LIPIcs Leibniz International Proceedings in Informatics

Verification and Refutation of Probabilistic Specifications via Games*

Mark Kattenbelt[1], **Michael Huth**[2]

[1] Oxford University Computing Laboratory
Wolfson Building, Parks Rd, Oxford, UK

[2] Department of Computing, Imperial College London
Huxley Building, 180 Queen's Gate, London, UK

ABSTRACT. We develop an abstraction-based framework to check probabilistic specifications of Markov Decision Processes (MDPs) using the stochastic two-player game abstractions (i.e. "games") developed by Kwiatkowska et al. as a foundation. We define an abstraction preorder for these game abstractions which enables us to identify many new game abstractions for each MDP — ranging from compact and imprecise to complex and precise. This added ability to trade precision for efficiency is crucial for scalable software model checking, as precise abstractions are expensive to construct in practice. Furthermore, we develop a four-valued probabilistic computation tree logic (PCTL) semantics for game abstractions. Together, the preorder and PCTL semantics comprise a powerful verification and refutation framework for arbitrary PCTL properties of MDPs.

1 Introduction

Model checking [5, 28] is a methodology for reasoning about the formal correctness of systems. The task of a model checker is to decide whether a *model* M satisfies a *property* ϕ. We write this as a *judgment* $M \models \phi$ and say a model checker *verifies* or *refutes* such judgments.

It is often intractable to verify or refute judgments involving large or infinite-state models directly. A recognised solution is to apply *abstraction*. That is, we can reason about the validity of $M \models \phi$ by model checking judgments involving abstractions A of M. Usually abstraction is used within an *abstraction-refinement loop* [7]. Starting with a very imprecise abstraction A this loop, if necessary, incrementally *refines* A until it is precise enough to either verify or refute $M \models \phi$. When model checking *software* one would typically use *existential abstractions* [8, 1], with which it is possible to verify a certain class of properties. However, to refute these properties, one has to concretise *abstract counter-examples* [7].

In this paper we focus on *probabilistic* models and properties. Unfortunately, counter-examples of probabilistic models are usually complex infinite structures [14]. Hence, refutation by concretising abstract counter-examples akin to existential abstractions is a lot more involved for probabilistic models [16]. This motivates us to consider abstraction schemes with which we can directly refute properties (c.f. *modal* or *mixed abstractions* [26, 10]). Such abstractions also potentially demonstrate the validity or falsity of $M \models \phi$ more compactly and more intuitively than probabilistic witnesses or counter-examples. That is, one may use abstractions as *diagnostic tools* or *certificates* demonstrating validity or falsity of $M \models \phi$ [27].

*This work was, in part, supported by UK EPSRC grants EP/D07956X/2 and EP/E028985/1.

Foundations of Software Technology and Theoretical Computer Science (Kanpur) 2009.
Editors: Ravi Kannan and K. Narayan Kumar; pp 251–262
Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany.
Digital Object Identifier: 10.4230/LIPIcs.FSTTCS.2009.2323

The models we consider are Markov decision processes (MDPs), which naturally model a wide range of probabilistic systems due to their ability to capture both *probabilistic* and *non-deterministic* choice. Our abstraction scheme is based on the *two-player stochastic game* (i.e. "game") abstractions suggested in [24]. In [24], these abstractions are used to over and under-approximate reachability probabilities of MDPs — as opposed to the verification and refutation of more complex properties.

Unfortunately, the definition of abstraction in [24] has a shortcoming. Given a particular partition of MDP states it only considers the optimal game over this partition. That is, other than the expected loss of precision that occurs due to joining MDP states, there is no mechanism that enables one to lose more precision — in fact, it is unclear what less precise abstractions over this partition would look like. Optimal abstractions over a partition are usually inefficient to represent and computationally expensive to construct. Moreover, the same job can often be accomplished with less precise abstractions. This is evident in most abstraction-based software model checkers which, for a fixed partition,[†] first consider a coarse abstraction over this partition and consider more precise abstractions over this partition only if this is necessary [1, 6]. Due to the shortcoming of [24] it is not possible to take such an approach with game abstractions and, as constructing optimal game abstractions is expensive, this significantly affects the scalability of, e.g., the method in [22].

Hence, the first issue we address in this paper is the development of an *abstraction preorder* for games which alleviates the shortcomings of [24]. Compared to the work in [24], this preorder identifies many additional game abstractions of varying precisions — even when restricted to a fixed partition. This opens up the possibility of adapting the method in [22] and other tools using game abstractions [25, 21, 20] to reason more efficiently via non-optimal games. Furthermore, instead of over and under-approximating reachability properties of MDPs, we develop a *four-valued probabilistic computation tree logic (PCTL) [15] semantics* for games and show our abstraction preorder preserves this semantics. Our abstraction preorder and PCTL semantics together comprise a powerful abstraction framework with which we can verify and refute arbitrary PCTL specifications of MDPs.

2 Background

Let AP be a fixed set of atomic propositions. Let $\mathbb{B}$ be the Boolean domain. Let $\mathbb{P}X$ be the powerset of a set X, excluding $\emptyset$. A probability distribution over X is a function $\lambda : X \to [0,1]$ such that $\sum_{x \in X} \lambda(x) = 1$ and the set $\{x \in X \mid \lambda(x) > 0\}$ is countable. Let $\mathbb{D}X$ be the set of all distributions over X. For $x \in X$ let $[x] \in \mathbb{D}X$ be the point distribution on x, i.e. $[x](x) = 1$. Every distribution over a countable set can be written as a countable sum of point distributions $\sum_i w_i \cdot [x_i]$.

We model four-valued logic [2] with a *must* (!) and *may* (?) modality of truth. Intuitively, ?-true corresponds to *possible truth* and !-true indicates *certain truth*. We represent *true* (resp. *false*) by being both !-true and ?-true (resp. !-false and ?-false). We represent *uncertainty* by being !-false and ?-true and *inconsistency* by being !-true and ?-false.

Given an arbitrary non-empty sequence $\pi = \omega_0; \omega_1; \omega_2, \ldots$ let $|\pi|$ be the number elements of π minus one. We let $\pi(i)$ be ω_i and, if $|\pi|$ is finite, let $\text{LAST}(\pi)$ be $\pi(|\pi|)$. Finally,

[†]That is to say, a fixed set of predicates when using predicate abstraction [1].

let π^i be the prefix of π such that $|\pi^i| = i$. We write $\pi;\pi'$ for the concatenation of sequences.

Markov decision processes We now introduce Markov decision processes (MDPs) which naturally model systems with both non-deterministic and probabilistic behaviours:

DEFINITION 1. *A* Markov decision process *(MDP) M is a tuple $\langle S, s_i, T, L\rangle$, where:*

- *S is a countable set of states;*
- *$s_i \in S$ is an* initial *state;*
- *$T \in S \to \mathbb{P}\mathbb{D}S$ is a* transition function*;*
- *$L \in S \times \text{AP} \to \mathbb{B}$ is a* labelling function.

The definition of $\mathbb{P}$ ensures totality*: i.e. $\forall s \in S : |T(s)| > 0$ (this totality requirement is not essential and is made for presentational reasons, only). We let $\mathcal{M}$ be the class of all MDPs.*

From a state $s \in S$, a non-deterministic choice *picks a distribution $\lambda \in T(s)$. Then, the next state $s' \in S$ is picked* probabilistically *according to λ. A* path *of M is a sequence over $S \cup \mathbb{D}S$ that strictly alternates between states and distributions as described. Let Π_M and Π_M^∞ be the set of all* finite *and* infinite *paths, respectively. For $\Omega \subseteq S \cup \mathbb{D}S$ we write, e.g., $\Pi_M(\Omega)$ to restrict to paths starting with an element in Ω.*

A strategy *of M is a partial function $\sigma\colon \Pi_M \to \mathbb{D}\mathbb{D}S$, with domain of definition all π with $\text{LAST}(\pi) \in S$, such that $\sigma(\pi) \in \mathbb{D}(T(\text{LAST}(\pi)))$. As is evident from the definition, we consider* randomised *strategies (i.e. strategies that resolve non-determinism with a probabilistic choice). Let Σ_M be all strategies of M. A path π is* consistent *with $\sigma \in \Sigma_M$ iff for all $i \leq |\pi| - 1$ with $\pi(i) \in S$ the probability $\sigma(\pi^i)(\pi(i+1))$ is positive. We write, e.g., $\Pi_{M,\sigma}$ to restrict to paths consistent with σ. For every $s \in S$ and $\sigma \in \Sigma_M$ we construct a probability measure $\mathbf{Pr}^s_{M,\sigma}$ over infinite paths $\Pi^\infty_{M,\sigma}(\{s\})$ with standard techniques [23].*

Probabilistic CTL We define an adequate PCTL fragment with unrestricted negation [15]:

DEFINITION 2. *A PCTL formula is defined with the following BNF-style syntax rules where $a \in \text{AP}$, $k \in \mathbb{N} \cup \{\infty\}$, $p \in [0,1]$ and $\bowtie \in \{\leq, <, \geq, >\}$:*

$$\phi ::= a \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \mathsf{P}_{\bowtie p}\langle\psi\rangle \qquad\qquad \psi ::= \mathsf{X}\phi \mid \phi_1 \mathsf{U}^{\leq k} \phi_2 \,.$$

We let Φ and Ψ be the set of all PCTL formulae of the form ϕ and ψ respectively.

PCTL semantics of MDPs Finally, we define standard PCTL semantics of MDPs via a satisfaction relation $\models \subseteq \mathcal{M} \times \Phi$ [3]:

DEFINITION 3. *Let $M = \langle S, s_i, T, L\rangle$ be an MDP, and let $\phi \in \Phi$ be a PCTL formula. We define satisfaction relations for states $\models \subseteq S \times \Phi$ and paths $\models \subseteq \Pi^\infty_M(S) \times \Psi$ as follows:*

$$\begin{array}{lcl}
\pi \models \mathsf{X}\phi & \iff & s_1 \models \phi \\
\pi \models \phi_1 \mathsf{U}^{\leq k} \phi_2 & \iff & \exists i \leq k : \big(s_i \models \phi_2 \wedge (\forall j < i : s_j \models \phi_1)\big) \\
s \models a & \iff & L(s,a) \\
s \models \neg\phi & \iff & s \not\models \phi \\
s \models \phi_1 \vee \phi_2 & \iff & (s \models \phi_1 \ \text{ or } \ s \models \phi_2) \\
s \models \mathsf{P}_{\triangleright p}\langle\psi\rangle & \iff & \inf_\sigma \ \mathbf{Pr}^s_{M,\sigma}\{\pi \in \Pi^\infty_{M,\sigma}(s) \mid \pi \models \psi\} \triangleright p \\
s \models \mathsf{P}_{\triangleleft p}\langle\psi\rangle & \iff & \sup_\sigma \mathbf{Pr}^s_{M,\sigma}\{\pi \in \Pi^\infty_{M,\sigma}(s) \mid \pi \models \psi\} \triangleleft p
\end{array}$$

with $\pi = s_0;\lambda_0;s_1;\lambda_1\ldots$, $\triangleright \in \{>, \geq\}$, $\triangleleft \in \{<, \leq\}$ and $\sigma \in \Sigma_M$. Moreover, $M \models \phi$ iff $s_i \models \phi$.

For any PCTL path formula, the set of paths satisfying it is measurable [30]. For MDPs, properties like $\mathsf{P}_{\bowtie p}\langle\psi\rangle$ are *universal* as threshold $\bowtie p$ must be met under *all* strategies. Conversely, properties like $\neg\mathsf{P}_{\bowtie p}\langle\psi\rangle$ hold iff there *exists* a strategy that violates the threshold.

We use standard methods to lift any relation to a relation over distributions [18]. For any relation $R \subseteq X \times Y$ we let $R_{\mathbb{D}} \subseteq \mathbb{D}X \times \mathbb{D}Y$ be the relation such that $\lambda_X \, R_{\mathbb{D}} \, \lambda_Y$ iff there is a weight function $\delta \in X \times Y \to [0,1]$ with (for all $x \in X$, $y \in Y$):

$$\lambda_X(x) = \textstyle\sum_{y' \in Y} \delta(x,y') \qquad \lambda_Y(y) = \textstyle\sum_{x' \in X} \delta(x',y) \qquad \delta(x,y) > 0 \Rightarrow x \, R \, y \tag{1}$$

3 Game-based abstraction framework

We now introduce the components of our abstraction framework. We start by formally defining a class of games ($\mathcal{G}$) and an *embedding function* ($emb : \mathcal{M} \to \mathcal{G}$), which casts MDPs into $\mathcal{G}$. We then define an *abstraction preorder* ($\sqsubseteq_p \subseteq \mathcal{G} \times \mathcal{G}$) as a relation over games. Our last component is a *four-valued PCTL semantics* ($\models^!, \models^? \subseteq \mathcal{G} \times \Phi$) over games. Finally, we show our components satisfy some necessary *soundness properties*. With these components and properties we then show how to verify and refute arbitrary PCTL properties of MDPs.

Stochastic two-player games ($\mathcal{G}$) In comparison to MDPs, games are equipped with an additional level of choice (i.e. their transition function yields *sets of sets* of distributions). Games also have four-valued propositional labelling (through a *must* and *may* labelling):

DEFINITION 4. *A* stochastic two-player game *G is a tuple $\langle S, s_i, T, L^!, L^? \rangle$, where:*

- *S is a countable set of states;*
- *$s_i \in S$ is an* initial *state;*
- *$T \in S \to \mathbb{P}\mathbb{P}\mathbb{D}S$ is a* transition function*;*
- *$L^!, L^? \in S \times \mathrm{AP} \to \mathbb{B}$ are* labelling functions.

By definition of $\mathbb{P}$ we ensure totality *for T: i.e. we have $|T(s)| > 0$ for all $s \in S$ and $|\Lambda| > 0$ for all $\Lambda \in T(s)$. Let $\mathcal{G}$ be the class of all games. We define* player 1 states *as elements of S,* player 2 states *as sets of distributions over player 1 states ($\mathbb{P}\mathbb{D}S$) and* probabilistic states *as distributions over player 1 states ($\mathbb{D}S$). From a player 1 state $s \in S$ player 1 can transition to a player 2 state $\Lambda \in \mathbb{P}\mathbb{D}S$ iff $\Lambda \in T(s)$ (written $s \to_1 \Lambda$). Analogously, from a player 2 state $\Lambda \in \mathbb{P}\mathbb{D}S$ player 2 can transition to a probabilistic state $\lambda \in \mathbb{D}S$ iff $\lambda \in \Lambda$ (written $\Lambda \to_2 \lambda$). Finally, from a probabilistic state $\lambda \in \mathbb{D}S$ the game transitions to a player 1 state $s' \in S$ with probability $\lambda(s')$ (written $\lambda \to_p s'$).*

A play *in G is a sequence of transitions[‡] and hence necessarily strictly alternates between player 1 states, player 2 states and probabilistic states. Let Π_G and Π_G^∞ be the set of all finite and infinite plays of G, respectively. For $\Omega \subseteq S \cup \mathbb{P}\mathbb{D}S \cup \mathbb{D}S$ we write, e.g., $\Pi_G(\Omega)$ to restrict to plays starting with an element in Ω.*

A player 1 strategy *is a partial function $\sigma_1 \in \Pi_G \to \mathbb{D}\mathbb{P}\mathbb{D}S$, with domain of definition all π with $\mathrm{LAST}(\pi) \in S$, such that $\sigma_1(\pi) \in \mathbb{D}(T(\mathrm{LAST}(\pi)))$. Analogously, a* player 2 strategy *is a partial function $\sigma_2 \in \Pi_G \to \mathbb{D}\mathbb{D}S$ with domain of definition all π with $\mathrm{LAST}(\pi) \in \mathbb{P}\mathbb{D}S$, such that $\sigma_2(\pi) \in \mathbb{D}(\mathrm{LAST}(\pi))$. We write Σ_G^1 and Σ_G^2 for the set of all player 1 and player*

[‡]We will manipulate plays as if they are sequences over $S \cup \mathbb{P}\mathbb{D}S \cup \mathbb{D}S$.

2 strategies, respectively. A play π of G is consistent *with $\sigma_1 \in \Sigma_G^1$ if for every $i \leq |\pi| - 1$ with $\pi(i) \in S$ the probability $\sigma_1(\pi^i)(\pi(i+1))$ is positive. Similarly, π is* consistent *with $\sigma_2 \in \Sigma_G^2$ if $\sigma_2(\pi^i)(\pi(i+1))$ is positive whenever $\pi(i) \in \mathbb{PD}S$. For $\sigma_1 \in \Sigma_G^1$ and $\sigma_2 \in \Sigma_G^2$ we write, e.g., $\Pi_{G,\sigma_1,\sigma_2}$ to restrict to plays consistent with σ_1 and σ_2. For $\sigma_1 \in \Sigma_G^1$ and $\sigma_2 \in \Sigma_G^2$ and $\Pi \subseteq \Pi_{G,\sigma_1,\sigma_2}$, we denote with $\Pi^{\uparrow_{\sigma_1,\sigma_2}}$ the infinite plays of G that are consistent with both σ_1 and σ_2 and have a prefix in Π.*

Given strategies $\sigma_1 \in \Sigma_G^1$ and $\sigma_2 \in \Sigma_G^2$ the behaviour of G is purely probabilistic. Hence, for each $\omega \in S \cup \mathbb{PD}S \cup \mathbb{D}S$, using standard techniques [23], we construct a probability space over infinite plays $\Pi^{\infty}_{G,\sigma_1,\sigma_2}(\{\omega\})$ with probability measure $\mathbf{Pr}^{\omega}_{G,\sigma_1,\sigma_2}$ such that $\mathbf{Pr}^{\omega}_{G,\sigma_1,\sigma_2}(\{\omega\}^{\uparrow_{\sigma_1,\sigma_2}}) = 1$ and, for every finite play of non-zero length $\pi \in \Pi_{G,\sigma_1,\sigma_2}(\{\omega\})$:

$$\mathbf{Pr}^{\omega}_{G,\sigma_1,\sigma_2}(\{\pi\}^{\uparrow_{\sigma_1,\sigma_2}}) = \begin{cases} \mathbf{Pr}^{\omega}_{G,\sigma_1,\sigma_2}(\{\pi'\}^{\uparrow_{\sigma_1,\sigma_2}}) \cdot \sigma_1(\pi')(\Lambda') & \text{if } (\pi = \pi' \to_1 \Lambda') \\ \mathbf{Pr}^{\omega}_{G,\sigma_1,\sigma_2}(\{\pi'\}^{\uparrow_{\sigma_1,\sigma_2}}) \cdot \sigma_2(\pi')(\lambda') & \text{if } (\pi = \pi' \to_2 \lambda') \\ \mathbf{Pr}^{\omega}_{G,\sigma_1,\sigma_2}(\{\pi'\}^{\uparrow_{\sigma_1,\sigma_2}}) \cdot \mathrm{LAST}(\pi')(s') & \text{if } (\pi = \pi' \to_p s') \end{cases}$$

REMARK 5. *In figures we depict player 1 states with big open circles, player 2 states with small filled squares and probabilistic states with filled black circles. Labels depict the probability of transitions (omitted for point distributions). We write $a!$, $a?$ and $a!?$ next to s iff $L^!(s,a) \wedge \neg L^?(s,a)$, $\neg L^!(s,a) \wedge L^?(s,a)$ or $L^!(s,a) \wedge L^?(s,a)$ resp., and nothing otherwise.*

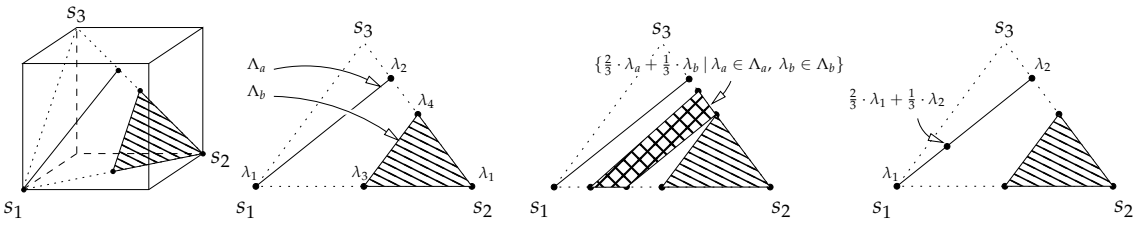
The roles of player 1 & 2 Before we define the components of our abstraction framework we first give an informal account of how a game $\hat{G}$ (over states $\hat{S}$) abstracts an MDP M (over states S). Intuitively, to be sound for PCTL — or any unrestricted branching-time logic — $\hat{G}$ must both under and over-approximate the strategies that are feasible in M. Observe that a strategy of M is simply a particular resolution of non-determinism in M and hence, to under and over-approximate feasible strategies of M, $\hat{G}$ must under and over-approximate the non-deterministic choice $T(s) \in \mathbb{PD}S$ in each state of M.[§] We use player 1 states of $\hat{G}$ to represent sets of states of M and player 2 states of $\hat{G}$ to approximate the non-deterministic choices of these states (i.e. player 2 resolves the non-determinism of M). Informally, in $\hat{s} \in \hat{S}$ player 1 can choose from $\hat{T}(\hat{s}) \in \mathbb{PPD}\hat{S}$ at least one player 2 state that under-approximates $T(s)$ and one player 2 state that over-approximates $T(s)$ for every concrete state $s \in S$ of M that $\hat{s}$ abstracts. As $\hat{s}$ may abstract many states of M, and for each such state we have both under and over-approximating player 2 choices, there may be many player 1 choices in $\hat{T}(\hat{s})$. Hence, player 1 resolves non-determinism introduced by abstraction.

The embedding function (*emb*) The first component of our framework is an *embedding function* $emb : \mathcal{M} \to \mathcal{G}$, which yields an exact representation $emb(M)$ in $\mathcal{G}$ for each MDP M. The embedding function allows us to treat MDPs as a special kind of game.

DEFINITION 6. *Let $emb \in \mathcal{M} \to \mathcal{G}$ be the function which for every MDP $M = \langle S, s_i, T, L \rangle$ yields a game $G = \langle S, s_i, \hat{T}, L, L \rangle$ such that $\hat{T}(s) = \{T(s)\}$ for every $s \in S$.*

Embedded MDPs are exact representations of MDPs in $\mathcal{G}$. Intuitively, we ascribe all of M's non-determinism to player 2. That is, player 2 strategies Σ_G^2 have a one-to-one correspondence with Σ_M. Moreover, player 1 has no power (i.e. $|\Sigma_G^1| = 1$).

[§]Below, we formalise this under/over-approximation in the definition of the preorder $\sqsubseteq_p$.



(a) Player 1 choices in $T(s_1) = \{\hat{\Lambda}_a, \hat{\Lambda}_b\}$. (b) Combined player 1 (left) and 2 (right) transitions.

Figure 1: A geometrical interpretation of games.

EXAMPLE 7. *Consider a program with two 8-bit unsigned integers x, y which first initialises y non-deterministically and then assigns to x, uniformly at random, a number between 0 and 255. We model this program with an MDP M and depict $emb(M)$ in Fig. 3(a).*

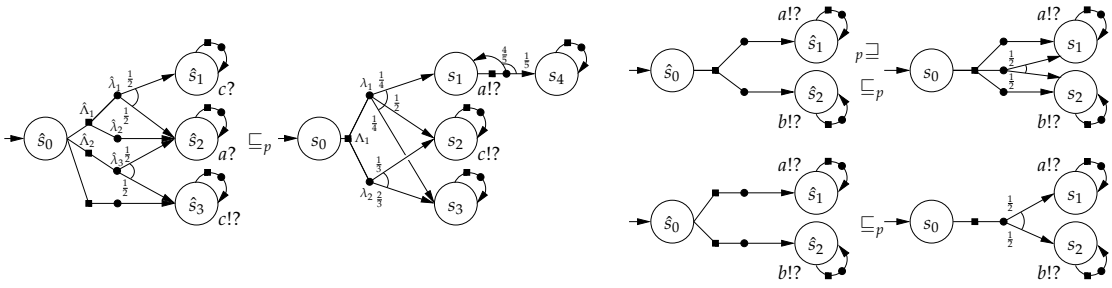
Combined player 1 & 2 transitions Prior to introducing our next component — the abstraction preorder $\sqsubseteq_p$ — we need to introduce *combined transitions*. Combined transitions will enable the preorder take into account that players can make *probabilistic* choices. Combined transitions are well understood for MDPs but are less well-known in games. To explain combined transitions observe that we can interpret probability distributions $\lambda \in \mathbb{D}S$ geometrically as points on a plane in $|S|$-dimensional Euclidean space [27]. Hence, we can interpret the choices available to player 2 and player 1 as a *set of points* and a *set of set of points*, respectively. We illustrate in Fig. 1(a) the choice $T(s_1) = \{\Lambda_a, \Lambda_b\} = \{\{\lambda_1, \lambda_2\}, \{\lambda_3, \lambda_4, \lambda_5\}\}$ available to player 1 in a state s_1 (over a state space $\{s_1,s_2,s_3\}$).

In the player 2 state $\Lambda_a \in \mathbb{P}\mathbb{D}S$ player 2 can make a *probabilistic* choice over probabilistic states $\{\lambda_1, \lambda_2\}$. Hence, it is appropriate to think of Λ_a as defining the hull of a convex shape from which player 2 can draw any probabilistic state (see Fig. 1(b)). To formalise this, akin to [29], we introduce *combined player 2 transitions*. A combined player 2 transition is a move from a player 2 state $\Lambda \in \mathbb{P}\mathbb{D}S$ to a probabilistic state $\lambda \in \mathbb{D}S$, denoted $\Lambda \rightarrow_1^C \lambda$, iff for some $\sum_i w_i \cdot [\lambda_i] \in \mathbb{D}\Lambda$ we have $\lambda = \sum_i w_i \cdot \lambda_i$.

Because player 2 choices are interpreted as convex shapes, the choice available to player 1 in s_1 is a set of convex shapes $T(s_1) = \{\Lambda_a, \Lambda_b\}$. Player 1 can also take any weighted combination of these convex shapes (see Fig. 1(b)). We extend the existing notion of combined transitions over sets of distributions from [29] to combined transitions over sets of sets of distributions as follows: a *combined player 1 transition* is a move from a player 1 state $s \in S$ to a player 2 state $\Lambda \in \mathbb{P}\mathbb{D}S$, denoted $s \rightarrow_1^C \Lambda$, if and only if for some $\sum_i w_i \cdot [\Lambda_i] \in \mathbb{D}(T(s))$ we have $\Lambda = \{\sum_i w_i \cdot \lambda_i \mid \lambda_i \in \Lambda_i \text{ for all } i\}$.

The abstraction preorder ($\sqsubseteq_p$) We can now define the *abstraction preorder* — a relation $\sqsubseteq_p \subseteq \mathcal{G} \times \mathcal{G}$ over games. Intuitively, this preorder defines a notion of precision in $\mathcal{G}$; that is, $\hat{G} \sqsubseteq_p G$ has the meaning that $\hat{G}$ is *less precise* (i.e. *an abstraction of*) G. We can therefore employ the embedding function to define when a game $\hat{G}$ abstracts an MDP M (i.e. when $\hat{G} \sqsubseteq_p emb(M)$).

We define $\sqsubseteq_p$ through a new notion of simulation over games. We consider simulations over disjoint unions $\hat{G} \oplus G$ of games (defined in the obvious way):



(a) Game $\hat{G}$ and embedding $emb(M)$ s.t. $\hat{G} \sqsubseteq_p emb(M)$. (b) Pairs of games $\hat{G}$ and G in $\sqsubseteq_p$.

Figure 2: Games illustrating various points in the paper.

DEFINITION 8. *Let $G = \langle S, s_i, T, L^!, L^? \rangle$ be a game and let $R \subseteq S \times S$ be a relation on S. We call R a* strong probabilistic game-simulation *iff for all s' R s the following conditions hold:*

(i) $L^!(s', a) \Rightarrow L^!(s, a)$ *for all* $a \in \text{AP}$
(ii) $L^?(s', a) \Leftarrow L^?(s, a)$ *for all* $a \in \text{AP}$
(iii) $\forall s \rightarrow_1 \Lambda : \exists s' \rightarrow_1^C \Lambda' : \forall \Lambda' \rightarrow_2 \lambda' : \exists \Lambda \rightarrow_2^C \lambda : \lambda' \, R_{\mathbb{D}} \, \lambda$
(iv) $\forall s \rightarrow_1 \Lambda : \exists s' \rightarrow_1^C \Lambda' : \forall \Lambda \rightarrow_2 \lambda : \exists \Lambda' \rightarrow_2^C \lambda' : \lambda' \, R_{\mathbb{D}} \, \lambda$

Moreover, for games $\hat{G} = \langle \hat{S}, \hat{s}_i, \hat{T}, \hat{L}^!, \hat{L}^? \rangle$ and $G = \langle S, s_i, T, L^!, L^? \rangle$ we let $\hat{G} \sqsubseteq_p G$ iff the largest[¶] *strong probabilistic game-simulation R on $\hat{G} \oplus G$ includes $\hat{s}_i \, R \, s_i$.*

Intuitively, the meaning of $\hat{s} \, R \, s$ is that $\hat{s}$ abstracts s. Conditions (i) and (ii) ensure that the labelling in $\hat{s}$ soundly approximates that of s. The innermost quantifier pair in (iii) formally defines under-approximation of player 2 states: i.e. $\hat{\Lambda} \in \mathbb{PD}\hat{S}$ under-approximates $\Lambda \in \mathbb{PD}S$ iff all transitions $\hat{\Lambda} \rightarrow_2 \hat{\lambda}$ can be simulated by a combined player 2 transition $\Lambda \rightarrow_2^C \lambda$. That is, for these player 2 states, player 2 in G is more powerful than player 2 in $\hat{G}$. The innermost quantifier pair of (iv) defines over-approximation analogously.

Recall that player 1 transitions $s \rightarrow_1 \Lambda$ are such that Λ represents an under or over-approximation of non-determinism in an MDP state that s abstracts. As $\hat{s}$ abstracts all MDP states that s abstracts, player 1 in $\hat{s}$ must both under and over-approximate all player 1 transitions $s \rightarrow_1 \Lambda$ with some combined player 1 move $\hat{s} \rightarrow_1^C \hat{\Lambda}$. This under/over-approximation is realised by the outermost quantifier pair of (iii) and (iv), respectively.

EXAMPLE 9. *Consider games $\hat{G}$ and $emb(M)$ in Fig. 2(a). The largest strong prob. game-simulation R over $\hat{G} \oplus emb(M)$ trivially includes $\langle \hat{s}_1, s_2 \rangle$, $\langle \hat{s}_1, s_3 \rangle$, $\langle \hat{s}_2, s_1 \rangle$, $\langle \hat{s}_2, s_3 \rangle$, $\langle \hat{s}_2, s_4 \rangle$ and $\langle \hat{s}_3, s_2 \rangle$. To see $\hat{s}_0 \, R \, s_0$, i.e. $\hat{G} \sqsubseteq_p emb(M)$, observe that (iii) for $\Lambda_1 \in T(s_0)$ is satisfied by $\hat{\Lambda}_2 \in \hat{T}(\hat{s}_0)$ as $\hat{\lambda}_3 \, R_{\mathbb{D}} \, \lambda_1$ and (iv) is satisfied by $\hat{\Lambda}_1 \in \hat{T}(\hat{s}_0)$ as $\hat{\lambda}_1 \, R_{\mathbb{D}} \, \lambda_1$ and $\frac{2}{3} \cdot \hat{\lambda}_1 + \frac{1}{3} \cdot \hat{\lambda}_2 \, R_{\mathbb{D}} \, \lambda_2$.*

Generality of $\sqsubseteq_p$ Intuitively, in [24], the non-determinism in each MDP state that $\hat{s}$ abstracts is exactly approximated (i.e. both under and over-approximated) by a normal player 1 transition $\hat{s} \rightarrow_1 \hat{\Lambda}$. Our main ability to lose precision arises from the ability to under/over-approximate $T(s')$ with separate player 1 transitions (in combination with the use of *combined* transitions). We illustrate the use of combined transitions with two examples.

[¶]The *largest* strong game-simulation in a game is the union of all its strong game-simulations.

Firstly, the use of combined transitions allows us to abstract probabilistic choice with player 1 non-determinism (see Fig. 2(b) (bottom)). As it is expensive to abstract probabilistic choice it may be advantageous to initially abstract probabilistic behaviour in this way.

Secondly, observe the equivalence classes of $\sqsubseteq_p$ define a notion of equivalence in $\mathcal{G}$ (i.e. G and G' are equivalent iff $G \sqsubseteq_p G'$ and $G' \sqsubseteq_p G$). Fig. 2(b) (top) illustrates that through this equivalence we can consider more compact representations without losing any precision.

Abstract PCTL semantics ($\models^!, \models^?$) The final component of our abstraction framework is a four-valued abstract PCTL semantics $\models^!, \models^? \subseteq \mathcal{G} \times \Phi$ for games. Informally, $G \models^! \phi$ (resp. $G \models^? \phi$) holds only if all MDPs that G abstracts *must* (resp. *may*) satisfy ϕ.

DEFINITION 10. *Let $G = \langle S, s_i, T, L^!, L^? \rangle$ be a game and let $\phi \in \Phi$ be a PCTL formula. We define must/may relations for states $\models^!, \models^? \subseteq S \times \Phi$ and plays $\models^!, \models^? \subseteq \Pi^\infty_G(S) \times \Psi$ as follows (letting $* \in \{!, ?\}$, $\neg! = ?$ and $\neg? = !$):*

$$
\begin{array}{lcl}
\pi \models^* \mathsf{X}\phi & \iff & s_1 \models^* \phi \\
\pi \models^* \phi_1 \mathsf{U}^{\trianglelefteq k} \phi_2 & \iff & \exists i \trianglelefteq k : \ (s_i \models^* \phi_2 \wedge (\forall j < i : s_j \models^* \phi_1) \\
s \models^* a & \iff & L^*(s, a) \\
s \models^* \neg\phi & \iff & s \not\models^{\neg *} \phi \\
s \models^* \phi_1 \vee \phi_2 & \iff & (s \models^* \phi_1 \ \ or \ \ s \models^* \phi_2) \\
s \models^! \mathsf{P}_{\triangleright p}\langle\psi\rangle & \iff & \inf_{\sigma_1} \ \inf_{\sigma_2} \ \mathbf{Pr}^s_{G,\sigma_1,\sigma_2}\{\pi \in \Pi^\infty_{G,\sigma_1,\sigma_2}(s) \mid \pi \models^! \psi\} \triangleright p \\
s \models^? \mathsf{P}_{\triangleright p}\langle\psi\rangle & \iff & \sup_{\sigma_1} \inf_{\sigma_2} \ \mathbf{Pr}^s_{G,\sigma_1,\sigma_2}\{\pi \in \Pi^\infty_{G,\sigma_1,\sigma_2}(s) \mid \pi \models^? \psi\} \triangleright p \\
s \models^! \mathsf{P}_{\triangleleft p}\langle\psi\rangle & \iff & \sup_{\sigma_1} \sup_{\sigma_2} \ \mathbf{Pr}^s_{G,\sigma_1,\sigma_2}\{\pi \in \Pi^\infty_{G,\sigma_1,\sigma_2}(s) \mid \pi \models^? \psi\} \triangleleft p \\
s \models^? \mathsf{P}_{\triangleleft p}\langle\psi\rangle & \iff & \inf_{\sigma_1} \ \sup_{\sigma_2} \ \mathbf{Pr}^s_{G,\sigma_1,\sigma_2}\{\pi \in \Pi^\infty_{G,\sigma_1,\sigma_2}(s) \mid \pi \models^! \psi\} \triangleleft p
\end{array}
$$

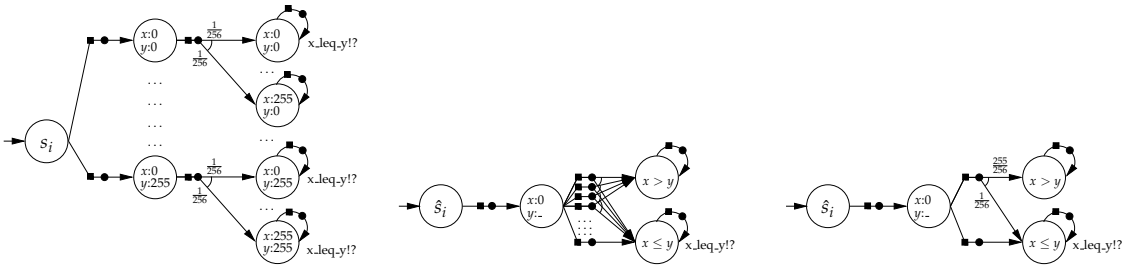
with $\pi = s_0; \Lambda_0; \lambda_0; s_1 \ldots, \triangleright \in \{>, \geq\}$, $\triangleleft \in \{<, \leq\}$, $\sigma_i \in \Sigma^i_G$. Moreover, $G \models^* \phi$ iff $s_i \models^* \phi$.

The four-valued semantics of propositional and temporal operators is standard. The only non-standard semantics is that of the probabilistic operator $\mathsf{P}_{\bowtie p}\langle\psi\rangle$. Recall $\mathsf{P}_{\bowtie p}\langle\psi\rangle$ holds for an MDP if *all* MDP-schedulings meet the threshold $\bowtie p$. As player 2 represents MDP non-determinism, for a lower threshold $\triangleright p$ (upper threshold $\triangleleft p$) we take the infimum (supremum) over player 2 strategies, regardless of whether we are evaluating in the must or may modality. In contrast, whether we take the infimum or supremum over player 1 strategies depends only on the modality. That is, if we are evaluating in the must modality we quantify pessimistically over player 1 strategies (inf. for $\triangleright p$, sup. for $\triangleleft p$) and in the may modality we quantify optimistically over player 1 strategies (sup. for $\triangleright p$, inf. for $\triangleleft p$). For lower thresholds, the modality in which we evaluate path properties corresponds to the modality in which we are evaluating — this has to be inverted for upper thresholds $\triangleleft p$.

Soundness properties Before we can verify and refute judgments over MDPs via games, we need to show our components satisfy the following properties:

LEMMA 11. *For all MDPs M and $\phi \in \Phi$ we have $M \models \phi \Leftrightarrow emb(M) \models^! \phi \Leftrightarrow emb(M) \models^? \phi$.*

PROOF. Follows directly from the fact that embedded MDPs have two-valued propositional labelling, i.e. $L^! = L^?$, and one trivial player 1 strategy, i.e. $|\Sigma^1_G| = 1$. ∎



(a) (b) (c)

Figure 3: The MDP (a) and the two game abstractions (b) and (c) from Example 14.

THEOREM 12. *For all games $\hat{G}, G$ such that $\hat{G} \sqsubseteq_p G$ and all PCTL properties $\phi \in \Phi$ we have $(\hat{G} \models^! \phi \Rightarrow G \models^! \phi)$ and $(\hat{G} \not\models^? \phi \Rightarrow G \not\models^? \phi)$.*

PROOF. We only sketch the structure of the proof here. Let $\hat{s}$ be any state of $\hat{G}$ and let s be any state of G, respectively. The proof shows that $\hat{s} \models^! \phi \Rightarrow s \models^! \phi$ and, dually, that $s \models^? \phi \Rightarrow \hat{s} \models^? \phi$. The main complexity of the proof is due to properties $\mathsf{P}_{\bowtie p}\langle\psi\rangle$.

Intuitively, due to (iii) of Def. 8, for any player 1 strategy in G, player 1 in $\hat{G}$ can choose a strategy under which it knows player 2 must be less powerful in $\hat{G}$ than in G, which ensures $s \models^? \mathsf{P}_{\bowtie p}\langle\psi\rangle \Rightarrow \hat{s} \models^? \mathsf{P}_{\bowtie p}\langle\psi\rangle$. Dually, (iv) of Def. 8 guarantees that for every player 1 strategy in G, player 1 in $\hat{G}$ can choose a strategy under which it knows player 2 is more powerful in $\hat{G}$ than it is in G, which ensures $\hat{s} \models^! \mathsf{P}_{\bowtie p}\langle\psi\rangle \Rightarrow s \models^! \mathsf{P}_{\bowtie p}\langle\psi\rangle$. ∎

Lem. 11 ensures consistency across the two representations of MDPs whereas Th. 12 ensures that any property that is $\models^!$-satisfied (not $\models^?$-satisfied) by a game $\hat{G}$ is also $\models^!$-satisfied (not $\models^?$-satisfied) by any game that is less abstract than $\hat{G}$ — including MDP embeddings.

Verification & refutation via games We can now *verify* the judgment $M \models \phi$ by constructing a game G that abstracts M (i.e. $G \sqsubseteq_p \mathit{emb}(M)$) and that !-satisfies ϕ (i.e. $G \models^! \phi$). It is easy to see this: by Th. 12 $\mathit{emb}(M) \models^! \phi$ and by Lem. 11 this is equivalent to $M \models \phi$. Analogously, we can *refute* $\mathcal{M} \models \phi$ by finding a game G such that $G \sqsubseteq_p \mathit{emb}(M)$ and $G \not\models^? \phi$ (i.e. by Th. 12 $\mathit{emb}(M) \not\models^? \phi$; by Lem. 11 $M \not\models \phi$).

For some games G it may be that both $G \not\models^! \phi$ and $G \models^? \phi$. In this case we can neither verify nor refute $M \models \phi$ and we need to consider *refining* G.

EXAMPLE 13. *For $\hat{G}$ and $\mathit{emb}(M)$ of Fig. 2(a), as $\hat{G} \sqsubseteq_p \mathit{emb}(M)$ and $\hat{G} \models^! \neg\mathsf{P}_{>0.5}\langle\mathsf{X}a\rangle$, using Lem. 11 and Th. 12, we have verified $M \models \neg\mathsf{P}_{>0.5}\langle\mathsf{X}a\rangle$: some scheduling of M satisfies $\mathsf{X}a$ with probability of at most 0.5. However, as $\hat{G} \not\models^! \mathsf{P}_{\leq 0.5}\langle\mathsf{X}a\rangle$ and $\hat{G} \models^? \mathsf{P}_{\leq 0.5}\langle\mathsf{X}a\rangle$ we can neither verify nor refute via $\hat{G}$ whether this threshold holds for* all *schedulings of M.*

Finally, once we find a game G via which we, say, verify $M \models \phi$, we can claim this judgment holds and use G as a certificate to our claim. To confirm our claim one would have to perform the checks $G \sqsubseteq_p \mathit{emb}(M)$ and $G \models^! \phi$. Analogously, we can use games as refutation certificates. We demonstrate our framework with a final motivating example:

EXAMPLE 14. *Reconsider the program and embedded MDP $emb(M)$ of Example 7. Suppose we wish to check the program stops with $x \leq y$ with a probability greater than* 0.003 *(i.e. $M \models \mathsf{P}_{>0.003}\langle true\, \mathsf{U}^{\leq\infty}\, x\_leq\_y\rangle$). We consider a partition which joins all states before the probabilistic assignment and which, after the assignment, divides states according to the predicate $x \leq y$. Fig. 3(b) depicts the (optimal) game G constructed with the techniques in [24]. We can verify our judgment with this game; however, G has many transitions because the probability of $x \leq y$ is different for each initial value of y. With the framework presented in this paper we can verify the judgment with the game $\hat{G}$ depicted in Fig. 3(c) — a much more compact game defined over the same partition.*

4 Discussion and conclusions

We motivated the need for an abstraction-based framework for the verification and refutation of PCTL specifications of MDPs. We constructed such a framework by taking the game abstractions from [24], developing an abstraction preorder and abstract PCTL semantics for these games, and proving these components meet certain soundness properties.

This preorder enables us to lose precision — even for a fixed partitioning of MDP states. This allows us to verify and refute properties of MDPs with more compact games. In many cases losing precision is essential. For example, when abstracting program statements under predicates that contain non-linear arithmetic, computing the optimal abstraction for a set of predicates is very inefficient. Through our abstraction preorder, by losing precision, we may be able to obtain abstractions more efficiently in such cases — for example by considering incrementally precise abstractions over a fixed partition, akin to software model checkers. However, to automate this we need to augment our framework with a refinement procedure. Although procedures exists to refine optimal partition abstractions of games [21, 22], we have not yet adapted these procedures to deal with the additional causes of imprecision that occur in our framework.

The game abstractions considered in practice (in, e.g., [22]) are known to be abstractions by construction — there is no need to check the conditions of $\sqsubseteq_p$. Nevertheless, the computational complexity of deciding $\sqsubseteq_p$ and $\models^!, \models^?$ are still of interest. In a preliminary unpublished version of this paper [19] we show $\sqsubseteq_p$ without combined player 1 transitions is decidable in P and $\models^!, \models^?$ are decidable in NP $\cap$ co-NP.

There is potential to improve precision of our framework as follows: one could equip games with separate must/may transition functions to distinguish under-approximating from over-approximating player 2 states (c.f. [26, 10]). That is, in (iii) of Def. 8 and in the must evaluations of Def. 10 one would use the must transitions and in (iv) of Def. 8 and in the may evaluations of Def. 10 one would use the may transitions. This change would not increase precision for partition abstractions.

Related work In recent papers, many orthogonal challenges related to game abstractions have been addressed: in [21, 22] it is outlined how optimal game abstractions can be constructed from language-level descriptions via SAT; in [20, 22] it is explained how good partition abstractions can be found using automated abstraction refinement.

For probabilistic systems, abstraction frameworks include probabilistic extensions of existential abstraction [11, 16], where MDP are abstracted by MDPs again through the strong

(probabilistic) simulation preorder of [18, 29]. Other frameworks abstract probabilities with intervals (e.g. [18, 17, 13, 4]). When considering these frameworks as an abstraction framework for MDPs we observe the abstract models are unable to distinguish non-determinism of the concrete MDP from the non-determinism that arises through abstraction. As a result, refuting a property $\mathsf{P}_{\leq p}\langle\psi\rangle$, i.e. showing there exists a strategy that exceeds p, can only be achieved by establishing *all* strategies exceed p. As this may not be true for some MDPs the property may not be refutable with these abstractions. By separating the two kinds of non-determinism, our framework does not suffer from the same problem. Note that our argument relies on the presence of non-determinism and does not occur when considering the abstraction schemes in, e.g., [29, 17, 13] as abstraction frameworks for Markov chains (MCs). In fact, our preorder is essentially the strong probabilistic simulation of [29] when restricted to MCs. Finally, we mention [4] in which more efficiently checkable abstractions are obtained by eliminating non-determinism (i.e. MDPs are abstracted by MCs).

For non-probabilistic systems, sound verification and refutation of temporal logics have mostly been developed in a (sometimes implicit) three-valued setting [26, 10]. Our results in the probabilistic setting are, notably, informed by work on modal/mixed transitions system [26, 10] and three-valued abstraction of games [12].

References

[1] T. Ball, T. Millstein, and S. K. Rajamani. Automatic predicate abstraction of C programs. *SIGPLAN Notices*, 36(5):203–213, 2001.

[2] N. D. Belnap. A useful four-valued logic. In *Modern Uses of Multiple-valued Logics*, pp. 8–37, 1977.

[3] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proc. of FSTTC '95*, vol. 1026 of *LNCS*, pp. 499–513, 1995.

[4] R. Chadha, M. Viswanathan, and R. Viswanathan. Least upper bounds for probability measures and their applications to abstractions. In *Proc. of CONCUR '08*, vol. 5201 of *LNCS*, pp. 264–278, 2008.

[5] E. Clarke and E. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In *Logic of Programs: Workshop*, vol. 131 of *LNCS*, 1981.

[6] E. Clarke, D. Kroening, N. Sharygina, and K. Yorav. SATABS: SAT-based predicate abstraction for ANSI-C. In *Proc. of TACAS '05*, vol. 3440 of *LNCS*, pp. 570–574, 2005.

[7] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In *Proc. of CAV '00*, vol. 1855 of *LNCS*, pp. 154–169, 2000.

[8] E. M. Clarke, O. Grumberg, and D. E. Long. Model checking and abstraction. *ACM TOPLAS*, 16(5):1512–1542, 1994.

[9] A. Condon. The complexity of stochastic games. *Inf. Comput.*, 96(2):203–224, 1992.

[10] D. Dams, R. Gerth, and O. Grumberg. Abstract interpretation of reactive systems. *ACM TOPLAS*, 19(2):253–291, 1997.

[11] P. R. D'Argenio, B. Jeannet, H. E. Jensen, and K. G. Larsen. Reduction and refinement strategies for probabilistic systems. In *Proc. of PAPM-PROBMIV '02*, vol. 2399 of *LNCS*, pp. 57–76, 2002.

[12] L. de Alfaro, P. Godefroid, and R. Jagadeesan. Three-valued abstractions of games: Uncertainty, but with precision. In *Proc. of LICS '04*, pp. 170–179, 2004.
[13] H. Fecher, M. Leucker, and V. Wolf. Don't know in probabilistic systems. In *Proc. of SPIN '06*, vol. 3925 of *LNCS*, pp. 71–88, 2006.
[14] T. Han and J. Katoen. Counterexamples in probabilistic model checking. In *Proc. of TACAS '07*, vol. 4424 of *LNCS*, pp. 72–86, 2007.
[15] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:512–535, 1994.
[16] H. Hermanns, B. Wachter, and L. Zhang. Probabilistic CEGAR. In *Proc. of CAV '08*, vol. 5123 of *LNCS*, pp. 162–175, 2008.
[17] M. Huth. On finite-state approximants for probabilistic computation tree logic. *TCS*, 346(1):113–134, 2005.
[18] B. Jonsson and K. G. Larsen. Specification and refinement of probabilistic processes. In *Proc. of LICS '91*, pp. 266–277, 1991.
[19] M. Kattenbelt and M. Huth. Abstraction framework for Markov decision processes and PCTL via games. TR RR-09-01, OUCL, 2009.
[20] M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker. A game-based abstraction-refinement framework for Markov decision processes. TR RR-08-06, OUCL, 2008.
[21] M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker. Game-based probabilistic predicate abstraction in PRISM. In *Proc. of QAPL '08*, vol. 220 of *ENTCS*, pp. 5–21, 2008.
[22] M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker. Abstraction refinement for probabilistic programs. In *Proc. of VMCAI '09*, vol. 5403 of *LNCS*, pp. 182–197, 2009.
[23] J. G. Kemeny, J. L. Snell, and A. W. Knapp. *Denumerable Markov Chains*. 2 edition, 1976.
[24] M. Kwiatkowska, G. Norman, and D. Parker. Game-based abstraction for Markov decision processes. In *Proc. of QEST '06*, pp. 157–166, 2006.
[25] M. Kwiatkowska, G. Norman, and D. Parker. Stochastic games for verification of probabilistic timed automata. In *Proc. of FORMATS '09*, pp. 212–227, 2009.
[26] K. G. Larsen and B. Thomsen. A modal process logic. In *Proc. of LICS '88*, pp. 203–210, 1988.
[27] A. K. McIver, C. C. Morgan, and C. Gonzalia. Proofs and refutations for probabilistic refinement. In *Proc. of FM '08*, vol. 5014 of *LNCS*, pp. 100–115, 2008.
[28] J. Quielle and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proc. of Symposium on Programming '05*, vol. 137 of *LNCS*, pp. 337–351, 1982.
[29] R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. In *Proc. of CONCUR '94*, LNCS, pp. 481–496, 1994.
[30] M. Vardi. Verification of probabilistic concurrent finite-state programs. In *Proc. of FOCS '85)*, pp. 327–338, 1985.