 0.2$, $Pr(\beta_1) = 0.3$, and $Pr(\beta_2) = 0.7$. Further, $Pr(\alpha_3) = 1 = Pr(\gamma)$. In the diagram, the directed arrow represent the *immediate* causality relation $\lessdot$ where $e \lessdot e'$ iff $e < e'$ and for every e'', $e \leq e'' \leq e'$ implies $e = e''$ or $e'' = e'$. The remaining members of the causality relation are obtained by taking the reflexive transitive closure of this relation. Similarly, the squiggly lines represent the *minimal* conflict relation $\widehat{\#}$ defined as $e \mathbin{\widehat{\#}} e'$ iff $e \# e'$ and $(\downarrow e \times \downarrow e') \cap \# = \{(e, e'), (e', e)\}$. Using the conflict inheritance requirement of an event structure, we can deduce all other members of the conflict relation. For example, in the event structure shown in Fig. 3, $e_3 \# e_4$ since $e_1 \# e_2 \leq e_4$ implies $e_1 \# e_4$ and since $e_1 \leq e_3$ and $\#$ is symmetric, we get $e_3 \# e_4$. In addition, we have listed the members of just one of the prime traces named e whose label is α_2. For the remaining events, we have just indicated their labels.

The behavior of $\mathcal{M}$ can be related to the behavior of $ES_{\mathcal{M}}$ as follows.

▶ **Proposition 15.** *Let $\mathcal{M}$ and $ES_{\mathcal{M}}$ be defined as above. Then the following statements hold.*

1. *Let $c = \{e_1, e_2, \ldots, e_n\}$ be a configuration such that $e_1 e_2 \cdots e_n$ is a linearization of the partial order $(c, \leq)$ where, by abuse of notation, $\leq$ also denotes the restriction of $\leq$ to $c \times c$. Then there exists $\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_n \in \mathbf{S}$ such that $\mathbf{s}_0 = \hat{\mathbf{s}}$ and $\mathbf{s}_0\lambda(e_1)\mathbf{s}_1\lambda(e_2)\mathbf{s}_2 \cdots \mathbf{s}_{n-1}\lambda(e_n)\mathbf{s}_n$ is a finite path in $\mathcal{M}$, which we shall call a c-*path (in $\mathcal{M}$).
2. *Let the function $state : C \to \mathbf{S}$ be given by (i) $state(\emptyset) = \hat{\mathbf{s}}$ and (ii) for a non-empty configuration c and c-path $\rho = \mathbf{s}_0\alpha_1\mathbf{s}_1 \cdots \mathbf{s}_n$ in $\mathcal{M}$, we define $state(c) = s_n$. Then, state is a well-defined map from C onto the set of reachable states of $\mathcal{M}$.*
3. *Let $c, c' \in C$ and $\alpha = (\mathbf{u}, a, \mathbf{v})$. Then $c \xrightarrow{\alpha}_{ES} c'$ iff $\mathsf{P}(state(c), \alpha, state(c')) = Pr(\alpha) > 0$.*
4. *Let $tr : C \to L_{\mathcal{M}}$ be the map given by (i) $tr(\emptyset) = \{\varepsilon\}$ and (ii) for a non-empty configuration c and c-path $\mathbf{s}_0\alpha_1\mathbf{s}_1 \cdots \mathbf{s}_n$ in $\mathcal{M}$ it is the case that $tr(c) = [\alpha_1\alpha_2 \cdots \alpha_n]$. Then, tr is well defined and a bijection.*

Most of these observations are standard [27] and directly carry over to our setting. The third part is specific to FMDPs but follows directly from the definition of an $\mathcal{M}$-event.

We close out this section with a useful result which will be needed in the next section. Let $\sigma = \alpha_1\alpha_2\cdots\alpha_n$ be an $\mathcal{M}$-event sequence in $\mathcal{M}$. Then $dg(\sigma)$, the subsequence of σ, is defined inductively by (i) $dg(\alpha_n) = \alpha_n$ and (ii) $dg(\alpha_{i-1}\alpha_i\cdots\alpha_n) = \alpha_{i-1}dg(\alpha_i\alpha_{i+1}\cdots\alpha_n)$ if there exists an $\mathcal{M}$-event β in $dg(\alpha_i\alpha_{i+1}\cdots\alpha_n)$ such that $\alpha_i\ D\ \beta$ and $dg(\alpha_i\alpha_{i+1}\cdots\alpha_n)$ otherwise. Basically, dg is the so called dependency graph that captures the causal past α_n in σ. We now define $ev(\alpha_1\alpha_2\cdots\alpha_n)$ to be the trace $= [dg(\alpha_1\alpha_2\cdots\alpha_n)]$. Finally, the relation $co \subseteq E \times E$ for the event structure $ES_{\mathcal{M}}$ is given by, $co = E \times E \setminus (\leq \cup \geq \cup \#)$. If $e\ co\ e'$ this can be interpreted as e and e' being causally independent.

▶ **Lemma 16.** *Let $\sigma = \alpha_1\alpha_2\cdots\alpha_n \in L^{seq}_{\mathcal{M}}$ a non-null $\mathcal{M}$-event sequence in $L^{seq}_{\mathcal{M}}$.*

1. *Then $ev(\sigma)$ is a prime trace and hence is an event in $ES_{\mathcal{M}}$.*
2. *Suppose that $e' \leq e$ in $ES_{\mathcal{M}}$. Then there exists a unique $i \in \{1, 2, \ldots, n-1\}$ such that $ev(\alpha_1\alpha_2\cdots\alpha_i) = e'$.*
3. *Suppose that $e' = ev(\alpha_1\alpha_2\cdots\alpha_i)$ for some $1 \leq i < n$. Then $e' \leq e$ or $e\ co\ e'$ in $ES_{\mathcal{M}}$.*
4. *Suppose that $\alpha_n = (\mathbf{u}, a, \mathbf{v})$ and $\sigma' = \alpha_1\alpha_2\cdots\alpha_{n-1}\alpha'_n$ such that $\alpha'_n = (\mathbf{u}, a, \mathbf{v}')$ and $\mathbf{v} \neq \mathbf{v}'$. Then $ev(\sigma)\ \#\ ev(\sigma')$ in $ES_{\mathcal{M}}$.*

The proof follows from [32]. The first part says that along a path in $\mathcal{M}$ every $\mathcal{M}$-event corresponds to the occurrence of an event in $ES_{\mathcal{M}}$. The second part says that every event e' that lies in the past of the event e represented by the $\mathcal{M}$-event sequence σ will appear as the event corresponding to a unique prefix of σ. The third part says if e corresponds to the $\mathcal{M}$-event sequence σ then every event that corresponds to a strict prefix of σ will either be causally earlier than e or will be causally independent of e in $ES_{\mathcal{M}}$. The last part says that two different stochastic choices made at a state along an $\mathcal{M}$-path will correspond to conflicting events in $ES_{\mathcal{M}}$.

▶ Remark 17. We conclude by noting that an event $e = [\alpha_1\alpha_2\ldots\alpha_n]$ in $ES_{\mathcal{M}} = (E, \leq, \#, \lambda, \mu)$ gets assigned a probability value via $\mu(e) = \prod_{1\leq i\leq n} Pr(\alpha_i)$. It is not difficult to provide a measure theoretic justification for this probability value by constructing a σ-algebra generated by the family of cylinder sets $\{CS(e)\}_{e\in E}$ where $CS(e) = \{c \in C^{\infty}_{\max} \mid \downarrow e \subseteq c\}$. Here, C^{∞} is the set of infinite configurations $ES_{\mathcal{M}}$ and $c \in C^{\infty}$ is maximal (i.e. $c \in C^{\infty}_{\max}$) iff $c \subseteq c' \in C^{\infty}$ implies $c = c'$. In other words, c cannot be extended to a larger (infinite) configuration. This distinction between infinite and maximal infinite configurations arises due to concurrency and corresponds to the distinction between complete and incomplete paths. We can define $Pr_{ES}(CS(e)) = \mu(e)$ and show that Pr_{ES} extends canonically to a probability measure over the σ-algebra generated by the above family of cylinder sets. We leave this construction for future work, since we merely need the probability values assigned to the events as common reference points to establish the main result of the next section, namely, all complete strategies determine the same probability values for local reachability properties.

5 The Key Result for CMDPs

Recall that we are given $T \subseteq S_p$ for some component p and aim to determine $\sup_{\pi\in\Pi}\mathsf{Pr}^{\pi}_{\mathcal{M}}[\Diamond\mathbf{T}]$, where $\mathbf{T} = \{\mathbf{s} \mid \mathbf{s}(p) \in T\}$. In Lem. 10, we argued that it suffices to consider complete strategies to achieve this. Here, we shall show that *all* complete strategies compute the same probability value for $\Diamond\mathbf{T}$. This allows us to choose a complete strategy greedily, which in turn enables us to efficiently compute the (optimal) probability of a local reachable set.

We first identify the set of events $E_{\Diamond T}$ in the event structure $ES_{\mathcal{M}}$, corresponding to paths in $\mathcal{M}$ reaching T. Let $e = [\alpha_1\alpha_2\cdots\alpha_n] \in E$ with $\alpha_j = (\mathbf{u}_j, a_j, \mathbf{v}_j)$ for $1 \leq j \leq n$. Then $e \in E_{\Diamond T}$ if $\mathbf{v}_n(p) \in T$ and $\mathbf{v}_i(p) \notin T$ for $1 \leq i < n$, in other words, when its last $\mathcal{M}$-event reaches a member of T and no earlier $\mathcal{M}$-event in the sequence representing e does so.

To establish the main goal of this section we proceed as follows. For the complete strategy π, we let $Paths^{\pi}_{comp}$ denote the set of complete paths of the Markov chain $\mathcal{M}^{\pi}$. We then identify, for a given $e \in E_{\diamond T}$, the set of finite paths $\text{PathReach}(\mathcal{M}^{\pi}, e)$ in $\mathcal{M}^{\pi}$ which are prefixes of complete paths and "reach" e. Specifically, suppose $\xi = \rho_0\alpha_1\rho_1\alpha_2\cdots\rho_{n-1}\alpha_n\rho_n$ is a path in $\mathcal{M}^{\pi}$. Then $\xi \in \text{PathReach}(\mathcal{M}^{\pi}, e)$ if (i) it is a prefix of a path in $Paths^{\pi}_{comp}$, (ii) $ev(\alpha_1\alpha_2\cdots\alpha_n) = e$ and (iii) no strict prefix of ξ satisfies (ii).

We first show that for each $e \in E_{\diamond T}$ it is the case that $\mu(e) = \mathsf{Pr}^{\pi}_{\mathcal{M}}[\bigcup_{\sigma\in\text{PathReach}(\mathcal{M}^{\pi},e)}\sigma]$. (Recall that $\mu(e)$ is the probability value assigned to e in $ES_{\mathcal{M}}$.) We then lift this result to $E_{\diamond T}$ and show that $\sum_{e\in E_{\diamond T}}\mu(e) = \sum_{e\in E_{\diamond T}}\mathsf{Pr}^{\pi}_{\mathcal{M}}[\text{PathReach}(\mathcal{M}^{\pi}, e)] = \mathsf{Pr}^{\pi}_{\mathcal{M}}[\diamond\mathbf{T}]$. Since these results apply to *every* complete strategy π, we are done.

5.1 Relating the Probability of e to the Probability of $\text{PathReach}(\mathcal{M}^{\pi}, e)$

Through this subsection, fix $e \in E_{\diamond T}$ and a complete strategy π. We wish to prove that $\mu(e) = \mathsf{Pr}^{\pi}_{\mathcal{M}}[\text{PathReach}(\mathcal{M}^{\pi}, e)]$. Our proof consists of three steps. First, we represent $\mathcal{M}^{\pi}$ as a transition system TS^{π} by labelling the transitions of the Markov chain with $\mathcal{M}$-events. Second, we represent $\text{PathReach}(\mathcal{M}^{\pi}, e)$ as a *finite* prefix of TS^{π}. Third, we use this finite prefix to establish that $\mu(e) = \mathsf{Pr}^{\pi}_{\mathcal{M}}[\text{PathReach}(\mathcal{M}^{\pi}, e)]$.

We begin by deriving the transition system TS^{π}. The states of TS^{π} are the states of $\mathcal{M}^{\pi}$ (i.e. finite paths in $\mathcal{M}$). To avoid confusion, we write ρ for these states and ξ for paths in TS^{π}. Moreover, there is a transition $\rho \xrightarrow{\alpha} \rho'$ iff (i) $\mathcal{M}^{\pi}(\rho, \rho') > 0$ and (ii) $\alpha = (\mathbf{u}, a, \mathbf{v})$ is the unique $\mathcal{M}$-event that satisfies $\text{last}(\rho)[a] = \mathbf{u}$ and $\mathbf{s}'[a] = \mathbf{v}$ where $\rho' = \rho a\mathbf{s}'$. In effect, TS^{π} is obtained from $\mathcal{M}^{\pi}$ by replacing the probability "labels" of transitions by the $\mathcal{M}$-event corresponding to that transition. In particular, note that for a state ρ of TS^{π}, $a \in Act(\text{last}(\rho))$ iff there exists an $\mathcal{M}$-event $\alpha = (\mathbf{u}, a, \mathbf{v})$ such that $\text{last}(\rho)[a] = \mathbf{u}$. Based on this, we can directly transfer the definition of complete paths to TS^{π}. We define the set of successor states in the obvious way, i.e. $succ(\rho) = \{\rho' \mid \exists\alpha.\ \rho \xrightarrow{\alpha} \rho'\}$. Observe that if $succ(\rho) = \{\rho_1, \rho_2, \ldots, \rho_k\}$ and $\rho \xrightarrow{\alpha_i} \rho_i$ for $1 \le i \le k$ then there exists an a such that $Act(\alpha_i) = a$ for every $i \in \{1, 2, \ldots, k\}$ and $\sum_{1\le i\le k} Pr(\alpha_i) = 1$. For the rest of this subsection, we work with this transition system.

We now turn to representing $\text{PathReach}(\mathcal{M}^{\pi}, e)$ as a finite prefix of TS^{π}. First we introduce some useful terminology. We set $c_0 = {\downarrow}e$. Next, suppose $\xi = \rho_0\alpha_1\rho_1\cdots\alpha_n\rho_n$ is a path in TS^{π}. Then, $EV(\xi) = \{ev(\alpha_1\alpha_2\cdots\alpha_i) \mid 1 \le i \le n\}$ denotes the set of events encountered along the path ξ. Naturally, $EV(\xi) = \emptyset$ if $\xi = \hat{\mathbf{s}}$. We write $G_e = (V, \Longrightarrow)$ to denote the finite prefix of TS^{π}_{comp} we are after. We construct G_e inductively by starting with $\hat{\mathbf{s}} \in V$ and mark it as unprocessed. We define ε to be a path in V and $\hat{\mathbf{s}} = \text{last}(\varepsilon)$. We note that $EV(\hat{\mathbf{s}}) = \emptyset \subset c_0$ (as usual, $\subset$ denotes a strict subset).

Suppose $\xi = \rho_0\alpha_1\rho_1\cdots\rho_{n-1}\alpha_n\rho_n$ is a path in V with ρ_n marked as unprocessed and all the other nodes preceding it in ξ marked as processed. Furthermore, assume that $EV(\xi) \subset c_0$. Let $succ(\rho_n) = \{\rho'_1, \rho'_2, \ldots, \rho'_k\}$ and $\beta_1, \beta_2, \ldots, \beta_k$ such that $\rho \xrightarrow{\beta_i} \rho'_i$ for $1 \le i \le k$. We now extend G_e by adding the nodes $\rho'_1, \rho'_2, \ldots, \rho'_k$ to V and the transitions (ρ, β_i, ρ'_i) for $1 \le i \le k$ to $\Longrightarrow$. We mark ρ_n as processed. To define the status of the new nodes that have been added, we consider two cases after setting $e_i = ev(\alpha_1\alpha_2\cdots\alpha_n\beta_i)$ for $1 \le i \le k$.

Case 1. Suppose there exists i with $e_i \le e$. Then $e_i \in c_0 \setminus EV(\xi)$ and hence $EV(\xi\beta_i\rho'_i) = EV(\xi) \cup \{e_i\}$. If $EV(\xi\beta_i\rho'_i) = c_0$ we mark ρ'_i as a *live leaf node* and do not process it any further. This is so since $ev(\xi\beta_i\rho'_i) = e$ and e has been hence reached. We also note that $\alpha_1\alpha_2\cdots\alpha_n\beta_i \in \text{PathReach}(\mathcal{M}^{\pi}, e)$. On the other hand, if $EV(\xi\beta_i\rho'_i) \subset c_0$, we mark ρ'_i as unprocessed.

In addition we mark, for each $l \in \{1, 2, \ldots, k\} \setminus \{i\}$, the node ρ'_l to be a *dead leaf node* and do not process it any further. To justify this, let $e_l = ev(\xi\beta_l\rho'_l)$ for $l \in \{1, 2, \ldots, k\} \setminus \{i\}$. Then clearly $e_i \leq e$ and hence by the last part of Lem. 16, we must have $e_i \# e_l$ for every $l \in \{1, 2, \ldots, k\} \setminus \{e_i\}$. But then $e_l \# e_i \leq e$ implies $e_l \# e$ since conflict is inherited via the causality relation in an event structure. Hence e_l and e can not together belong to any configuration and we can never "reach" e by exploring ρ'_l any further.

Case 2. Suppose $e_i \not\leq e$ for each i. Then, by the third part of Lem. 16, we must have $e_i \mathrel{co} e$. This implies that $EV(\xi\beta_i\rho'_i) = EV(\xi)$ for each i and we mark each node ρ'_i as unprocessed. The idea is that the chosen action a at ρ_n does not contribute to uncovering any of the events in c_0 and hence all the successors of this node must be further explored.

Starting from the root node we repeatedly apply the above rules until there are no unprocessed nodes left. It remains to be shown that G_e is a *finite* prefix of TS^π and consequently the construction procedure for G_e always terminates. To this end, we require some terminology. Let $\xi^\omega = \rho_0\alpha_1\rho_1\alpha_2\cdots$ be a complete path in TS^π. For $n \geq 0$, let $\xi^n = \rho_0\alpha_1\rho_1\cdots\rho_n$ denote the finite prefix of ξ^ω of length n. We say that $\rho_n \xrightarrow{\alpha_{n+1}} \rho_{n+1}$ is a *useful* transition if there exists $e' \in c_0 \setminus EV(\xi^n)$ such that $\pi(\rho_n) = act(e')$. Otherwise it is a *useless* transition. Moreover, we set $EV_e(\xi^n) = EV(\xi^n) \cap c_0$. Finally, we say that $\xi^n = \rho_0\alpha_1\rho_1\cdots\rho_n$ is a *live* path if (i) ρ_i is not a dead leaf node for $1 \leq i \leq n$ and (ii) $EV_e(\xi^n) \subset c_0$.

▶ **Lemma 18.** *Suppose $\xi^n = \rho_0\alpha_1\rho_1\cdots\rho_n$ is a live path.*

1. *If $e' \in min(c_0) \setminus EV(\xi^n)$ then $act(e') \in Act(\rho_n)$*
2. *$\rho_i \in V$ for $0 \leq i \leq n+1$ and $\rho_j \stackrel{\alpha_{j+1}}{\Longrightarrow} \rho_{j+1}$ for $0 \leq j < n+1$.*
3. *If $\rho_n \xrightarrow{\alpha_{n+1}} \rho_{n+1}$ is a useful transition, then ρ_{n+1} is a dead leaf node or $|EV_e(\xi^{n+1})| = |EV_e(\xi^n)| + 1$. Further, ρ_{n+1} is a live leaf node if $EV_e(\xi^{n+1}) = c_0$*
4. *If $\rho_n \xrightarrow{\alpha_{n+1}} \rho_{n+1}$ is a useless transition then $EV_e(\xi^{n+1}) = EV_e(\xi^n)$ and ξ^{n+1} is a live path.*

Proof. For the first part, let $e' \in \min(c_0 \setminus EV(\xi^n))$. If $e'' < e'$ then $e'' \in EV(\xi^n)$. Otherwise $e'' \in c_0 \setminus EV(\xi^n)$ which contradicts $e' \in \min(c_0 \setminus EV(\xi^n))$. Thus $c' = EV(\xi^n) \cup \{e'\}$ is a configuration and $EV_e(\xi^n) \xrightarrow{e'}_{ES} c'$. From the first part of Prop. 15 we get $act(e') \in Act(\rho_n)$. The rest follows from the construction rules for G_e and their explanations. ◀

▶ **Lemma 19.** *The following assertions hold.*

1. *Let $\xi^\omega \in Paths^\pi_{comp}$ with $\xi^n = \rho_0\alpha_1\rho_1\cdots\rho_n$. Then there exists $k > 0$ such that ρ_k is a live or dead leaf node.*
2. *G_e is a finite tree.*

Proof. From the third part of Lem. 18, it follows that there can be at most $|c_0|$ useful transitions along ξ^ω before a dead or live leaf node is encountered. We now claim that there can be only a finite number of *consecutive* useless moves along ξ^ω. This follows from the first and fourth parts of Lem. 18 and the definition of a complete path. Hence ξ^ω will eventually hit a dead or live node. The second part of the lemma now follows from the first part and König's lemma since TS^π is finitely branching. ◀

Since G_e is a finite tree it is immediate that its construction procedure always terminates. It is also easy to see that the set of live branches, i.e. paths from the root node to the live leaf nodes in G_e, correspond to $\mathrm{PathReach}(\mathcal{M}^\pi, e)$.

For the event e of Fig. 3, our construction produces the tree shown in the left of Fig. 4. The boxes denote dead leaf nodes and the circle is the lone live leaf node. On the other hand, for the event e', the resulting tree can be arbitrarily large. After the γ event, the strategy

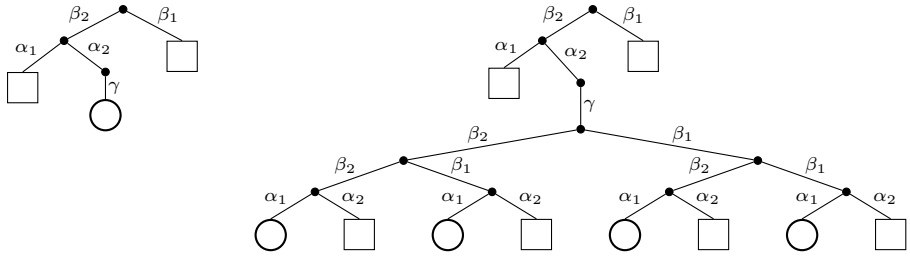


Figure 4 The trees for the events e (left) and e' (right).

can choose to execute the action b a large number of times before executing the action a. For the case where the strategy chooses to do b twice whenever possible before choosing to do an a or c, the resulting tree is shown on the right of Fig. 4, using the same notation.

We can now establish the main result of this subsection.

▶ **Lemma 20.** *Suppose* $e \in E_{\diamond T}$. *Then* $\mu(e) = \mathsf{Pr}^{\pi}_{\mathcal{M}}[\mathrm{PathReach}(\mathcal{M}^{\pi}, e)]$.

Proof. In G_e, each edge in the tree is an $\mathcal{M}$-event α, accompanied by the probability value $Pr(\alpha)$. Hence the probability of a path is fixed to be the product of the probabilities of the labels of the edges encountered on the path. Let V' be the set of nodes in the finite tree consisting of nodes that are not dead leaf nodes. In what follows, ρ ranges over V'. Clearly, the root node ρ_0 is in V'. We now define the probability associated with a node ρ, denoted Pr_ρ, to be the sum of the probabilities of the paths leading from ρ to live leaf nodes. By convention, the probability associated with a live leaf node is 1. We claim that $Pr_{\rho_0} = \mu(e)$.

To prove the claim, we first associate the partial order $(c_\rho, \leq)$ with each node ρ where $c_\rho = c_0 \setminus EV(\xi_\rho)$ with ξ_ρ being the unique path from ρ_0 to ρ in G_e. Next, for each $\rho \in V'$, we let ht_ρ be the length of the longest path from ρ to a live leaf node. We now wish to show by induction on ht_ρ that $Pr_\rho = \prod_{e' \in c_\rho} Pr(\lambda(e'))$ if $c_\rho \neq \emptyset$ and $Pr_\rho = 1$ otherwise. If we do so, then $Pr_{\rho_0} = \mu(e)$ will follow at once. To start with, let ρ be a live leaf node. Then $ht_\rho = 0$ and $Pr_\rho = 1$ by convention.

Next, suppose $c_\rho \neq \emptyset$ and $\pi(\xi_\rho) = a$. We consider two cases. First assume there exists $e' \in \min(c_\rho)$ such that $a = act(e')$. Then ρ has a unique child node ρ' with $(c', \leq)$ as the associated partial order satisfying $c' = c \setminus \{e'\}$. All other successor nodes of ρ will be dead leaf nodes. Now, every path from ρ to a live leaf node consists of the edge (ρ, α, ρ') followed by a path from ρ' to a live leaf node. This implies $Pr_\rho = Pr(\lambda(e')) \cdot Pr_{\rho'}$. By the induction hypothesis, $Pr_{\rho'} = \prod_{e'' \in c'} Pr(\lambda(e''))$. However $c_{\rho'} = c_\rho \setminus \{e'\}$ implies $Pr_\rho = Pr(\lambda(e')) \cdot \prod_{e'' \in c_{\rho'}} Pr(\lambda(e'')) = \prod_{e' \in c_\rho} Pr(\lambda(e'))$ as required.

Next, assume there does not exist $e' \in \min(c)$ such that $a = act(e')$. Let the set of successors of ρ be $\{\rho'_1, \rho'_2, \ldots, \rho'_k\}$ and $\{\beta_1, \beta_2, \ldots, \beta_k\}$ such that $\rho \xrightarrow{\beta_i} \rho'_i$ for $1 \leq i \leq k$. Then every path from ρ to a live leaf node is an edge (ρ, β_i, ρ'_i) followed by a path from ρ'_i to that live leaf node for some i. This implies that $Pr_\rho = \sum_{1 \leq i \leq k} Pr(\beta_i) \cdot Pr_{\rho'_i}$. But then $(c_\rho, \leq)$ is the partial order associated with each ρ_i by the construction of G_e. Hence, by induction hypothesis, $Pr_{\rho_i} = \prod_{e' \in c} Pr(\lambda(e))$. Let $t = \prod_{e' \in c} Pr(\lambda(e'))$. Then $Pr_\rho = \sum_{1 \leq i \leq k} Pr(\beta_i) \cdot t$. But then $\sum_{1 \leq i \leq k} Pr(\beta_i) = 1$. Hence $Pr_\rho = t$ as required. ◀

5.2 All Complete Strategies Achieve the Same Value

We now lift Lem. 20 to $E_{\diamond T}$, i.e. show that $\sum_{e \in E_{\diamond T}} \mu(e) = \bigcup_{e \in E_{\diamond T}} \mathsf{Pr}^{\pi}_{\mathcal{M}}[\mathrm{PathReach}(\mathcal{M}^{\pi}, e)]$.

First, we observe that TS^π naturally inherits a probability measure from $\mathcal{M}^\pi$. To see this, by the definition of TS^π we are assured that $\rho_0\rho_1\cdots\rho_n$ is a path in $\mathcal{M}^\pi$ iff $\rho_0 \xrightarrow{\alpha_1} \rho_1\cdots\rho_{n-1} \xrightarrow{\alpha_n} \rho_n$ is a path in TS^π where the sequence of $\mathcal{M}$-events $\alpha_1\alpha_2\cdots\alpha_n$ is uniquely determined by the sequence $\rho_1\rho_2\cdots\rho_n$. As a result, the σ-algebra generated by the (cylinder set of) finite paths of TS^π will be in a bijective relation with the usual σ-algebra generated by the finite paths of $\mathcal{M}^\pi$. Consequently, we can transfer the probability measure $\mathsf{Pr}^\pi_{\mathcal{M}}$ to a probability measure over the σ-algebra of TS^π. By abuse of notation, we shall denote this measure too as $\mathsf{Pr}^\pi_{\mathcal{M}}$ in what follows.

Now consider $e \in E_{\diamond T}$ and G_e, the finite tree constructed in the previous subsection. Let $Paths_e$ be the set of branches from the root node to the live leaf nodes in G_e. Further, let $CS(\xi)$ be the cylinder set of the finite path ξ in TS^π. Then from the proof of Lem. 20 it follows that $\mathsf{Pr}^\pi_{\mathcal{M}}[\mathrm{PathReach}(\mathcal{M}^\pi, e)] = \mathsf{Pr}^\pi_{\mathcal{M}}[\bigcup_{\xi\in Paths_e} CS(\xi)]$. Consequently, $\mathsf{Pr}^\pi_{\mathcal{M}}[\diamond\mathbf{T}] = \bigcup_{e\in E_{\diamond T}}\bigcup_{\xi\in Paths_e} CS(\xi)$. Since E is a countable set, this probability value is well-defined. To show that this value is the same for all complete strategies, we establish that $\bigcup_{e\in E_{\diamond T}}\bigcup_{\xi\in Paths_e} CS(\xi) = \sum_{e\in E_{\diamond T}} \mu(e)$. The key to doing this is the next result.

▶ **Lemma 21.** *Let $e_1, e_2 \in E_{\diamond T}$ such that $e_1 \neq e_2$. Then $e_1 \# e_2$.*

Proof. Let $e_1 = [\alpha_1\alpha_2\cdots\alpha_n]$ and $e_2 = [\beta_1\beta_2\cdots\beta_m]$. If $e_1 < e_2$, then there exists $i < n$ such that $ev(\beta_1\beta_2\cdots\beta_i) = e_1$. But this contradicts the requirement that α_n is the *first* $\mathcal{M}$-event in the sequence $\alpha_1\alpha_2\cdots\alpha_n$ with $\mathbf{v}_n(p) \in T$ where $\alpha_j = (\mathbf{u}_j, a_j, \mathbf{v}_j)$ for $1 \leq j \leq n$. Thus $e_1 \not< e_2$ and similarly $e_2 \not< e_1$. Next suppose $e_1 \mathrel{co} e_2$.

Then $c_{12} = {\downarrow}e_1 \cup {\downarrow}e_2$ is a configuration. To see this, let x and y be events such that $x \in c_{12}$ and $y \leq x$. Suppose $x \in {\downarrow}e_1$. Then $y \in {\downarrow}e_1 \subseteq c_{12}$. Similarly, $x \in {\downarrow}e_2$ implies that $y \in c_{12}$. Next, suppose that $x \# y$. Then, it can not be the case that x, y are both in ${\downarrow}e_1$ or ${\downarrow}e_2$ since both ${\downarrow}e_1$ and ${\downarrow}e_2$ are configurations and hence conflict-free. Hence, assume that $x \in {\downarrow}e_1$ and $y \in {\downarrow}e_2$. Then $x \leq e_1$ and $y \leq e_2$, which implies that $e_1 \# e_2$, contradicting $e_1 \mathrel{co} e_2$. Thus c_{12} indeed is a configuration.

This implies that ${\downarrow}e_1 \uparrow {\downarrow}e_2$. Hence by the last part of Prop. 12, there exist $\mathcal{M}$-event sequences $\gamma_1\gamma_2\cdots\gamma_l$, $\alpha'_1\alpha'_2\cdots\alpha'_{n'}$, and $\beta'_1\beta'_2\cdots\beta'_{m'}$ such that (i) $\gamma_1\gamma_2\cdots\gamma_l\alpha'_1\alpha'_2\cdots\alpha'_{n'} \approx \alpha_1\alpha_2\cdots\alpha_n$, (ii) $\gamma_1\gamma_2\cdots\gamma_l\beta'_1\beta'_2\cdots\beta'_{m'} \approx \beta_1\beta_2\cdots\beta_m$, and (iii) $\alpha'_i \mathrel{I} \beta'_j$ for $1 \leq i \leq n'$ and $1 \leq j \leq m'$. Since $[\alpha_1\alpha_2\cdots\alpha_n]$ and $[\beta_1\beta_2\cdots\beta_m]$ are both prime traces we must have $\alpha'_{n'} = \alpha_n$ and $\beta'_{m'} = \beta_m$. This leads to $\alpha_n \mathrel{I} \beta_m$, which is a contradiction since $p \in \mathrm{loc}(act(\alpha_n)) \cap \mathrm{loc}(act(\beta_m))$ and hence $\alpha_n \mathrel{D} \beta_m$. ◀

▶ **Lemma 22.** $\mathsf{Pr}^\pi_{\mathcal{M}}[\diamond \boldsymbol{T}] = \sum_{e\in E_{\diamond T}} \mu(e)$.

Proof. We have $\mathsf{Pr}^\pi_{\mathcal{M}}[\diamond\mathbf{T}] = \mathsf{Pr}^\pi_{\mathcal{M}}[\bigcup_{e\in E_{\diamond T}}\bigcup_{\xi\in Paths_e} CS(\xi)]$ from the remarks preceding Lem. 20, where $Paths_e$ is the set of branches from the root node to live leaf nodes in G_e, the finite tree constructed in the proof of Lem. 20. Let $e_1, e_2 \in E_{\diamond T}$ such that $e_1 \neq e_2$. Then $e_1 \# e_2$ by Lem. 21. Let $\xi_1 \in Paths_{e_1}$ and $\xi_2 \in Paths_{e_2}$. Then from the definition of $Paths_e$ it follows directly that $CS(\xi_1) \cap CS(\xi_2) = \emptyset$. This implies that $\mathsf{Pr}^\pi_{\mathcal{M}}[\diamond\mathbf{T}] = \sum_{e\in E_{\diamond T}} \mathsf{Pr}^\pi_{\mathcal{M}}[\bigcup_{\xi\in Paths_e} CS(\xi)]$. From Lem. 20 we get $\mathsf{Pr}^\pi_{\mathcal{M}}[\diamond\mathbf{T}] = \sum_{e\in E_{\diamond T}} \mu(e)$. ◀

This at once leads to our main result.

▶ **Theorem 23.** *Let π and π' be two deterministic complete strategies for the CMDP $\mathcal{M}$. Then $\mathsf{Pr}^\pi_{\mathcal{M}}[\diamond \boldsymbol{T}] = \mathsf{Pr}^{\pi'}_{\mathcal{M}}[\diamond \boldsymbol{T}]$.*

Combined with Lem. 10, we have that in order to compute the optimal local reachability value, we can confine ourselves to complete strategies and from among them, greedily choose one.

6 Implementation and Experimental Evaluation

We implemented a prototype tool and evaluated it on a few models, as we describe in the following. The tool is written in Java and based on PET [22]. It uses PRISM [18] to parse models. We used the pure-Java library `oj! Algorithms` to solve linear programs. We empirically validated the soundness of our implementation by comparing its output on about 20 models to the results of Storm [8] in its sound, exact mode. The tool, its source code, all used models, and further models can be obtained from [23].

6.1 Algorithm Description

Our tool (i) provides a syntactic over-approximation for checking the CD property, and (ii) computes the maximal reachability probability of a local reachability set, assuming that the input MDP is a CMDP. In the interest of space, we only sketch the computation procedure here. More details and a formal description can be found in App. A. Intuitively, the goal is to construct only the part of the system that is reached by one specific complete strategy, chosen as follows. First, we heuristically fix a priority over the set of actions. Then, we begin exploring the global FMDP by starting in the initial state, picking the available action with the highest priority, and determine all its stochastic successors. We repeat this process for the discovered successors until a fixpoint of states is reached. One must however ensure that this greedy prioritization avoids neglecting an available action forever. To this end, we check in each *bottom maximal end component* whether any available action is never chosen. If so, we pick, for each bottom component, the constantly omitted action with highest priority and explore as above. Eventually, this process will terminate with no bottom MECs having any omitted action. Then, we determine the maximal reachability probability of the target local state on the constructed subsystem. In our implementation we use the standard linear program for reachability (see, e.g., [2, Thm. 10.105]). We note that for the computation, CD is only required for correctness, not for termination.

Our implementation is quite simplistic and can be optimized in multiple ways. In particular, the priority order of actions will have a large influence on the size of the resulting subsystem, and this could be significantly improved by intelligent adaptive techniques and learning-based approaches. However, our current heuristical ordering already provides convincing results. Hence we did not explore this issue further.

6.2 Setup and Results

We consider four types of models, each of which was either constructed from scratch or obtained by adapting an existing model to fit into our framework. Unfortunately, most models of the PRISM benchmark suite [19] are not immediately CD and one needs to examine which ones can be adapted to fit into our framework, which we leave for future work. We provide a brief intuitive description of the models we used. The concrete specification in the PRISM modelling language can be found in [23]. The **sync** model consists of 20 processes running in parallel, each repeatedly tossing a (biased) coin and progressing when head is obtained, and finally synchronizing on a common action with all other processes to reach their final states. We next consider a variant of the classical **dining philosophers**, where philosophers alternate between eating and thinking. In our variant, the thinking process of each philosopher has several (probabilistic) steps with each philosopher initially "musing" and eventually becoming "enlightened" or "bewildered", and we seek the probability of one philosopher achieving

Table 1 Overview of results for our four models. From left to right, we list the model name, its overall size (as reported by Storm), the runtime of Storm with sparse and symbolic engine, respectively, the size of the reduced model constructed by our tool, and the overall runtime of our tool. T/O denotes a runtime of over 5 minutes. We also carried out a comparison to PRISM, however Storm was faster in all cases.

Model	Size	Storm-sparse	Storm-symbolic	Reduced Size	Our Tool
sync	$2.1 \cdot 10^6$	17s	2s	22	2s
philosophers	$8.6 \cdot 10^6$	T/O	T/O	3264	4s
production	$6.6 \cdot 10^7$	T/O	T/O	11669	8s
scheduling	$2.8 \cdot 10^{10}$	T/O	T/O	111	<1s
scheduling (large)	??	T/O	T/O	1021	3s

enlightenment. The **production** model comprises a production network where resources are used to assemble (through several steps) a final product. Resources have a chance of becoming exhausted every time they are mined and we are interested in the probability of producing a given quantity of the final product. Finally, **scheduling** models a central process C which proceeds in ten stages. In stage i, the process needs to synchronize with the process p_i to proceed to stage $i+1$. The sub-processes are independent, but may fail to complete. We are interested in the probability of the central process finishing the final stage. For scalability analysis, we also consider a "large" variant where C has 20 stages and each p_i has 50 sequential steps.

We executed our tool on standard hardware and compared our results with those obtained using the model checker Storm. We considered both the default sparse as well as symbolic engines of Storm and otherwise let Storm run in its default configuration. Notably, we did not require exact or sound results (i.e. Storm could decide to use classical, unsound value iteration), while our tool computed correct, exact results using linear programming (up to floating point precision). We summarize our findings in Table 1. One can see that our (basic, unoptimized) approach significantly outperforms existing approaches on the chosen models. This improvement is due to our method being able to avoid visiting a lot number of "useless" states by not exploring every interleaving. On the "large" variant of **scheduling**, Storm fails to even output a state count, which we estimate to be of the order of 50^{20} ($\approx 10^{34}$).

7 Conclusion

We introduced a class of factored MDPs where through the notion of locations, we cleanly separate the causality, concurrency, and conflict relations between the stochastic events in the system. This leads to an event structure semantics for our FMDPs. We mainly used this representation to provide the basis for a powerful partial order based quantitative analysis technique for CMDPs, a natural subclass of FMDPs.

In the future, we plan to study the class of CMDPs from the standpoint of expressiveness. In particular it will be interesting to separate CMDPs from FMDPs that *inherently* do not have the CD property but are unavoidable in practice. Here we suspect that the property called confusion-freeness will play an important role [30]. We also wish to emphasize that the class of FMDPs we identify and their event structure semantics are of independent interest. In particular, it opens up the possibility of using techniques such as finite prefixes of event structures [11] and stubborn sets [15] to analyze FMDPs. These techniques can be applied for model checking the FMDPs for probabilistic temporal logical specifications. To secure the foundations for doing so, the probability measure for events structures that was alluded to at the end of Sec. 4 will need to be fleshed out.

Our experiments suggest that the presented method has significant potential for practical applicability, especially in light of the fact that the method itself can be improved and extended in multiple ways; for instance, by considering reachability properties for a small number of components or by formulating weaker versions of the CD property.

References

1 Samy Abbes and Albert Benveniste. True-concurrency probabilistic models: Branching cells and distributed probabilities for event structures. *Information and Computation*, 204(2):231–274, 2006. doi:10.1016/j.ic.2005.10.001.

2 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

3 Marco Beccuti, Giuliana Franceschinis, and Serge Haddad. Markov decision petri net and markov decision well-formed net formalisms. In Jetty Kleijn and Alexandre Yakovlev, editors, *Petri Nets and Other Models of Concurrency – ICATPN 2007, 28th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency, ICATPN 2007, Siedlce, Poland, June 25-29, 2007, Proceedings*, volume 4546 of *Lecture Notes in Computer Science*, pages 43–62. Springer, 2007. doi:10.1007/978-3-540-73094-1_6.

4 Craig Boutilier, Richard Dearden, and Moisés Goldszmidt. Exploiting structure in policy construction. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 1104–1113. Morgan Kaufmann, 1995. URL: http://ijcai.org/Proceedings/95-2/Papers/012.pdf.

5 Carlos E. Budde, Christian Dehnert, Ernst Moritz Hahn, Arnd Hartmanns, Sebastian Junges, and Andrea Turrini. JANI: quantitative model and tool interaction. In *Tools and Algorithms for the Construction and Analysis of Systems – 23rd International Conference, TACAS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings, Part II*, volume 10206 of *Lecture Notes in Computer Science*, pages 151–168, 2017. doi:10.1007/978-3-662-54580-5_9.

6 Luca de Alfaro. *Formal verification of probabilistic systems*. PhD thesis, Stanford University, USA, 1997. URL: https://searchworks.stanford.edu/view/3910936.

7 Luca de Alfaro, Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Roberto Segala. Symbolic model checking of probabilistic processes using mtbdds and the kronecker representation. In Susanne Graf and Michael I. Schwartzbach, editors, *Tools and Algorithms for Construction and Analysis of Systems, 6th International Conference, TACAS 2000, Held as Part of the European Joint Conferences on the Theory and Practice of Software, ETAPS 2000, Berlin, Germany, March 25 – April 2, 2000, Proceedings*, volume 1785 of *Lecture Notes in Computer Science*, pages 395–410. Springer, 2000. doi:10.1007/3-540-46419-0_27.

8 Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. A storm is coming: A modern probabilistic model checker. In *Computer Aided Verification – 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part II*, volume 10427 of *Lecture Notes in Computer Science*, pages 592–600. Springer, 2017. doi:10.1007/978-3-319-63390-9_31.

9 Volker Diekert and Grzegorz Rozenberg, editors. *The Book of Traces*. World Scientific, 1995. doi:10.1142/2563.

10 Javier Esparza. Decidability and complexity of Petri net problems – an introduction. In *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996*, volume 1491 of *Lecture Notes in Computer Science*, pages 374–428. Springer, 1996. doi:10.1007/3-540-65306-6_20.

11 Javier Esparza and Keijo Heljanko. *Unfoldings – A Partial-Order Approach to Model Checking*. Monographs in Theoretical Computer Science. An EATCS Series. Springer, 2008. doi:10.1007/978-3-540-77426-6.

12 Robert Givan, Thomas L. Dean, and Matthew Greig. Equivalence notions and model minimization in Markov decision processes. *Artif. Intell.*, 147(1-2):163–223, 2003. doi:10.1016/S0004-3702(02)00376-4.

13 Marcus Größer and Christel Baier. Partial order reduction for Markov decision processes: A survey. In Frank S. de Boer, Marcello M. Bonsangue, Susanne Graf, and Willem P. de Roever, editors, *Formal Methods for Components and Objects, 4th International Symposium, FMCO 2005, Amsterdam, The Netherlands, November 1-4, 2005, Revised Lectures*, volume 4111 of *Lecture Notes in Computer Science*, pages 408–427. Springer, 2005. doi:10.1007/11804192_19.

14 Carlos Guestrin, Daphne Koller, Ronald Parr, and Shobha Venkataraman. Efficient solution algorithms for factored MDPs. *J. Artif. Intell. Res.*, 19:399–468, 2003. doi:10.1613/JAIR.1000.

15 Henri Hansen, Marta Z. Kwiatkowska, and Hongyang Qu. Partial order reduction for model checking Markov decision processes under unconditional fairness. In *Eighth International Conference on Quantitative Evaluation of Systems, QEST 2011, Aachen, Germany, 5-8 September, 2011*, pages 203–212. IEEE Computer Society, 2011. doi:10.1109/QEST.2011.35.

16 Mark Kattenbelt, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. A game-based abstraction-refinement framework for Markov decision processes. *Formal Methods Syst. Des.*, 36(3):246–280, 2010. doi:10.1007/S10703-010-0097-6.

17 Daphne Koller and Nir Friedman. *Probabilistic Graphical Models – Principles and Techniques*. MIT Press, 2009. URL: http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=11886.

18 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *Computer Aided Verification – 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011. doi:10.1007/978-3-642-22110-1_47.

19 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. The PRISM benchmark suite. In *Ninth International Conference on Quantitative Evaluation of Systems, QEST 2012, London, United Kingdom, September 17-20, 2012*, pages 203–204. IEEE Computer Society, 2012. doi:10.1109/QEST.2012.14.

20 Marco Ajmone Marsan, Gianfranco Balbo, Gianni Conte, Susanna Donatelli, and Giuliana Franceschinis. Modelling with generalized stochastic petri nets. *SIGMETRICS Perform. Evaluation Rev.*, 26(2):2, 1998. doi:10.1145/288197.581193.

21 Antoni W. Mazurkiewicz. Introduction to trace theory. In Volker Diekert and Grzegorz Rozenberg, editors, *The Book of Traces*, pages 3–41. World Scientific, 1995. doi:10.1142/9789814261456_0001.

22 Tobias Meggendorfer. PET – A partial exploration tool for probabilistic verification. In *Automated Technology for Verification and Analysis – 20th International Symposium, ATVA 2022, Virtual Event, October 25-28, 2022, Proceedings*, volume 13505 of *Lecture Notes in Computer Science*, pages 320–326. Springer, 2022. doi:10.1007/978-3-031-19992-9_20.

23 Tobias Meggendorfer. Causally deterministic markov decision processes, July 2024. Software (visited on 2024-08-20). doi:10.5281/zenodo.12657579.

24 Mogens Nielsen, Gordon D. Plotkin, and Glynn Winskel. Petri nets, event structures and domains, part I. *Theor. Comput. Sci.*, 13:85–108, 1981. doi:10.1016/0304-3975(81)90112-2.

25 Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994. doi:10.1002/9780470316887.

26 Wolfgang Reisig. *Understanding Petri Nets – Modeling Techniques, Analysis Methods, Case Studies*. Springer, 2013. doi:10.1007/978-3-642-33278-4.

27 Brigitte Rozoy and P. S. Thiagarajan. Event structures and trace monoids. *Theor. Comput. Sci.*, 91(2):285–313, 1991. doi:10.1016/0304-3975(91)90087-I.

28 Ratul Saha, Javier Esparza, Sumit Kumar Jha, Madhavan Mukund, and P. S. Thiagarajan. Distributed Markov chains. In Deepak D'Souza, Akash Lal, and Kim Guldstrand Larsen, editors, *Verification, Model Checking, and Abstract Interpretation*, pages 117–134, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. doi:10.1007/978-3-662-46081-8_7.

29 P. S. Thiagarajan and Shaofa Yang. A theory of distributed Markov chains. *Fundam. Informaticae*, 175(1-4):301–325, 2020. doi:10.3233/FI-2020-1958.

30 Daniele Varacca, Hagen Völzer, and Glynn Winskel. Probabilistic event structures and domains. *Theoretical Computer Science*, 358(2):173–199, 2006. Concurrency Theory (CONCUR 2004). doi:10.1016/j.tcs.2006.01.015.

31 Kazuki Watanabe, Clovis Eberhart, Kazuyuki Asada, and Ichiro Hasuo. Compositional probabilistic model checking with string diagrams of MDPs. In *Computer Aided Verification – 35th International Conference, CAV 2023, Paris, France, July 17-22, 2023, Proceedings, Part III*, volume 13966 of *Lecture Notes in Computer Science*, pages 40–61. Springer, 2023. doi:10.1007/978-3-031-37709-9_3.

32 Glynn Winskel and Mogens Nielsen. *Models for concurrency*, pages 1–148. Oxford University Press, Inc., USA, 1995.

A Algorithm Description

In this section, we provide a more detailed description of our algorithmic approach. We assume that we are given the description of each process in an MDP network. As mentioned above, our tool reads models given in the PRISM language, which introduces additional modes of synchronization. For example, guards and updates can read the value of other processes' states without explicitly synchronizing with them.

Checking Causal Determinacy

To syntactically check whether a given model is CD, we check for every local state of every process and every pair of actions available for that process that the intersection of the action guards is empty. This directly implies that the model is CD. However, this is also an over-approximation, since a potential violation might not be reachable in the actual system. All models except the **philosophers** model directly satisfy this simple syntactic property. For the model, **philosophers** model we verified the CD property by manual inspection.

Constructing a Complete Strategy

As mentioned in the main body, our first goal is to heuristically fix a priority order on the available actions. To this end we first record all "dependencies" between processes, i.e. whenever a process reads from or synchronizes with another process, we add an edge in the module dependency graph. Then, starting from the process for which we have the local reachability query, we explore this dependency graph in a breadth-first fashion and order the processes according to this search. We then derive the action priority as follows: We iterate over the processes in the above order, and consider each action this process is involved in which has not yet been processed (i.e. all actions a for which the current module has the highest priority among all processes in $\mathrm{loc}(a)$). These actions are then sorted according to the process with the lowest priority among all of those involved with the action, i.e. $\mathrm{loc}(a)$. This then gives us the overall priority ordering over all actions appearing in the system.

The second part then is to construct a sub-system of the global FMDP which contains at least one complete strategy. By computing the maximal reachability probability on this sub-system, we obtain the overall maximal reachability probability, as any complete strategy

is optimal under CD. To this end, we start in the global initial state and explore the graph induced by the following rule: (i) In a state $\mathbf{s}$, compute the set of available actions $Act(\mathbf{s})$. (ii) Among those actions, pick the action with the highest priority according to the determined order. (iii) Return the set of successors under this action. We fully explore the system induced by this transition relation using BFS. In other words, we explore the sub-system induced by greedily following actions according to our priority order.

As already mentioned, this alone does not guarantee that we get a complete strategy: For example, it might be the case that the highest priority action a available in some state $\mathbf{s}$ simply self-loops, but another action b (with lower priority) would lead to a new successor $\mathbf{s}'$. To ensure this, we determine the set of bottom maximal end components, i.e. all regions where the strategy we are following is "looping". Let R be a set of states forming such an end component in the explored sub-system and for every state $\mathbf{s}$ let $\mathtt{A}(\mathbf{s})$ the action we chose according to our greedy rule. We then compute $\mathcal{A}(R) = \bigcup_{\mathbf{s}\in R} Act(\mathbf{s}) \setminus \bigcup_{\mathbf{s}\in R} \mathtt{A}(\mathbf{s})$. When $\mathcal{A}(R) = \emptyset$, we are finished with the end component R. If not, we pick for each bottom end component R with $\mathcal{A}(R) \neq \emptyset$ the action with the highest priority from $\mathcal{A}(R)$ according to our priority rule and again apply the exploration rule from above.

Correctness

We argue that the subsystem explored in this way contains a complete strategy, independent of the action priority used, by explicitly constructing one. Let $\mathcal{B}$ the set of states in bottom maximal end components in the explored sub-system. Let π a strategy that (i) reaches $\mathcal{B}$ with probability 1 and (ii) uses each action available in $\mathcal{B}$ infinitely often with probability 1 (e.g., by using round-robin memory). Such a strategy exists due to standard results on the properties of end components [2, Chapter 10], [6]. We claim that this strategy is complete.

Assume for contradiction that it is not, i.e. the set of incomplete paths under this strategy has non-zero measure. Since the set of state-action pairs is finite, there exists at least one pair $(\mathbf{s}, a)$ which is "responsible" for the incompleteness. In other words, under the strategy we reach (after a finite number of steps) a state $\mathbf{s}$ where a is available, but from that point onward we never see a with some non-zero probability. Formally, there exists $(\mathbf{s}, a)$ and index i such that $\mathcal{P} = \{\varrho \mid \mathbf{s} = \varrho_i \wedge \forall j \geq i.\ A(\varrho, j) \neq a\}$ has non-zero measure (where $A(\rho, j)$ denotes the action in path ϱ at step j). Observe that by the CD condition, for the paths in $\mathcal{P}$ the action a is available at all subsequent states after i.

Next, let $\mathrm{Inf}(\varrho) \subseteq \mathbf{S}$ the set of states visited infinitely often by path ϱ. Consider the (finite) partitioning of $\mathcal{P}$ by Inf, i.e. grouping runs that visit the same set of states infinitely often. By additivity of $\mathsf{Pr}^{\pi}_{\mathcal{M}}$, there exists at least one partition S_∞ that has non-zero measure. Thus, by the definition of π, S_∞ is a subset of $\mathcal{B}$: Almost all paths under π end up in $\mathcal{B}$, so there can be no non-zero measure set that does not.

To conclude, recall that a is available on all states of all paths in $\mathcal{P}$, including all paths in S_∞. Let R a maximal bottom end component in the explored subsystem (i.e. $R \subseteq \mathcal{B}$) with a non-empty intersection with S_∞. By the definition of π, almost all paths of $\mathcal{P}$ that end up in R visit all states of R infinitely often. Together, a must be available in all states of R, but is never chosen by the strategy π. However, by construction, we would have explored a, as it is an available action in a bottom end component of the subsystem. Concretely, we have that $\mathcal{A}(R)$ is not empty, hence we would explore further, contradicting that R is a bottom end component. This concludes the proof.

B Proof of Lemma 10

▶ **Lemma 10.** *There exists a deterministic, complete strategy $\pi \in \Pi$ which achieves the optimal value, i.e.* $\mathsf{Pr}^{\pi}_{\mathcal{M}}[\Diamond \mathbf{T}] = \sup_{\pi' \in \Pi} \mathsf{Pr}^{\pi'}_{\mathcal{M}}[\Diamond \mathbf{T}]$.

Proof. We show this by arguing that an optimal, possibly non-complete strategy can be modified into a complete one without losing any reachability probability. To this end, let π a *memoryless* deterministic strategy that achieves the optimal value. Assume this strategy is incomplete. We now show how to extend it to a complete strategy. Consider the bottom strongly connected components $\mathcal{B} = \{B_1, \ldots, B_n\}$ in the induced Markov chain $\mathcal{M}^{\pi}$. With probability 1, these are eventually reached (i.e. $\mathsf{Pr}^{\pi}_{\mathcal{M}}[\Diamond \bigcup B_i] = 1$), and, likewise, once in a BSCC B_i, every state within it is reached with probability 1 [2, Chp. 10]. Consider the following strategy π': Follow π, waiting until one of the BSCCs B_i is reached. Meanwhile, track a set of actions A. At each state s, add all actions $Act(s)$ to A and then remove $\pi(s)$. In other words, A tracks all actions that were available but have not been played since they became available. Then, wait until every state in B_i was seen at least once. Until now, π' has behaved exactly as π and has only stored a bounded amount of information.

At this stage π' switches to a different behaviour. Store the set of actions A which have not been played to A' and clear A. By CD, all actions in A are still available. So, π' chooses the actions in A' one by one, and, in the meantime, keeps updating A as before. Once A' is empty, again A is copied to A', A is cleared and the whole process is repeated. (If A is empty at this stage, π simply picks any action.)

This strategy clearly reaches every state that π reaches with at least the same probability, since π' only deviates from π once all states that π can see have been encountered. In addition, this strategy is complete since every action that is available is played within a finite number of steps with probability 1. ◀