Decentralized Bisimulation for Multiagent Systems

Lei Song
Quantum Computation & Intelligent Systems
University of Technology, Sydney
lei.song@uts.edu.au

Yuan Feng
Quantum Computation & Intelligent Systems
University of Technology, Sydney
yuan.feng@uts.edu.au

Lijun Zhang
State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences
zhanglj@ios.ac.cn

ABSTRACT

The notion of bisimulation has been introduced as a powerful way to abstract from details of systems in the formal verification community. When applying to multiagent systems, classical bisimulations will allow one agent to make decisions based on full histories of others. Thus, as a general concept, classical bisimulations are unrealistically powerful for such systems. In this paper, we define a coarser notion of bisimulation under which an agent can only make realistic decisions based on information available to it. Our bisimulation still implies trace distribution equivalence of the systems, and moreover, it allows a compositional abstraction framework of reasoning about the systems.

Categories and Subject Descriptors

F.1.2 [**Models of Computation**]Parallelism and concurrency, Probabilistic computation

General Terms

Theory, Verification

Keywords

Bisimulation; multiagent systems; decentralized

1. INTRODUCTION

Compositional theories have become a foundation for modeling and analyzing stochastic systems. In particular, the notion of compositional minimization [8, 5, 6] approaches has been introduced as a powerful way to abstract from details of systems in the formal verification community [23, 15].

Probabilistic automata (PA), also known as Markov Decision Processes in the planning community, were proposed by Segala in [29] as a compositional behavioral model, see Fig. 1 for some examples. Later, probabilistic automata were used to describe probabilistic multiagent systems [11].

Bisimulation is an important technique to cope with the infamous state-explosion problem in model checking [4]. Essentially, it tells whether two given systems exhibit the same behavior or should be distinguished. Intuitively, two systems are bisimilar if and only if each observable action of one of them can be simulated by the other by performing the same action, and furthermore, the resultant systems are again bisimilar. Ideally, a suitable notion of bisimulation for probabilistic multiagent systems should satisfy the following two natural requirements:

1. It should have an *appropriate distinguishing power*. If two agents output a sequence of observable actions with different probabilities, i.e., they are not trace distribution equivalent, then one can easily distinguish them by only observation (i.e., without interaction with the system). Therefore, a bisimulation should preserve trace distribution equivalence. On the other hand, for the sake of state space reduction, the bisimulation should be as coarse as possible, as coarser bisimulation means fewer states in the quotient system, which in turn means less effort in later verification and analysis.
2. It should be *compositional*, i.e., it is preserved by natural operators for constructing systems. As a multiagent system is typically built from more than one component agents, compositionality of bisimulation means that once we show two agents are bisimilar, no observer can distinguish them in any context. This is crucial in performing minimization compositionally. Note that trace distribution equivalence is not compositional.

Many variants of bisimulations have been proposed as powerful ways for equating behaviorally equivalent states [29, 16, 19]. However, as illustrated by the following example, which is inspired by [29, 17], these bisimulations are not suitable for probabilistic multiagent systems, thus motivating our definition of a coarser notion of distribution-based bisimulation and compositionality under realistic schedulers.

Example 1. *Consider a game with two players: Alice and Bob. Alice tosses a fair coin, while Bob tries to guess the result of the coin throw. If Bob guesses correctly, he wins; otherwise Alice wins. Suppose Alice can choose from two strategies to play the game: 1) She first informs Bob that the game starts and then tosses the coin, or 2) She first tosses the coin, keeps the result secret, and then asks Bob to guess. Formally, these two strategies are described in Fig. 1 (a) and (b), respectively, as probabilistic automata, where i denotes the announcement that the game begins, while h and t denote the coin tossing results "head" and "tail", respectively.*

Obviously, as the only difference between Alice's two strategies is whether or not the coin tossing is made before she announces the start of the game, which is unobservable

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May, 4–8, 2015, Istanbul, Turkey.*

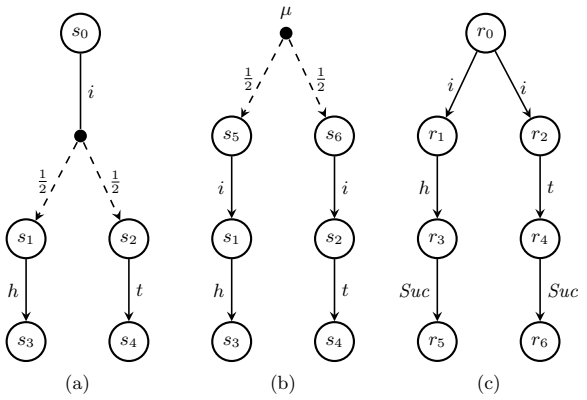

Figure 1: s_0 and μ represent two different strategies of Alice in playing the coin tossing game and r_0 denotes the guesser Bob.

for Bob, these strategies should work equally well for Alice. However, we can show that $s_0 \not\sim \mu$ for any state-based bisimulation $\sim$ defined as in, say [29]. To see this, note that state-based bisimulation is defined over states and then lifted to distributions. Thus, if $s_0 \sim \mu$, then we must have both $s_0 \sim s_5$ and $s_0 \sim s_6$, which is impossible.

The above argument motivates our first design decision to define bisimulation directly on distributions. In our notion of bisimulation, the distribution μ is regarded as a whole when compared with s_0. Thus we have both the transitions $s_0 \xrightarrow{i} \nu$ and $\mu \xrightarrow{i} \nu$ where $\nu = \{s_1 : \frac{1}{2}, s_2 : \frac{1}{2}\}$, making them bisimilar as expected.

To illustrate our second design decision, let us model the behaviors of Bob as in Fig. 1 (c): once he gets to know that Alice starts the game (by synchronizing with Alice through action i), he guesses Alice's coin tossing result by non-deterministically choosing the action h or t. Now consider a play of the game which is obtained by the parallel composition of Alice and Bob. We use a *CSP*-style parallel composition, where synchronization of corresponding actions of Alice and Bob in the set $A = \{i, h, t\}$ is enforced. If Bob wins (i.e., he successfully synchronizes with Alice through either h or t), he performs the action Suc to announce his success.

In view of our criterion (2) of an ideal bisimulation, we would like the two systems $s_0 \parallel_A r_0$ and $\mu \parallel_A r_0$ to be bisimilar. However, if a traditional way of defining compositionality is adopted, they are not bisimilar, even with the weaken notions of distribution-based bisimulation defined in this paper or [16, 19]. To see this, we depict the two systems $s_0 \parallel_A r_0$ and $\mu \parallel_A r_0$ in Fig. 2. For $s_0 \parallel_A r_0$, the probability of observing action Suc is exactly 0.5, no matter how a scheduler resolves the non-deterministic choice (which corresponds to how Bob guesses the outcome of the coin). In contrast, for system $\mu \parallel_A r_0$, an (unrestricted, as in the definition of classical bisimulations) scheduler can choose the lower i action from state $s_5 \parallel_A r_0$ while the upper i from state $s_6 \parallel_A r_0$. Obviously, this scheduler induces an execution for which action Suc is observed with probability 1.

To further explain why the traditional compositionality is a too strong requirement for multiagent systems, let us have a closer look at the scheduler that distinguishes $s_0 \parallel_A r_0$ and $\mu \parallel_A r_0$. The scheduler chooses different i actions for Bob in states $s_5 \parallel_A r_0$ and $s_6 \parallel_A r_0$. In other words, from Bob's view, the resolution of his non-deterministic choice of which transition to perform in state r_0 is based on the state Alice has reached by performing her internal probabilistic choice, namely either state s_5 or s_6. As in the context of multiagent systems, each agent runs autonomously and is partially observable to other agents as well as the environment, this is not a realistic scheduler. Specifically, Bob would need to see the internal state of Alice (actually the result of the coin throw) to make his decision. However, no communication between Alice and Bob has happened at this point in time, by which this information could have been conveyed. Thus, when agents only share the information they gain through explicit communication via observable actions [22], this behavior is prohibited. Note that it was also argued in [22] that in case all agents in a multiagent system share information with each other, the system can be transformed into an equivalent system with only one agent.

In this paper, we tackle this problem by restricting realistic schedulers to those which are decentralized [17], meaning that the scheduler can be decomposed into sub-schedulers of component agents, and with partial information [10], meaning that each sub-scheduler of component agents can only make decisions based on its local information obtained so far. The compositionality is then defined based on realistic schedulers instead of general schedulers considered in previous bisimulations. Both partial information and decentralized schedulers have been coined as principal means to exclude undesired or unrealistically powerful schedulers. We provide a co-inductive definition for distribution-based bisimulation which echoes these considerations on the automaton level, thereby resulting in a very coarse, yet reasonable, notion of equality for multiagent systems.

Summarizing, our contribution in this paper is a novel notion of bisimulation tailored for distributed systems, including multiagent systems, which

1. preserves observable behaviors, i.e., trace distribution equivalence, under partial information schedulers;
2. is compositional with respect to parallel operator under decentralized schedulers;
3. is proved to be the coarsest bisimulation relation satisfying the above two conditions.

Due to these results several things can be deduced directly, for instance (bounded and unbounded) reachability properties are preserved with respect to parallel composition by our bisimulation relations.

Organization of the Paper.

Section 2 recalls some notations used in the paper. In Section 3, a notion of distribution-based bisimulation is proposed and discussed, and its properties under realistic schedulers established in Section 4. Section 5 concludes the paper.

2. PRELIMINARIES

Let S be a finite set of states ranged over by $r, s, \ldots$. A *distribution* is a function $\mu : S \to [0, 1]$ satisfying $\mu(S) = \sum_{s \in S} \mu(s) \leq 1$. If $\mu(S) = 1$, it is called a *full distribution*, otherwise it is a *sub-distribution*. Let $Dist(S)$ denote the set of distributions over S, ranged over by $\mu, \nu, \gamma, \ldots$. Define $Supp(\mu) = \{s \mid \mu(s) > 0\}$ as the support set of μ. If $\mu(s) = 1$, then μ is called a *Dirac* distribution, written as δ_s. Let $|\mu| = \mu(S)$ denote the size of the distribution μ. Given

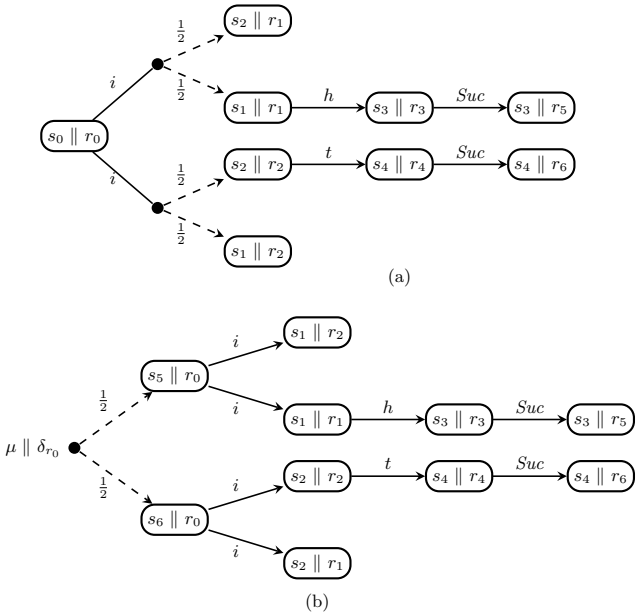

Figure 2: Executions of $s_0 \parallel_A r_0$ and $\mu \parallel_A \delta_{r_0}$ where A is omitted.

a real number $x \geq 0$, $x \cdot \mu$ is the distribution such that $(x \cdot \mu)(s) = x \cdot \mu(s)$ for each $s \in Supp(\mu)$ if $x \cdot |\mu| \leq 1$, while $\mu = \mu_1 + \mu_2$ whenever $\mu(s) = \mu_1(s) + \mu_2(s)$ for each $s \in S$ and $|\mu| \leq 1$. We often write $\{s : \mu(s) \mid s \in Supp(\mu)\}$ alternatively for a distribution μ. For instance, $\{s_1 : 0.4, s_2 : 0.6\}$ denotes a distribution μ such that $\mu(s_1) = 0.4$ and $\mu(s_2) = 0.6$.

2.1 Probabilistic Automata

We consider multiagent systems with probabilistic behaviors. In this paper, each agent as well as the environment is modeled as a probabilistic automaton [29].

Definition 1. *A* PA $\mathcal{P}$ *is a tuple* $(S, Act, \rightarrow, \bar{\mu})$ *where* $\bar{\mu} \in Dist(S)$ *is the initial distribution,* S *is a finite but non-empty set of states,* Act *is a set of actions, and* $\rightarrow \subseteq S \times Act \times Dist(S)$ *is a finite set of probabilistic transitions.*

Let $\alpha, \beta, \gamma, \ldots$ range over the actions in Act. We write $s \xrightarrow{\alpha} \mu$ if $(s, \alpha, \mu) \in \rightarrow$. Let $EA(s) = \{\alpha \in Act \mid \exists \mu. s \xrightarrow{\alpha} \mu\}$, i.e., $EA(s)$ is the set of actions enabled at state s. A *path* is a finite or infinite alternative sequence $\pi = s_0, \alpha_0, s_1, \alpha_1, s_2 \ldots$ of states and actions, such that for each $i \geq 0$ there exists a distribution μ with $s_i \xrightarrow{\alpha_i} \mu$ and $\mu(s_{i+1}) > 0$. We introduce some notations as follows:

- $|\pi|$, the length of π, i.e., the number of states on π, provided π is finite,
- $\pi\downarrow$, the last state of π, provided π is finite,
- $\pi[i] = s_i$ with $i \geq 0$, the $(i+1)$-th state on π if it exists,
- $\pi[0..i] = s_0, \alpha_0, s_1, \alpha_1, \ldots, s_i$, the prefix of π ending at state $\pi[i]$,

Let $Paths^{\omega}(\mathcal{P}) \subseteq S \times (Act \times S)^{\omega}$ and $Paths^{*}(\mathcal{P}) \subseteq S \times (Act \times S)^{*}$ denote the sets containing all infinite and finite paths of $\mathcal{P}$ respectively. Let $Paths(\mathcal{P}) = Paths^{\omega}(\mathcal{P}) \cup Paths^{*}(\mathcal{P})$. In case $\mathcal{P}$ is clear from the context, we simply omit it. We also let $Paths(s)$ be the set containing all paths starting from $s \in S$, similarly for $Paths^{*}(s)$ and $Paths^{\omega}(s)$.

Due to non-deterministic choices in PAs, a probability measure cannot be defined directly. As usual, we shall introduce the definition of *schedulers* to resolve the non-determinism. Intuitively, a scheduler will decide which transition to choose at each step, based on the history execution. Formally,

Definition 2. *A scheduler is a function* $\xi : Paths^{*} \mapsto Dist(Act \times Dist(S))$ *such that* $\xi(\pi)(\alpha, \mu) > 0$ *implies* $\pi\downarrow \xrightarrow{\alpha} \mu$. *A scheduler* ξ *is* deterministic *if it returns only Dirac distributions, that is,* $\xi(\pi)(\alpha, \mu) = 1$ *for some* α *and* μ.

In the sequel we will write $\xi(\pi) = \alpha$ to denote the fact that ξ only chooses transitions with the same label at each step, i.e., for all π, $\sum_{\mu \in Dist(S)} \xi(\pi)(\alpha, \mu) = 1$ for some α.

The *cone* of a finite path π, denoted C_π, is the set of infinite paths having π as their prefix, i.e., $C_\pi = \{\pi' \in Paths^{\omega} \mid \pi \leq \pi'\}$, where $\pi \leq \pi'$ iff π is a prefix of π'. Fixing a starting state s and a scheduler ξ, the measure $Pr_{\xi,s}$ of a cone C_π, where $\pi = s_0, \alpha_0, s_1, \alpha_1, \ldots, s_k$, is defined inductively as: $Pr_{\xi,s}(C_\pi) = 0$ if $s \neq s_0$, $Pr_{\xi,s}(C_\pi) = 1$ if $s = s_0$ and $k = 0$, and otherwise, $Pr_{\xi,s}(C_\pi) = Pr_{\xi,s}(C_{\pi[0..k-1]}) \times$

$$\left(\sum_{(s_{k-1}, \alpha_{k-1}, \mu) \in \rightarrow} \xi(\pi[0..k-1])(\alpha_{k-1}, \mu) \cdot \mu(s_k) \right).$$

Let $\mathcal{B}$ be the smallest algebra that contains all the cones and is closed under complement and countable unions. By standard measure theory [18, 26], this algebra is a *σ-algebra* and all its elements are measurable sets of paths. Moreover, $Pr_{\xi,s}$ can be extended to a unique measure on $\mathcal{B}$.

In this paper, a multiagent system is defined as a set of PAs running in parallel. For this, we recall the parallel operator for PAs [29].

Definition 3. *Let* $\mathcal{P}_i = (S_i, Act, \rightarrow_i, \bar{\mu}_i)$, $i = 1, 2$, *be two* PA*s and* $A \subseteq Act$, *then* $\mathcal{P}_1 \parallel_A \mathcal{P}_2 = (S, Act, \rightarrow, \bar{\mu})$ *such that*

1. $S = \{s_1 \parallel_A s_2 \mid (s_1, s_2) \in S_1 \times S_2\}$,
2. $s_1 \parallel_A s_2 \xrightarrow{\alpha} \mu_1 \parallel_A \mu_2$ *iff either* $\alpha \in A$ *and* $s_i \xrightarrow{\alpha}_i \mu_i$ *for each* $i \in \{1, 2\}$, *or* $\alpha \notin A$ *while* $s_i \xrightarrow{\alpha}_i \mu_i$ *and* $\mu_{3-i} = \delta_{s_{3-i}}$ *for some* $i \in \{1, 2\}$,
3. $\bar{\mu} = \bar{\mu}_1 \parallel_A \bar{\mu}_2$,

where $\mu_1 \parallel_A \mu_2$ *is a distribution over* S *such that* $(\mu_1 \parallel_A \mu_2)(s_1 \parallel_A s_2) = \mu_1(s_1) \cdot \mu_2(s_2)$.

Similar idea has been adopted in [11] to define the parallel composition of multiple probabilistic agents represented as *Markov Chains* (MCs) and the resulting composite system is described by a *Markov Decision Process* (MDP). Our definition of parallel operator generalizes the one in [11] in the sense that PAs subsume MDPs, hence also MCs. Note Definition 3 can be generalized to cases with more than two PAs in parallel, i.e.,

$$\parallel_A \{\mathcal{P}_i\}_{0 \leq i \leq n} \equiv (\parallel_A \{\mathcal{P}_i\}_{1 \leq i \leq n}) \parallel_A \mathcal{P}_0.$$

Below gives a simple example showing how the parallel operator works.

Example 2. *Let* s_0 *and* t_0 *be two states as in Fig. 3. Let* $A = \{\alpha\}$. *Intuitively,* $s_0 \parallel_A t_0$ *is a state obtained by running states* s_0 *and* t_0 *in parallel while imposing synchronization on the action* α. *By Definition 3,* $s_0 \parallel_A t_0$ *will have two transitions:* $s_0 \parallel_A t_0 \xrightarrow{\alpha} \nu_1 \equiv \mu_1 \parallel_A \mu_3$ *and* $s_0 \parallel_A t_0 \xrightarrow{\beta} \nu_2 \equiv$

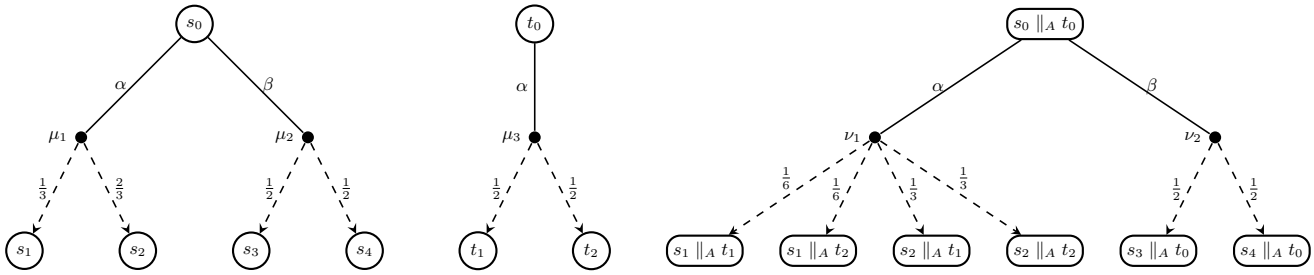

Figure 3: The parallel composition of s_0 and t_0 synchronizing on $A = \{\alpha\}$.

$\mu_2 \parallel_A \delta_{t_0}$, where the transition with label α corresponds to the synchronization of s_0 and t_0 on α, while the transition labelled by β is the transition executed by s_0 spontaneously, as t_0 remains after the transition.

Definition 4. *A* probabilistic multiagent system *(*PMA*) $\mathcal{M}$ is a set of* PA*s, called* local agents *and one of which may represent the environment, composed in parallel as $\parallel_A \{\mathcal{P}_i\}_{0\leq i\leq n}$. States of $\mathcal{M}$ are referred as* global states*, while states of local agents are called* local states*.*

Local agents execute actions not in A independently without synchronizing with others, while for actions in A, they need to synchronize with each other, which can be seen as a way to exchange information among all agents. It is easy to see from Definition 3 that each PMA is also a PA.

Note that we can also define a PMA hierarchically to model the case where some communication is only made between a subset of local agents. For example, in the PMA $(\mathcal{P}_1 \parallel_A \mathcal{P}_2) \parallel_B \mathcal{P}_3$, $\mathcal{P}_1$ and $\mathcal{P}_2$ may exchange information through actions in $A \backslash B$, while $\mathcal{P}_1$ or $\mathcal{P}_2$ may communicate with $\mathcal{P}_3$ through $B \backslash A$. Furthermore, information exchanged through $A \cap B$ becomes common knowledge shared among all the three agents.

3. BISIMULATIONS FOR PMAS

In this section, we first recall the definition of distribution-based bisimulation on probabilistic automata in [14, 12], which we have argued that may distinguish too many states in certain scenarios. We then propose a novel definition of distribution-based bisimulation for probabilistic multiagent systems.

Let $\mu \xrightarrow{\alpha} \mu'$ iff there exists a transition $s \xrightarrow{\alpha} \mu_s$ for each $s \in Supp(\mu)$ such that $\mu' = \sum_{s\in Supp(\mu)} \mu(s) \cdot \mu_s$. Given a transition relation $\rightsquigarrow \subseteq S \times Act \times Dist(S)$, we let $s \overset{\alpha}{\rightsquigarrow}_c \mu$ iff there exists a finite number of real numbers $w_i > 0$ and transitions $s \overset{\alpha}{\rightsquigarrow} \mu_i$ such that $\sum_i w_i = 1$, and $\sum_i w_i \cdot \mu_i = \mu$. We call $\rightsquigarrow_c$ *combined transitions* (of $\rightsquigarrow$).

Definition 5. *The distribution-based bisimulation defined in [14, 12] is a symmetric relation $\mathcal{R} \subseteq Dist(S) \times Dist(S)$ such that $\mu \mathcal{R} \nu$ implies:*

1. *whenever $\mu \xrightarrow{\alpha}_c \mu'$, there exists a $\nu \xrightarrow{\alpha}_c \nu'$ such that $\mu' \mathcal{R} \nu'$;*
2. *whenever $\mu \equiv \sum_{0\leq i\leq n} p_i \cdot \mu_i$ there exists a decomposition $\nu \equiv \sum_{0\leq i\leq n} p_i \cdot \nu_i$ such that $\mu_i \mathcal{R} \nu_i$ for each i, where $\sum_{0\leq i\leq n} p_i = 1$ with $p_i > 0$ for each i.*

We say that μ and ν are bisimilar, written as $\mu \sim \nu$, iff there exists such a bisimulation relation $\mathcal{R}$ with $\mu \mathcal{R} \nu$. Moreover $s \sim r$ iff $\delta_s \sim \delta_r$.

Clause 1 is standard. Clause 2 says that no matter how we split μ, there always exists a splitting of ν to simulate the splitting of μ. Clause 2 in Definition 5 is, in fact, the cause why this relation is unrealistically strong for scenarios as those discussed in Example 1. The reason is that in order to establish a bisimulation, *every* splitting of μ into distributions must be matched by ν. This also includes splittings into Dirac distributions. Intuitively, this means that still the individual behavior of each single state in $Supp(\mu)$ must be matched. In our scenarios, however, we want to focus on the behavior of distributions over states and not their individual supporting states. We will correct this in our definition of distribution-based bisimulation later.

Definition 6. *A distribution μ is* transition consistent*, written as $\overrightarrow{\mu}$, if for any $s_1, s_2 \in Supp(\mu)$, $EA(s_1) = EA(s_2)$.*

Intuitively, a distribution is transition consistent if and only if all states in its support have the same set of enabled actions. When a distribution is transition consistent, then $\mu \xrightarrow{\alpha}$ whenever there is a state $s \in Supp(\mu)$ with $s \xrightarrow{\alpha}$. This also means that when a distribution is *not* transition consistent, there must be a transition that a certain state in the support can perform but the distribution cannot. We then say that this state is *blocked* from taking this transition.

We are now ready to define our notion of distribution-based bisimulation in this paper, which serves as the basis for the decentralized bisimulation we are going to introduce in Section 4. To simplify notation, we simply call our bisimulation *distribution-based bisimulation*, which should not be confused with Definition 5.

Definition 7. *A symmetric relation $\mathcal{R} \subseteq Dist(S) \times Dist(S)$ is a* distribution-based bisimulation *iff $\mu \mathcal{R} \nu$ implies:*

1. *whenever $\mu \xrightarrow{\alpha}_c \mu'$, there exists a $\nu \xrightarrow{\alpha}_c \nu'$ such that $\mu' \mathcal{R} \nu'$;*
2. *if not $\overrightarrow{\mu}$, then there exists $\mu \equiv \sum_{0\leq i\leq n} p_i \cdot \mu_i$ and $\nu \equiv \sum_{0\leq i\leq n} p_i \cdot \nu_i$ such that $\overrightarrow{\mu_i}$ and $\mu_i \mathcal{R} \nu_i$ for each $0 \leq i \leq n$ where $\sum_{0\leq i\leq n} p_i = 1$ with $p_i > 0$ for each i.*

We say that μ and ν are distribution-based bisimilar, written as $\mu \sim^d \nu$, iff there exists a distribution-based bisimulation $\mathcal{R}$ such that $\mu \mathcal{R} \nu$. Moreover $s \sim^d r$ iff $\delta_s \sim^d \delta_r$.

The main difference between $\sim$ and $\sim^d$ is that in Definition 5 we require for *any* split $\mu \equiv \sum_{0\leq i\leq n} p_i \cdot \mu_i$ of μ, there exists $\nu \equiv \sum_{0\leq i\leq n} p_i \cdot \nu_i$ with $\mu_i \mathcal{R} \nu_i$ for each i, while in Definition 7, we require to split μ *only if* it is not transition consistent. We further require that in the splitting, each distribution μ_i is transition consistent. We do not require this for ν_i. It can be shown, however, that $\overrightarrow{\mu_i}$ and $\mu_i \mathcal{R} \nu_i$ implies $\overrightarrow{\nu_i}$. These conditions ensure that no states in the

support of μ are blocked from executing certain transitions. Clearly, if μ is already transition consistent, we do not need to split μ further, since no transition of states in $Supp(\mu)$ is blocked, thus the distribution transitions in clause 1 suffice to capture every behavior.

Remark 1. *Essentially, in Definition 7 we keep all states with the same set of enabled actions together. This is similar to the idea in [10], where all states with the same enabled actions are non-distinguishable from the outside. Once a distribution becomes transition consistent, we will not try to split it anymore – but rather match the lifted transitions according to the first clause.*

The notion of consistent distributions defined in Definition 6 actually induces an equivalence relation $R = \{(s,t) \in S \times S \mid EA(s) = EA(t)\}$ *over the state space, which is in turn used in Definition 7 to guide the splitting of a distribution. Therefore, our distribution-based bisimulation can be seen as defined upon R. The main purpose of R is to characterize indistinguishable states in a multiagent system, which is also referred to as an* epistemic accessibility relation *in [32] or an* information function *in [27]. The relation R is the coarsest one for schedulers to be well defined in such systems, since otherwise an action scheduled at one state may not be enabled at all at another equivalent state. This is not acceptable, particularly in the setting of planning [27] and model checking with incomplete information [20], schedulers transferring [7] and so on. However, nothing hampers the usage of finer relations than R. By doing so, all results in this paper are preserved, as long as we change the definition of partial information scheduler accordingly, which we will introduce soon. Furthermore, like in [32, 27], we can also define different levels of accessibility (or distinguishability) for different agents such that two global states* $\|_A \{s_i\}_{0\leq i\leq n}$ *and* $\|_A \{t_i\}_{0\leq i\leq n}$ *are not distinguishable to the i-th agent iff* s_i *and* t_i *are not distinguishable. To simplify the presentation, we assume the same accessibility relation R to all agents in the sequel.*

Example 3. *Back to Example 1, we are going to show that in Fig. 1,* $\delta_{s_0} \sim^d \mu$. *Let* $\mathcal{R} = \{(\delta_{s_0}, \mu), (\mu, \delta_{s_0})\} \cup \mathrm{ID}$ *where* ID *is the identity relation. It is easy to show that* $\mathcal{R}$ *is a distribution-based bisimulation. The key point is that both* δ_{s_0} *and* $\{s_5 : 0.5, s_6 : 0.5\}$ *are transition consistent, thus we do not need to split them any further. Conversely, we can show that* $\mathcal{R}$ *is not a bisimulation in the settings of [14, 12]. According to clause* 1 *of Definition 5, we require that for any split of* $\{s_5 : 0.5, s_6 : 0.5\}$, *there must exist a matching split of* δ_{s_0}, *which cannot be established. For instance the split* $\{s_5 : 0.5, s_6 : 0.5\} = 0.5 \cdot \delta_{s_5} + 0.5 \cdot \delta_{s_6}$ *cannot be matched by any split of* δ_{s_0}. □

The following example shows that the transition consistency condition of Definition 7 is necessary to avoid equating states which should be distinguished.

Example 4. *Suppose there are two states* s_0 *and* r_0 *such that* $s_0 \xrightarrow{\alpha} s_1$ *and* $r_0 \xrightarrow{\alpha} \{r_1 : 0.5, r_2 : 0.5\}$ *where all of* s_1, r_1, *and* r_2 *have a transition to themselves with label* α. *In addition,* $r_1 \xrightarrow{\beta} r_1$ *where* $\alpha \neq \beta$.

If clause 2 in Definition 7 was dropped, then we would have $s_0 \sim^d r_0$. *To see this, let* $\mathcal{R}$ *be the symmetric closure of* $\{(\delta_{s_0}, \delta_{r_0}), (\delta_{s_1}, \{r_1 : 0.5, r_2 : 0.5\})\}$. *Note that the distribution* $\{r_1 : 0.5, r_2 : 0.5\}$ *can only perform an* α *transition and then return to itself (the* β *transition of* r_1 *would be blocked as it is disabled in* r_2*). Thus* $\mathcal{R}$ *is a distribution-based bisimulation, and* $s_0 \sim^d r_0$. *However,* s_0 *and* r_0 *should be distinguished, because* r_0 *can reach state* r_1 *which is able to perform a* β *transition, while* s_0 *cannot.*

According to our Definition 7, however, as the distribution $\{r_1 : 0.5, r_2 : 0.5\}$ *is not transition consistent, we should split it further to a convex combination of* δ_{r_1} *and* δ_{r_2}. *Then* $s_0 \not\sim^d r_0$ *can be easily seen from the fact that* δ_{r_1} *cannot be matched by any distribution from* s_1.

The following theorem shows that both $\sim$ and $\sim^d$ are equivalence relations. Moreover $\sim^d$ is strictly coarser than $\sim$, which is straightforward from Definitions 5 and 7.

Theorem 1. *1.* $\sim$ *and* $\sim^d$ *are equivalence relations; 2.* $\sim \subseteq \sim^d$.

4. OBSERVABLE BEHAVIOR AND COMPOSITION

We consider important properties of distribution-based bisimulation in this section, namely preservation of trace distributions equivalence, and compositionality. While these properties do *not* hold if considering all schedulers, we establish them for the subclass of partial information decentralized schedulers. Partial information schedulers $\mathcal{S}_P$ have been coined by De Alfaro [10], and decentralized schedulers $\mathcal{S}_D$ stem from D'Argenio and Giro [17]. Both have been proposed to rule out unrealistic scheduling decisions such as the ones discussed in Fig. 2. We echo these arguments to back our claim that distribution-based bisimulation is a valuable relation in the context of realistic scheduling, especially in multiagent systems. To get started, we review some desirable properties we are going to discuss.

We first introduce the notion of trace distribution equivalence [28] adapted to our setting. Let $\varsigma \in Act^*$ denote a finite trace of a PA $\mathcal{P}$ consisting of an ordered sequence of actions. Moreover, the cylinder C_ς induced by ς is defined by: $C_\varsigma = \cup\{C_\pi \mid \pi \in Paths^* \wedge trace(\pi) = \varsigma\}$ where $trace(\pi) = \epsilon$ denotes an empty trace if $|\pi| \leq 1$, and $trace(\pi) = trace(\pi') \circ \alpha$, where $\pi = \pi' \circ (\alpha, s')$. The measurability of C_ς is straightforward from its definition since it is a countable union of cylinders. Below we define a family of equivalences, parametrized by certain classes of schedulers.

Definition 8. *Let* μ_1 *and* μ_2 *be two distributions of a* PA, *and* $\mathcal{S}$ *a set of schedulers. Then,* $\mu_1 \equiv_{\mathcal{S}} \mu_2$ *iff for each scheduler* $\xi_1 \in \mathcal{S}$ *there exists a scheduler* $\xi_2 \in \mathcal{S}$, *such that*

$$\sum_{s \in Supp(\mu_1)} \mu_1(s) \cdot Pr_s^{\xi_1}(C_\varsigma) = \sum_{s \in Supp(\mu_2)} \mu_2(s) \cdot Pr_s^{\xi_2}(C_\varsigma)$$

for each finite trace ς *and vice versa. If* $\mathcal{S}$ *is the set of all schedulers, we simply write* $\equiv$. *We write* $s_1 \equiv_{\mathcal{S}} s_2$ *iff* $\delta_{s_1} \equiv_{\mathcal{S}} \delta_{s_2}$.

Below follow examples (and counterexamples) of trace distribution equivalent states:

Example 5. *Let* s_0 *and* μ *be as in Fig. 1, then we have* $\delta_{s_0} \equiv \mu$, *since the only trace distribution of* δ_{s_0} *and* μ *is* $\{ih : \frac{1}{2}, it : \frac{1}{2}\}$. *In contrast,* δ_{s_0} *and* ν *in Fig. 4 are not trace distribution equivalent, since there are two possible trace distributions for* $\delta_{s_0} : \{\beta : 1\}$ *and* $\{\alpha : 1\}$, *but for* ν *there are four trace distributions:* $\{\alpha : 1\}$, $\{\beta : 1\}$, $\{\alpha : \frac{1}{3}, \beta : \frac{2}{3}\}$, *and* $\{\beta : \frac{1}{3}, \alpha : \frac{2}{3}\}$. *The last two distributions are induced*

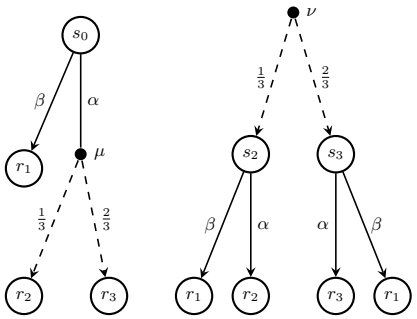

Figure 4: $\delta_{s_0} \sim^d \nu$.
by schedulers of ν which choose transitions with different labels at states s_2 and s_3, which cannot be simulated by any scheduler of s_0[1].

4.1 Realistic Schedulers

We now refine the very liberal Definition 2 where the set of all schedulers was introduced. As discussed, this class is too powerful for multiagent systems, since it includes unrealistic schedules such as the one which distinguishes the two automata in Fig. 2.

In the following we define two prominent sub-classes of schedulers, where only limited information is employed for scheduling. Recall $EA(s)$ denotes the set of actions enabled at s, which can be seen as the interface of s observable by other agents. We generalize the function EA to paths as follows: $EA(\pi) = EA(s)$ if $\pi = s$, and $EA(\pi) = EA(\pi')\alpha EA(s)$, where $\pi = \pi' \circ (\alpha, s)$. Other agents can only observe actions performed by s and its ancestors as well as their interfaces. We are now ready to define the *partial information schedulers* [10] as follows:

Definition 9. *A scheduler ξ is a partial information scheduler of s if for any $\pi_1, \pi_2 \in Paths^*(s)$, $EA(\pi_1) = EA(\pi_2)$ implies $\xi(\pi_1) = \alpha$ and $\xi(\pi_2) = \alpha$ for some α. We say ξ is a partial information scheduler of a* PMA *iff it is a partial information scheduler for all states in the* PMA.

We denote the set of all partial information schedulers by $\mathcal{S}_P$. Intuitively a partial information scheduler can only distinguish states via different enabled actions. It therefore excludes the possibility to schedule differently only because of different state identities. This fits well to a behavior-oriented rather than state-oriented view, as it is typical for multiagent systems. Consequently, for two different paths π_1 and π_2 with $EA(\pi_1) = EA(\pi_2)$, namely π_1 and π_2 are not distinguishable, a partial information scheduler chooses transitions labelled with the same action for both π_1 and π_2.

In order to exclude unrealistic schedulers when composing agents in parallel, another important sub-class of schedulers called *decentralized schedulers* has been introduced [17]. The idea is to assume that an agent running in parallel to other agents needs to make its local scheduling decisions in isolation, and thus can use only information about other agents that has been communicated to them beforehand. For instance the guesser Bob in Fig. 1 cannot base his local scheduling decision on the tossing outcome at the moment when his guess is to be scheduled. This fits perfectly in the setting of multiagent systems, as in such systems, each agent performs actions spontaneously and only interacts with other agents and the environment when necessary.

To formalize this locality idea, we first need to define the projection of a path to the path of its component agents. Let $s = \|_A \{s_i \mid 0 \le i \le n\}$ be a global state which is composed of $n+1$ local states with $n \ge 1$ in parallel such that all the local states synchronize on actions in A. Let π be a path starting from s, then the i-projection of π, denoted by $[\pi]_i$, is defined as follows: $[\pi]_i = [s]_i$ if $\pi = s$; otherwise if $\pi = \pi' \circ (\alpha, s')$,

$$[\pi]_i = \begin{cases} [\pi']_i \circ (\alpha, [s']_i) & \alpha \in A \vee (\alpha \notin A \wedge [\pi'\downarrow]_i \xrightarrow{\alpha} [s']_i) \\ [\pi']_i & \alpha \notin A \wedge (\exists j \neq i. [\pi'\downarrow]_j \xrightarrow{\alpha} [s']_j) \end{cases}$$

where $[s]_i = s_i$ with $0 \le i \le n$. Intuitively, given a path π of a state s, the i-projection of π is the path that only keeps track of the execution of the i-th agent of s during its execution. Also note any scheduler ξ of s can be decomposed into $n+2$ functions: a global scheduler $\xi_g : Paths^* \times \{0, \ldots, n\} \mapsto \{0, 1\}$ and $n+1$ local schedulers $\{\xi_i\}_{0 \le i \le n}$ such that for any π with $\pi\downarrow = \|_A \{s_i \mid 0 \le i \le n\}$, $\xi(\pi)(\alpha, \|_A \{\mu_i\}_{0 \le i \le n}) =$

$$\Pi_{0 \le i \le n}[\xi_g(\pi, i) \cdot \xi_i(\pi)(\alpha, \mu_i) + (1 - \xi_g(\pi, i)) \cdot Eq(\delta_{s_i}, \mu_i)],$$

where $Eq(\delta_{s_i}, \mu_i)$ returns 1 if $\delta_{s_i} = \mu_i$ and 0 otherwise. Intuitively, the global scheduler ξ_g chooses the agents which will participate in the next transition, while ξ_i guides the execution of s_i in case the i-th agent is chosen by ξ_g. In case the i-th agent is not chosen by the global scheduler, it will remain without changing its state. Below defines the decentralized schedulers:

Definition 10. *A scheduler ξ is* decentralized *for $s = \|_A \{s_i \mid 0 \le i \le n\}$ iff its corresponding global scheduler ξ_g and local schedulers $\{\xi_i\}_{0 \le i \le n}$ satisfy: for any $\pi, \pi' \in Paths^*$ and $0 \le i \le n$, $[\pi]_i = [\pi']_i$ implies $\xi_g(\pi, i) = \xi_g(\pi', i)$ and $\xi_i(\pi) = \xi_i(\pi')$. A decentralized scheduler for a* PMA *is a scheduler decentralized for all states in* PMA.

We denote the set of all decentralized schedulers by $\mathcal{S}_D$. In case $n = 1$, decentralized schedulers degenerate to ordinary schedulers defined in Definition 2. According to Definition 10, a scheduler ξ is decentralized, if ξ cannot distinguish different paths starting from s, provided the projections of these paths to each agent coincide. Note that the scheduler inducing the unrealistic execution in Fig. 2 is not decentralized, since the decision of r_0 depends on the execution history of μ, i.e., at state s_5, r_0 will choose the left transition, and it will choose the right transition while at state s_6. By restricting to the set of decentralized schedulers, we can avoid the unrealistic execution of $\mu \|_A \delta_{r_0}$ depicted in Fig. 2.

4.2 Properties of $\sim^d$

In this section we show properties of distribution-based bisimulation under realistic schedulers. We first introduce some notations: A transition from s to ν with label α is induced by a scheduler ξ, written as $s \xrightarrow{\alpha}_\xi \mu$, iff $\mu \equiv \sum_{\mu' \in Dist(S)} \xi(s)(\alpha, \mu') \cdot \mu'$. Accordingly, we write $\mu \xrightarrow{\alpha}_\xi \nu$ to denote that μ can evolve into ν by performing a transition with label α under the guidance of ξ, where $s \xrightarrow{\alpha}_\xi \nu_s$ for each $s \in Supp(\mu)$ and $\nu \equiv \sum_{s \in Supp(\mu)} \mu(s) \cdot \nu_s$. Intuitively, given a distribution μ, for each $s \in Supp(\mu)$ we use s as the history information for ξ to guide the execution — the so called *memoryless* schedulers which suffices to define bisimulations,

[1]For simplicity, we only consider deterministic schedulers here. A similar example can be constructed for general schedulers.

since it is the only priori information we have known so far. Based on the notations introduced above, we can modify Definition 7 with schedulers being considered explicitly.

Let $\mathcal{S}$ be a set of schedulers. We denote $\mu \xrightarrow{\alpha}_{\mathcal{S}} \mu'$ if $\mu \xrightarrow{\alpha}_{\xi} \mu'$ for some $\xi \in \mathcal{S}$.

Definition 11. *A symmetric relation $\mathcal{R} \subseteq Dist(S) \times Dist(S)$ is a distribution-based bisimulation with respect to a given set of schedulers $\mathcal{S}$ iff $\mu \ \mathcal{R} \ \nu$ implies:*

1. *whenever $\mu \xrightarrow{\alpha}_{\mathcal{S}} \mu'$, there exists $\nu \xrightarrow{\alpha}_{\mathcal{S}} \nu'$ such that $\mu' \ \mathcal{R} \ \nu'$;*
2. *if not $\overrightarrow{\mu}$, then there exists $\mu \equiv \sum_{0 \le i \le n} p_i \cdot \mu_i$ and $\nu \equiv \sum_{0 \le i \le n} p_i \cdot \nu_i$ such that $\overrightarrow{\mu_i}$ and $\mu_i \ \mathcal{R} \ \nu_i$ for each $0 \le i \le n$ where $\sum_{0 \le i \le n} p_i = 1$ with $p_i > 0$ for each i.*

We write $\mu \sim^d_{\mathcal{S}} \nu$, iff there exists a distribution-based bisimulation $\mathcal{R}$ with respect to $\mathcal{S}$ such that $\mu \ \mathcal{R} \ \nu$. Moreover $s \sim^d_{\mathcal{S}} r$ iff $\delta_s \sim^d_{\mathcal{S}} \delta_r$.

Definition 11 is almost the same as Definition 7 except that we require every transition to be induced by a scheduler in $\mathcal{S}$. When $\mathcal{S}$ is taken to be the set of all possible schedulers, these two definitions are equivalent. Furthermore, as we shall prove later, partial information schedulers are already enough to make them equivalent.

As mentioned before, distribution-based bisimulation has a flavor similar to partial information schedulers in the sense that, due to the transition consistency requirement, there is no difference between states in the support of a distribution if the same set of actions is enabled. Indeed, distribution-based bisimulation and partial information schedulers are closely related. The following theorem states that partial information schedulers are enough to discriminate distribution-based bisimulation with respect to arbitrary schedulers, and that if restricting to partial information schedulers, distribution-based bisimulation implies trace distribution equivalence.

Theorem 2. *For any distributions μ_1 and μ_2, $\mu_1 \sim^d \mu_2 \iff \mu_1 \sim^d_{\mathcal{S}_P} \mu_2$; $\mu_1 \sim^d_{\mathcal{S}_P} \mu_2 \Longrightarrow \mu_1 \equiv_{\mathcal{S}_P} \mu_2$.*

Theorem 2 does not hold if we consider general schedulers:

Example 6. *Let s_0 and ν be as in Fig. 4. In Example 5 we have shown that $\delta_{s_0} \not\equiv \nu$. It is also not hard to check that $\delta_{s_0} \sim^d \nu$. Therefore, if general schedulers were considered, $\sim^d$ does not always imply $\equiv$.*

However, we can show $\delta_{s_0} \equiv_{\mathcal{S}_P} \nu$. To see this, note that the trace distributions $\{\alpha : \frac{1}{3}, \beta : \frac{2}{3}\}$ and $\{\beta : \frac{1}{3}, \alpha : \frac{2}{3}\}$, which witness $\delta_{s_0} \not\equiv \nu$, can only be produced by a scheduler which chooses different transitions at states s_2 and s_3. As s_2 and s_3 have the same enabled actions, this scheduler is not a partial information one, thus should be excluded when the relation $\equiv_{\mathcal{S}_P}$ is considered.

For parallel composition, we need to restrict to decentralized schedulers to establish compositionality, as indicated by the following theorem:

Theorem 3. *For any distributions μ_1 and μ_2,*

$$\mu_1 \sim^d_{\mathcal{S}_D} \mu_2 \Longrightarrow \mu_1 \parallel_A \mu_3 \sim^d_{\mathcal{S}_D} \mu_2 \parallel_A \mu_3$$

for any μ_3 and A.

Below is an example showing why Theorem 3 does not hold if general schedulers are considered.

Example 7. *Let s_0, μ, and r_0 be as in Fig. 1. We have shown in Example 3 that $\delta_{s_0} \sim^d \mu$. However, if general schedulers are considered, $\delta_{s_0} \parallel_A \delta_{r_0} \not\sim^d \mu \parallel_A \delta_{r_0}$, as the unrealistic execution of $\mu \parallel_A \delta_{r_0}$ depicted in Fig. 2 cannot be simulated by $\mu \parallel_A \delta_{r_0}$, no matter how the transitions of $\delta_{s_0} \parallel_A \delta_{r_0}$ are scheduled.*

In contrast, when restricting to decentralized schedulers, we can show that both $s_0 \parallel_A r_0$ and $\mu \parallel_A \delta_{r_0}$ can reach states $s_3 \parallel_A r_5$ and $s_4 \parallel_A r_6$ with probability 0.5 at most, since the scheduler of $\mu \parallel_A \delta_{r_0}$, which induces the unrealistic execution in Fig. 2 is not decentralized.

When restricting to the set of schedulers in $\mathcal{S} = \mathcal{S}_P \cap \mathcal{S}_D$, $\sim^d_{\mathcal{S}}$ will be called *decentralized bisimulation*. As we have demonstrated, decentralized bisimulation is compositional and implies trace distribution equivalence. Actually, we can show that decentralized bisimulation is the coarsest congruence preserving trace distribution equivalence, which in turn can be seen as the symmetric version of trace distribution precongruence defined in [24].

Theorem 4. *Let $\mathcal{S} = \mathcal{S}_P \cap \mathcal{S}_D$, then $\mu_1 \sim^d_{\mathcal{S}} \mu_2$ iff $\mu_1 \equiv^c_{\mathcal{S}} \mu_2$ for any μ_1 and μ_2, where $\mu_1 \equiv^c_{\mathcal{S}} \mu_2$ iff $\mu_1 \equiv_{\mathcal{S}} \mu_2$ and $\mu_1 \parallel_A \mu_3 \equiv_{\mathcal{S}} \mu_2 \parallel_A \mu_3$ for any μ_3 and A.*

4.3 Decision Algorithm

In this subsection we show that the problem of computing distribution-based bisimulation relations is decidable. For this, we first prove the linearity of $\sim^d$.

Lemma 1. *$\sim^d$ is (convex) linear, i.e., $\mu_1 \sim^d \nu_1$ and $\mu_2 \sim^d \nu_2$ imply that for any $p \in [0, 1]$,*

$$(p \cdot \mu_1 + (1-p) \cdot \mu_2) \sim^d (p \cdot \nu_1 + (1-p) \cdot \nu_2).$$

For systems with finite state space, we impose a fixed order for states in $S = (s_1, \ldots, s_n)$ with $n = |S|$ and identify a distribution μ with a vector $(\mu(s_1), \ldots, \mu(s_n))$. The algorithm presented in [19] can be applied for our bisimulation with some changes. The idea of the algorithm is to construct an $n \times m$ *bisimulation matrix* E such that $\mu \sim^d \nu$ iff $(\mu - \nu)E = 0$, where $0 < m \le n$. Intuitively, E contains all linear constraints which should be satisfied for each variable $\mu(s_i) - \nu(s_i)$ with $1 \le i \le n$ for μ and ν being bisimilar. The following lemma shows that such bisimulation matrix always exists for $\sim^d$.

Lemma 2 ([19])**.** *For every linear bisimulation relation $R \subseteq Dist(S) \times Dist(S)$, there exists a bisimulation matrix E such that $\mu \ R \ \nu$ iff $(\mu - \nu)E = 0$.*

We only sketch the algorithm here, interested readers can refer to [19] for details. Let $\mathcal{P}$ be a PA. For simplicity, we assume that $\mathcal{P}$ is deterministic, namely, for each state s in $\mathcal{P}$ and $\alpha \in Act$, whenever $s \xrightarrow{\alpha} \mu_1$ and $s \xrightarrow{\alpha} \mu_2$, it holds $\mu_1 = \mu_2$. In other words, all transitions of each state are labelled differently. For each $A \subseteq Act$ and $\alpha \in A$, let P^α_A be the matrix such that for all $s, t \in S$,

$$\mathsf{P}^\alpha_A(s,t) = \begin{cases} \mu(t) & EA(s) = A \text{ and } s \xrightarrow{\alpha} \mu \\ 0 & otherwise. \end{cases}$$

Note for deterministic PAs, P^α_A is uniquely determined for any $A \subseteq Act$ and $\alpha \in A$.

To construct a bisimulation matrix E for $\mathcal{P}$, the algorithm starts with a matrix with only one column $\mathbf{1}$, i.e., every

entry is 1. Then for each $A \subseteq Act$ and $\alpha \in A$, compute $E_A^\alpha = \mathcal{P}_A^\alpha E$ and let E be the new matrix by adding columns in E_A^α, but removing columns linearly dependent on the others. This process continues until E is stable, i.e., no independent column can be added. The algorithm terminates, since there are at most n independent columns for n variables. For deterministic PAs, the algorithm will terminate in polynomial time [30, 13].

For PAs which are not deterministic, $\mathcal{P}_A^\alpha$ may not be unique for some $A \subseteq Act$ and $\alpha \in A$, since a state may have more than one α-transitions and will induce uncountable many choices due to combined transitions. Fortunately, it was shown in [19] that it suffices to restrict to finitely many transitions to compute E. However, this will cause an exponential blow-up in the worst case and it is still an open problem whether polynomial algorithms exist for general PAs.

4.4 Related Work and Discussion

In this subsection we review some related work. In the classical setting of multiagent systems without probabilistic behavior, the notion of bisimulation has been introduced and studied intensively. For instance, [1] showed the relation between bisimulation and Alternating-time Temporal Logic [3] (ATL) under different semantics. Similar work has been done for epistemic logic [33]. In [2], bisimulation was adopted as an abstraction technique when performing epistemic model checking on dining cryptographers-based protocols. Apart from being an abstraction technique, bisimulation metrics (relaxed versions of bisimulations allowing some probability perturbation) have been used to transfer policies between Markov decision processes [25, 7]. The problem of planning [27] and model checking with incomplete information [21, 20] have also been investigated.

As state-based bisimulations are too distinguishable in certain scenarios, distribution-based bisimulation relations have attracted much attention recently [13, 14, 16, 19]. The equivalence in [13] was defined upon *reactive probabilistic automata*, i.e., PAs where each action has exactly one corresponding transition at all states. To the best of our knowledge, distribution-based bisimulation was first defined upon general PAs in [14] to cascade internal executions. It was shown that their definition of bisimulation is the coarsest one which is compositional and preserves trace distribution equivalence for general schedulers. However, their bisimulation still distinguishes the automata in Fig. 1 (a) and (b).

Later, two other variants of distribution-based bisimulation were defined in [16, 19], which equate the automata in Fig. 1 (a) and (b). However, they do not preserve observable behaviors, namely, trace distribution equivalence, even under the realistic classes of schedulers we will consider in the sequel. To see this, we show why bisimulations defined in [19] do not always imply trace distribution equivalence. Similar arguments can be applied to bisimulations in [16].

Before giving the formal definition, we first recall some notations. Let $S_A = \{s \in S \mid EA(s) \cap A \neq \emptyset\}$ be the set of states such that at least one of their transitions is labelled by an action in A. Let $\mu(A) = \mu(S_A)$ and $\mu \overset{A}{\hookrightarrow} \mu'$ iff $\mu(A) > 0$ and $\mu' = \frac{1}{\mu(A)} \sum_{s \in Supp(\mu) \cap S_A} \mu(s) \cdot \mu_s$, where for each $s \in Supp(\mu) \cap S_A$, $s \xrightarrow{\alpha} \mu_s$ for some $\alpha \in A$. Note, in $\mu \overset{A}{\hookrightarrow} \mu'$, states in $Supp(\mu)$ may perform transitions with different labels as long as they are in A. Below we recall their definition of bisimulations [19, Def. 2], which will be referred as HKK-bisimulation in the sequel.

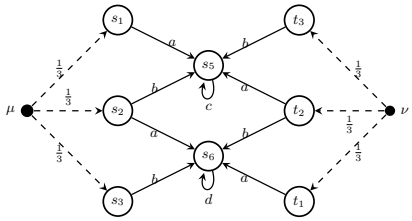

Figure 5: $\mu \sim_{\text{HKK}} \nu$ but $\mu \not\equiv_{SP} \nu$.

Definition 12. *A symmetric relation $\mathcal{R} \subseteq Dist(S) \times Dist(S)$ is an* HKK*-bisimulation iff $\mu \mathcal{R} \nu$ implies:*

1. *$\mu(A) = \nu(A)$ for each $A \subseteq Act$,*
2. *whenever $\mu \overset{A}{\hookrightarrow} \mu'$, there exists a transition $\nu \overset{A}{\hookrightarrow} \nu'$ such that $\mu' \mathcal{R} \nu'$.*

We write $\mu \sim_{\text{HKK}} \nu$, iff there exists an HKK*-bisimulation $\mathcal{R}$ such that $\mu \mathcal{R} \nu$. Moreover $s \sim_{\text{HKK}} r$ iff $\delta_s \sim_{\text{HKK}} \delta_r$.*

Now we are ready to show that $\sim_{\text{HKK}}$ is not strictly coarser than $\equiv_{SP}$. Let μ and ν be two distributions as in Fig. 5. Let $R = \{(\mu, \nu), (\nu, \mu)\} \cup ID$. By Definition 12, it is easy to see that R is an HKK-bisimulation. Therefore $\mu \sim_{\text{HKK}} \nu$. However, $\mu \not\equiv_{SP} \nu$. For instance, from μ we can obtain a trace distribution $\{\frac{1}{3} : ac, \frac{1}{3} : bc, \frac{1}{3} : bd\}$, which cannot be simulated by any scheduler of ν. Since all states in μ have different enabled actions, the scheduler induced the trace distribution is a partial information scheduler.

5. CONCLUSION AND FUTURE WORK

In this paper, we have presented a novel and very coarse bisimulation called distribution-based bisimulation for probabilistic multiagent systems, which has interesting properties under two well-known subclasses of schedulers: it implies trace distribution equivalence under partial information schedulers, while it is compositional under decentralized schedulers. Working in the intersection of both scheduler classes ensures a restricted form of compositionality for a reasonably coarse bisimulation, where the restriction excludes undesired or unrealistically powerful schedulers.

As shown in [31], model checking ATL [3] against multiagent systems is EXPTIME-complete, when the systems are represented by certain reactive languages, say "Simple Reactive Modules Language" (SRML). Therefore, abstraction and state space minimization are particularly important for these systems. For future work we will explore algorithms to compute our bisimulation in a symbolic and compositional manner. Another direction of future research is to investigate the relation between distribution-based bisimulation and some popular logics in multiagent setting, for instance probabilistic ATL [9]. The complexity of deciding distribution-based bisimulations will also be interesting.

6. ACKNOWLEDGMENTS

The authors are supported by Australian Research Council under Grant DP130102764, the National Natural Science Foundation of China (Grant Nos. 61428208, 61472473 and 61361136002), AMSS-UTS Joint Research Laboratory for Quantum Computation, Chinese Academy of Sciences, and the CAS/SAFEA International Partnership Program for Creative Research Team.

7. REFERENCES

[1] T. Ågotnes, V. Goranko, and W. Jamroga. Alternating-time temporal logics with irrevocable strategies. In *TARK*, pages 15–24, 2007.

[2] O. I. Al-Bataineh and R. van der Meyden. Abstraction for epistemic model checking of dining cryptographers-based protocols. In *TARK*, pages 247–256. ACM, 2011.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

[4] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[5] E. Böde, M. Herbstritt, H. Hermanns, S. Johr, T. Peikenkamp, R. Pulungan, J. Rakow, R. Wimmer, and B. Becker. Compositional dependability evaluation for statemate. *IEEE Trans. Software Eng.*, 35(2):274–292, 2009.

[6] H. Boudali, P. Crouzen, and M. Stoelinga. A rigorous, compositional, and extensible framework for dynamic fault tree analysis. *IEEE Trans. Dependable Sec. Comput.*, 7(2):128–143, 2010.

[7] P. S. Castro and D. Precup. Using bisimulation for policy transfer in MDPs. In *AAAI*. AAAI Press, 2010.

[8] G. Chehaibar, H. Garavel, L. Mounier, N. Tawbi, and F. Zulian. Specification and Verification of the PowerScale™ bus arbitration protocol: An industrial experiment with lotos. In *FORTE*, pages 435–450, 1996.

[9] T. Chen and J. Lu. Probabilistic alternating-time temporal logic and model checking algorithm. In *FSKD*, pages 35–39. IEEE, 2007.

[10] L. De Alfaro. The verification of probabilistic systems under memoryless partial-information policies is hard. Technical report, DTIC Document, 1999.

[11] C. A. D. M. Delgado and M. R. F. Benevides. Verification of epistemic properties in probabilistic multi-agent systems. In *MATES*, volume 5774 of *Lecture Notes in Computer Science*, pages 16–28. Springer, 2009.

[12] Y. Deng and M. Hennessy. On the semantics of markov automata. *Information and Computation*, 222:139–168, 2013.

[13] L. Doyen, T. A. Henzinger, and J. Raskin. Equivalence of labeled markov chains. *Int. J. Found. Comput. Sci.*, 19(3):549–563, 2008.

[14] C. Eisentraut, H. Hermanns, and L. Zhang. On probabilistic automata in continuous time. In *LICS*, pages 342–351, 2010.

[15] L. Feng, T. Han, M. Kwiatkowska, and D. Parker. Learning-based compositional verification for synchronous probabilistic systems. In *ATVA*, volume 6996 of *Lecture Notes in Computer Science*, pages 511–521. Springer, 2011.

[16] Y. Feng and L. Zhang. When equivalence and bisimulation join forces in probabilistic automata. In *FM*, volume 8442 of *Lecture Notes in Computer Science*, pages 247–262. Springer, 2014.

[17] S. Giro and P. R. D'Argenio. Quantitative model checking revisited: neither decidable nor approximable. In *FORMATS*, volume 4763 of *Lecture Notes in Computer Science*, pages 179–194. Springer, 2007.

[18] P. R. Halmos. *Measure theory*, volume 1950. Springer, 1974.

[19] H. Hermanns, J. Krcál, and J. Kretínský. Probabilistic bisimulation: Naturally on distributions. In *CONCUR 2014*, volume 8704 of *Lecture Notes in Computer Science*. Springer, 2014.

[20] X. Huang and C. Luo. A logic of probabilistic knowledge and strategy. In *AAMAS*, pages 845–852. IFAAMAS, 2013.

[21] X. Huang, K. Su, and C. Zhang. Probabilistic alternating-time temporal logic of incomplete information and synchronous perfect recall. In *AAAI*. AAAI Press, 2012.

[22] P. Kazmierczak, T. Ågotnes, and W. Jamroga. Multi-agency is coordination and (limited) communication. In *PRIMA*, volume 8861 of *Lecture Notes in Computer Science*, pages 91–106. Springer, 2014.

[23] M. Kwiatkowska, G. Norman, D. Parker, and H. Qu. Assume-guarantee verification for probabilistic systems. In *TACAS*, volume 6015 of *Lecture Notes in Computer Science*, pages 23–37. Springer, 2010.

[24] N. Lynch, R. Segala, and F. Vaandrager. Observing branching structure through probabilistic contexts. *SIAM J. Comput.*, 37(4):977–1013, Sept. 2007.

[25] C. Phillips. Knowledge transfer in Markov decision processes. Technical report, Technical report, McGill University, 2006.

[26] W. Rudin. *Real and complex analysis*. Tata McGraw-Hill Education, 2006.

[27] H. Schnoor. Strategic planning for probabilistic games with incomplete information. In *AAMAS*, pages 1057–1064. IFAAMAS, 2010.

[28] R. Segala. A compositional trace-based semantics for probabilistic automata. In *CONCUR*, volume 962 of *Lecture Notes in Computer Science*, pages 234–248. Springer, 1995.

[29] R. Segala. *Modeling and Verification of Randomized Distributed Realtime Systems*. PhD thesis, MIT, 1995.

[30] W.-G. Tzeng. A polynomial-time algorithm for the equivalence of probabilistic automata. *SIAM J. Comput.*, 21(2):216–227, 1992.

[31] W. van der Hoek, A. Lomuscio, and M. Wooldridge. On the complexity of practical ATL model checking. In *AAMAS*, pages 201–208. ACM, 2006.

[32] W. van der Hoek and M. Wooldridge. Tractable multiagent planning for epistemic goals. In *AAMAS*, pages 1167–1174. ACM, 2002.

[33] H. P. van Ditmarsch, D. F. Duque, and W. van der Hoek. On the definability of simulability and bisimilarity by finite epistemic models. In *CLIMA*, pages 74–87. Springer, 2011.