Verification and Refutation of Probabilistic Specifications via Games

Mark Kattenbelt¹, Michael Huth²

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 $\phi$. 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].
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) 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 \( \mathcal{AP} \) be a fixed set of atomic propositions. Let \( \mathbb{B} \) be the Boolean domain. Let \( \mathcal{P}X \) 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 \( \mathcal{D}X \) be the set of all distributions over \( X \). For \( x \in X \) let \( [x] \in \mathcal{D}X \) 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 \( \pi \) minus one. We let \( \pi(i) \) be \( \omega_i \) and, if \( |\pi| \) is finite, let \( \text{LAST}(\pi) \) be \( \pi(|\pi|) \). Finally,

---

\[1\] That is to say, a fixed set of predicates when using predicate abstraction [1].
let \( \pi' \) be the prefix of \( \pi \) such that \(|\pi'| = i\). We write \( \pi_i; \pi' \) 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 \(<S, s_i, T, L>\), where:
- \( S \) is a countable set of states;
- \( s_i \in S \) is an initial state;
- \( T \in S \rightarrow \text{PDS} \) is a transition function;
- \( L \in S \times AP \rightarrow \mathbb{B} \) is a labelling function.

The definition of \( \text{P} \) 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 \( \lambda \). A path of \( M \) is a sequence over \( S \cup \text{DS} \) that strictly alternates between states and distributions as described. Let \( \Pi_M \) and \( \Pi_M^\infty \) be the set of all finite and infinite paths, respectively. For \( \Omega \subseteq S \cup \text{DS} \) we write, e.g., \( \Pi_M(\Omega) \) to restrict to paths starting with an element in \( \Omega \).

A strategy of \( M \) is a partial function \( \sigma : \Pi_M \rightarrow \text{DDS}, \) with domain of definition all \( \pi \) 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 \( \Sigma_M \) be all strategies of \( M \). A path \( \pi \) 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 \( \sigma \). For every \( s \in S \) and \( \sigma \in \Sigma_M \) we construct a probability measure \( \Pr_{M,\sigma} \) over infinite paths \( \Pi_M^\infty\{s\} \) 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 \( a \in AP, k \in \mathbb{N} \cup \{\infty\}, p \in [0,1] \) and \( \sqsubset \in \{\leq, <, \geq, >\} \):

\[
\begin{align*}
\phi &::= a \mid \neg \phi \mid \phi_1 \vee \phi_2 \mid P_{\preceq \rho}(\phi) \\
\psi &::= X\phi \mid \phi_1 \sqsubset \phi_2.
\end{align*}
\]

We let \( \Phi \) and \( \Psi \) be the set of all PCTL formulae of the form \( \phi \) and \( \psi \) 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 \( \models \subseteq \Pi_M^\infty(S) \times \Psi \) as follows:

\[
\begin{align*}
\pi \models X\phi &\iff s_1 \models \phi \\
\pi \models \phi_1 \sqsubset \phi_2 &\iff \exists i \leq k : (s_i \models \phi_2 \land (\forall j < i : s_j \models \phi_1)) \\
s \models a &\iff L(s,a) \\
s \models \neg \phi &\iff s \not\models \phi \\
s \models \phi_1 \sqcup \phi_2 &\iff (s \models \phi_1 \text{ or } s \models \phi_2) \\
s \models P_{\preceq \rho}(\phi) &\iff \inf_{\epsilon} \Pr_{M,\epsilon}^s\{\pi \in \Pi_M^\infty(s) : \pi \models \phi \} > p \\
s \models P_{\succeq \rho}(\phi) &\iff \sup_{\epsilon} \Pr_{M,\epsilon}^s\{\pi \in \Pi_M^\infty(s) : \pi \models \phi \} < p
\end{align*}
\]

with \( \pi = s_0; \lambda_0; s_1; \lambda_1 \ldots, \rho \in \{>, \geq\}, \triangleleft \in \{<, \leq\} \) 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 \( P_{\infty} (\psi) \) are universal as threshold \( \propto p \) must be met under all strategies. Conversely, properties like \( \neg P_{\infty} (\psi) \) 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 \( R_D \subseteq \mathcal{D}X \times \mathcal{D}Y \) be the relation such that \( \lambda \in R_D \lambda_Y \) iff there is a weight function \( \delta \in X \times Y \rightarrow [0,1] \) with (for all \( x \in X, y \in Y \)):

\[
\lambda_X(x) = \sum_{y' \in Y} \delta(x, y') \quad \lambda_Y(y) = \sum_{x' \in X} \delta(x', y) \quad \delta(x, y) > 0 \Rightarrow x \in R y \quad (1)
\]

### 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 \( \text{emb} : \mathcal{M} \rightarrow \mathcal{G} \), which casts MDPs into \( \mathcal{G} \). We then define an abstraction preorder \( (\Sigma_p \subseteq \mathcal{G} \times \mathcal{G}) \) as a relation over games. Our last component is a four-valued PCTL semantics \((\vdash_1, \vdash_2 \subseteq \mathcal{G} \times \Phi)\) 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 \(<S,s_i,T,L_1,L_2>\), where:

- \( S \) is a countable set of states;
- \( s_i \in S \) is an initial state;
- \( T : S \rightarrow \mathcal{P}\mathcal{D}S \) is a transition function;
- \( L_1, L_2 \subseteq S \times \mathcal{A} \mathcal{P} \rightarrow \mathcal{B} \) are labelling functions.

By definition of \( \mathcal{P} \) 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 (P\( \mathcal{D} \)S) and probabilistic states as distributions over player 1 states (D\( \mathcal{S} \)). From a player 1 state \( s \in S \) player 1 can transition to a player 2 state \( \Lambda \in \mathcal{P}\mathcal{D}S \) iff \( \Lambda \in T(s) \) (written \( s \rightarrow_1 \Lambda \)). Analogously, from a player 2 state \( \Lambda \in \mathcal{P}\mathcal{D}S \) player 2 can transition to a probabilistic state \( \lambda \in \mathcal{D}S \) iff \( \lambda \in \Lambda \) (written \( \Lambda \rightarrow_2 \lambda \)). Finally, from a probabilistic state \( \Lambda \in \mathcal{D}S \) the game transitions to a player 1 state \( s' \in S \) with probability \( \lambda(s') \) (written \( \lambda \rightarrow_0 s' \)).

A play in \( G \) is a sequence of transitions\[4\] and hence necessarily strictly alternates between player 1 states, player 2 states and probabilistic states. Let \( \Pi_G \) and \( \Pi_G^\infty \) be the set of all finite and infinite plays of \( G \), respectively. For \( \Omega \subseteq S \cup \mathcal{P}\mathcal{D}S \cup \mathcal{D}S \) we write, e.g., \( \Pi_G(\Omega) \) to restrict to plays starting with an element in \( \Omega \).

A player 1 strategy is a partial function \( \sigma_1 \in \Pi_G \rightarrow \mathcal{D}\mathcal{P}D\mathcal{S} \), with domain of definition all \( \pi \) with \( \text{last}(\pi) \in S \), such that \( \sigma_1(\pi) \in \mathcal{D}(T(\text{last}(\pi))) \). Analogously, a player 2 strategy is a partial function \( \sigma_2 \in \Pi_G \rightarrow \mathcal{D}\mathcal{S} \), with domain of definition all \( \pi \) with \( \text{last}(\pi) \in \mathcal{P}\mathcal{D}S \), such that \( \sigma_2(\pi) \in \mathcal{D}(\text{last}(\pi)) \). We write \( \Sigma_G^1 \) and \( \Sigma_G^2 \) for the set of all player 1 and player

\[4\]We will manipulate plays as if they are sequences over \( S \cup \mathcal{P}\mathcal{D}S \cup \mathcal{D}S \).
2 strategies, respectively. A play $\pi$ 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, $\pi$ is consistent with $\sigma_2 \in \Sigma_G^2$ if $\sigma_2(\pi^i)(\pi(i + 1))$ is positive whenever $\pi(i) \in \Pi^D$.

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 $\omega \in S \cup \Pi^D \cup \Pi^S$, using standard techniques [23], we construct a probability space over infinite plays $\Pi^S_{G,\sigma_1,\sigma_2}(\{\omega\})$ with probability measure $\Pr_{G,\sigma_1,\sigma_2}^\omega(\{\omega\}) = 1$ and, for every finite play of non-zero length $\pi \in \Pi_{G,\sigma_1,\sigma_2}(\{\omega\})$:

$$\Pr_{G,\sigma_1,\sigma_2}^\omega(\{\pi\}) = \begin{cases} \Pr_{G,\sigma_1,\sigma_2}^\omega(\{\pi'\}) \cdot \sigma_1(\pi')(\Lambda') & \text{if } (\pi = \pi' \rightarrow_1 \Lambda') \\ \Pr_{G,\sigma_1,\sigma_2}^\omega(\{\pi'\}) \cdot \sigma_2(\pi')(\Lambda') & \text{if } (\pi = \pi' \rightarrow_2 \Lambda') \\ \Pr_{G,\sigma_1,\sigma_2}^\omega(\{\pi^i\}) \cdot \text{LAST}(\pi')(s') & \text{if } (\pi = \pi' \rightarrow_p s') \end{cases}$$

**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 space over infinite plays $\Pi^S_{G,\sigma_1,\sigma_2}(\{\omega\})$.

**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 $\hat{G}$ must be consistent with both $\hat{S}$ and $\hat{D}$. Hence, for every finite play of non-zero length $\pi \in \Pi_{\hat{G},\hat{S},\hat{D}}(\{\omega\})$:

$$\Pr_{\hat{G},\hat{S},\hat{D}}(\{\pi\}) = \begin{cases} \Pr_{\hat{G},\hat{S},\hat{D}}(\{\pi'\}) \cdot \sigma_1(\pi')(\Lambda') & \text{if } (\pi = \pi' \rightarrow_1 \Lambda') \\ \Pr_{\hat{G},\hat{S},\hat{D}}(\{\pi'\}) \cdot \sigma_2(\pi')(\Lambda') & \text{if } (\pi = \pi' \rightarrow_2 \Lambda') \\ \Pr_{\hat{G},\hat{S},\hat{D}}(\{\pi^i\}) \cdot \text{LAST}(\pi')(s') & \text{if } (\pi = \pi' \rightarrow_p s') \end{cases}$$

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

**Definition 6.** Let $\text{emb} : M \rightarrow G$ be the function which for every MDP $M = \langle S, s_0, T, L \rangle$ yields a game $G = \langle S, s_0, \hat{T}, L, L \rangle$ such that $\hat{T}(s) = \{ T(s) \}$ for every $s \in S$.

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

\footnote{Below, we formalise this under/over-approximation in the definition of the preorder $\preceq_P$.}
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 \( \text{emb}(M) \) 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 to 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 \( \lambda \in \mathbb{DS} \) 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 geometrically as points on a plane in \(|S|\)-dimensional Euclidean space.

In the player 2 state \( \Lambda_\alpha \in \mathbb{PD} \) player 2 can make a probabilistic choice over probabilistic states \( \{\lambda_1, \lambda_2\} \). Hence, it is appropriate to think of \( \Lambda \) 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 \( \Lambda \in \mathbb{PD} \) to a probabilistic state \( \lambda \in \mathbb{DS} \), denoted \( \Lambda \transformation{\lambda} \), iff for some \( \sum_i w_i \cdot |\lambda_i| \in \mathbb{DS} \) 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 \( \Lambda \in \mathbb{PD} \), denoted \( s \transformation{\Lambda} \), 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 \( \sqsubseteq_p \subseteq \hat{G} \times \hat{G} \) over games. Intuitively, this preorder defines a notion of precision in \( \hat{G} \); 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 \( \hat{G} \sqsubseteq_p \text{emb}(M) \)).

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):
Figure 2: Games illustrating various points in the paper.

**Definition 8.** Let \( G = \langle S, s_i, T, L^1, L^2 \rangle \) 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) \( L^1(s', a) \Rightarrow L^1(s, a) \) for all \( a \in AP \)
(ii) \( L^2(s', a) \Leftarrow L^2(s, a) \) for all \( a \in AP \)
(iii) \( \forall s \rightarrow_1 \Lambda : \exists s' \rightarrow_c \Lambda' : \forall \Lambda' \rightarrow_2 \lambda' : \exists \lambda \rightarrow_c \lambda : \Lambda \rightarrow_D \lambda \)
(iv) \( \forall s \rightarrow_1 \Lambda : \exists s' \rightarrow_c \Lambda' : \forall \Lambda \rightarrow_2 \lambda' : \exists \lambda' \rightarrow_c \lambda' : \Lambda' \rightarrow_D \lambda' \)

Moreover, for games \( \hat{G} = \langle \hat{s}, \hat{s_i}, \hat{T}, \hat{L}^1, \hat{L}^2 \rangle \) and \( G = \langle S, s_i, T, L^1, L^2 \rangle \) we let \( \hat{G} \subseteq_p G \) iff the largest\(^\dagger\) strong probabilistic game-simulation \( R \) on \( \hat{G} \oplus G \) includes \( \hat{s}, \hat{s_i} \).

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. \( \hat{\Lambda} \in \text{PDS} \) under-approximates \( \Lambda \in \text{PDS} \) iff all transitions \( \hat{\Lambda} \rightarrow_2 \hat{\lambda} \) can be simulated by a combined player 2 transition \( \Lambda \rightarrow_2 \lambda \). That is, for these player 2 states, player 2 in \( \hat{G} \) is more powerful than player 2 in \( G \).

Recall that player 1 transitions \( s \rightarrow_1 \Lambda \) are such that \( \Lambda \) 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 \( \hat{s} \rightarrow_c \hat{\Lambda} \). This under/over-approximation is realised by the outermost quantifier pair of (iii) and (iv), respectively.

**Example 9.** Consider games \( \hat{G} \) and \( \text{emb}(M) \) in Fig. 2(a). The largest strong prob. game-simulation \( R \) over \( \hat{G} \oplus \text{emb}(M) \) 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 \( \hat{s}_0 R \hat{s}_0 \), i.e. \( \hat{G} \subseteq_p \text{emb}(M) \), observe that (iii) for \( \Lambda_1 \in T(s_0) \) is satisfied by \( \hat{\Lambda}_2 \in \hat{T}(\hat{s}_0) \) as \( \hat{\lambda}_2 \rightarrow_D \hat{\lambda}_1 \) and (iv) is satisfied by \( \hat{\Lambda}_1 \in \hat{T}(\hat{s}_0) \) as \( \lambda_1 \rightarrow_D \lambda_1 + \frac{1}{2} \cdot \hat{\lambda}_2 \rightarrow_D \lambda_2 \).

**Generality of \( \subseteq_p \)** 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 \( \hat{s} \rightarrow_1 \Lambda \). 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.

\(^\dagger\)The largest strong game-simulation in a game is the union of all its strong game-simulations.
Abstract PCTL semantics \( (\models^1, \models^?) \) The final component of our abstraction framework is a four-valued abstract PCTL semantics \( (\models^1, \models?) \subseteq G \times \Phi \) for games. Informally, \( G \models^1 \phi \) (resp. \( G \models^? \phi \)) holds only if all MDPs that \( G \) abstracts must (resp. may) satisfy \( \phi \).

**Definition 10.** Let \( G = \langle S, s_i, T, L_i, L_1 \rangle \) be a game and let \( \phi \in \Phi \) be a PCTL formula. We define must/may relations for states \( (\models^1, \models?) \subseteq S \times \Phi \) and plays \( (\models^1, \models?) \subseteq \Pi^\omega_G (S) \times \Psi \) as follows (letting \( * \in \{!, ?\}, \neg = ? \) and \( \neg = ! \)):

\[
\begin{align*}
\pi \models^* \chi \phi &\iff s_1 \models^* \phi \\
\pi \models^* \phi_1 U^? \phi_2 &\iff \exists i \leq k : (s_i \models^* \phi_2 \land (\forall j < i : s_j \models^* \phi_1)) \\
s \models^a &\iff L^*(s,a) \\
s \models^\neg \phi &\iff s \models^{\neg*} \phi \\
s \models^1 \phi_1 \lor \phi_2 &\iff (s \models^1 \phi_1 \lor s \models^* \phi_2) \\
s \models^1 P_{\sigma_i}(\psi) &\iff \inf_{c_1} \inf_{c_2} \Pr_{G,\sigma_1,\sigma_2}^c \{ \pi \in \Pi^\omega_G (s) \mid \pi \models^1 \psi \} > p \\
s \models^? P_{\sigma_i}(\psi) &\iff \sup_{c_1} \inf_{c_2} \Pr_{G,\sigma_1,\sigma_2}^c \{ \pi \in \Pi^\omega_G (s) \mid \pi \models^? \psi \} < p \\
s \models^1 P_{\sigma_i}(\psi) &\iff \sup_{c_1} \sup_{c_2} \Pr_{G,\sigma_1,\sigma_2}^c \{ \pi \in \Pi^\omega_G (s) \mid \pi \models^1 \psi \} \leq p \\
s \models^? P_{\sigma_i}(\psi) &\iff \inf_{c_1} \sup_{c_2} \Pr_{G,\sigma_1,\sigma_2}^c \{ \pi \in \Pi^\omega_G (s) \mid \pi \models^? \psi \} \geq p \\

\end{align*}
\]

with \( \pi = s_0; \Lambda_0; s_1; \ldots \) \( \sigma_i \in \Sigma_G \). 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 \( P_{\sigma_i}(\psi) \). Recall \( P_{\sigma_i}(\psi) \) holds for an MDP if all MDP-schedulings meet the threshold \( \gg p \). As player 2 represents MDP non-determinism, for a lower threshold \( > p \) (upper threshold \( < p \)) 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 \( > p \), sup. for \( < p \)) and in the may modality we quantify optimistically over player 1 strategies (sup. for \( > p \), inf. for \( < p \)). 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 \( < p \).

**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 \( M \models \phi \iff \text{emb}(M) \models^1 \phi \iff \text{emb}(M) \models^? \phi \).

**Proof.** Follows directly from the fact that embedded MDPs have two-valued propositional labelling, i.e. \( L_i^1 = L_i^2 \), and one trivial player 1 strategy, i.e. \( |\Sigma^1_G| = 1 \).
**Theorem 12.** For all games $\hat{G}, G$ such that $\hat{G} \sqsubseteq_p G$ and all PCTL properties $\phi \in \Phi$ we have $(\hat{G} \models ^1 \phi \Rightarrow G \models ^1 \phi)$ and $(\hat{G} \not\models ^? \phi \Rightarrow G \not\models ^? \phi)$.

**Proof.** We only sketch the structure of the proof here. Let $s$ be any state of $\hat{G}$ and let $s$ be any state of $G$, respectively. The proof shows that $s \models ^1 \phi \Rightarrow s \models ^1 \phi$ and, dually, that $s \models ^? \phi \Rightarrow s \models ^? \phi$. The main complexity of the proof is due to properties $P_{\gg P}(\psi)$.

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 $s \models ^? P_{\gg P}(\psi) \Rightarrow s \models ^? P_{\gg P}(\psi)$. 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 $s \models ^1 P_{\gg P}(\psi) \Rightarrow s \models ^1 P_{\gg P}(\psi)$.

Lem. 11 ensures consistency across the two representations of MDPs whereas Th. 12 ensures that any property that is $\models ^1$-satisfied (not $\models ^?$-satisfied) by a game $\hat{G}$ is also $\models ^1$-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. $G \sqsubseteq_p emb(M)$) and that $!$-satisfies $\phi$ (i.e. $G \models ^! \phi$). It is easy to see this: by Th. 12, $emb(M) \models ^! \phi$ and by Lem. 11 this is equivalent to $M \models \phi$. Analogously, we can refute $\bar{M} \models \phi$ by finding a game $G$ such that $G \sqsubseteq_p emb(M)$ and $G \not\models ^? \phi$ (i.e. by Th. 12, $emb(M) \not\models ^? \phi$; by Lem. 11, $M \not\models \phi$).

For some games $G$ it may be that both $G \not\models ^1 \phi$ and $G \not\models ^? \phi$. 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 $emb(M)$ of Fig. 2(a), as $\hat{G} \sqsubseteq_p emb(M)$ and $\hat{G} \models ^! \neg P_{>0.5}(Xa)$, using Lem. 11 and Th. 12 we have verified $M \models \neg P_{>0.5}(Xa)$: some scheduling of $M$ satisfies $Xa$ with probability of at most 0.5. However, as $\hat{G} \not\models ^1 P_{\leq 0.5}(Xa)$ and $\hat{G} \models ^? P_{\leq 0.5}(Xa)$ we cannot verify nor refute via $\hat{G}$ whether this threshold holds for all scheduleings 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 $G \sqsubseteq_p emb(M)$ and $G \models ^1 \phi$. Analogously, we can use games as refutation certificates. We demonstrate our framework with a final motivating example: 

![Figure 3: The MDP (a) and the two game abstractions (b) and (c) from Example 14.](image-url)
Example 14. Reconsider the program and embedded MDP $\text{emb}(M)$ of Example 2. Suppose we wish to check the program stops with $x \leq y$ with a probability greater than 0.003 (i.e. $M \models P > 0.003(\text{true} \cup^\infty x \leq y)$). 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 $\models_p^!$, $\models_p^?$ 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 $\models_p^!$, $\models_p^?$ 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 \(P \leq p \langle \psi \rangle\), 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


This work is licensed under the Creative Commons Attribution-NonCommercial-No Derivative Works 3.0 License.