Verification and Refutation of Probabilistic Specifications via Games

Mark Kattenbelt*  
Michael Huth†

*Oxford University  
†Imperial College London

December 8, 2009
Games as abstractions [KNP06]

Currently:

- game abstractions and abstraction refinement have yielded good results in verifying probabilistic systems such as: PRISM models [KKNP08], ANSI-C programs [KKNP09] and PTA [KNP09]
- for a fixed model & predicates [KNP09] only considers one game abstraction, which is typically expensive to construct/analyse

This paper:

- we develop a more fine-grained notion of abstraction for games via an abstraction relation over games
- this enables us to consider a “hierarchy of abstractions” with varying precisions (even for fixed predicates!)
- we develop a verification/refutation framework for arbitrary PCTL specifications (c.f. modal abstractions [Lar90])
Games as abstractions (cont’d)

Currently:

- identifies one game abstraction

This paper:

- identifies many game abstractions
Overview  (definitions follow later!)

Concrete:

- $M$ are models  \hspace{1cm} (Markov Decision Processes)
- $P$ are specifications \hspace{1cm} (Probabilistic CTL, [HJ94])
- $|=\ $ is a satisfaction relation \hspace{1cm} ([BdA95])

Abstract:

- $G$ are abstract models \hspace{1cm} (Games, [KNP06])
- $emb : M \rightarrow G$ is an embedding function \hspace{1cm} (this paper)
- $\sqsubseteq_p \subseteq G \times G$ is an abstraction relation \hspace{1cm} (this paper)
- $|=\text{may},|=\text{must} \subseteq G \times P$ are PCTL semantics \hspace{1cm} (this paper)

General idea:

- for $m \in M$ and $p \in P$, \text{verify} or \text{refute} $m |= p$ via model checks on games $g \in G$ using properties of $emb$, $\sqsubseteq_p$ and $|=\text{may},|=\text{must}$
Overview (definitions follow later!)

Intuition of $\text{emb}$, $\sqsubseteq_p$:

- $\text{emb}(m)$ is an exact representation of an MDP $m$ in $G$
- $g \sqsubseteq_p g$ means $g$ abstracts $g$
- $g \sqsubseteq_p \text{emb}(m)$ means the game $g$ abstracts the MDP $\text{emb}(m)$
Overview (definitions follow later!)

Intuition of $\text{emb}$, $\sqsubseteq_p$:

- $\text{emb}(m)$ is an exact representation of an MDP $m$ in $G$
- $g \sqsubseteq_p g$ means $g$ abstracts $g$
- $g \sqsubseteq_p \text{emb}(m)$ means the game $g$ abstracts the MDP $\text{emb}(m)$
Overview (definitions follow later!)

Intuition of $\text{emb, } \sqsubseteq_p$:

- $\text{emb}(m)$ is an exact representation of an MDP $m$ in $G$
- $g \sqsubseteq_p g$ means $g$ abstracts $g$
- $g \sqsubseteq_p \text{emb}(m)$ means the game $g$ abstracts the MDP $\text{emb}(m)$
Overview (still no definitions!)

PCTL evaluations on games:

- \( g \models^{\text{may}} p \) means concretisations of \( g \) possibly satisfies \( p \)
- \( g \models^{\text{must}} p \) means concretisations of \( g \) definitely satisfies \( p \)

Soundness requirements:

- if \( g \models^{\text{may}} p \) then all abstractions of \( g \) also may-satisfy \( p \)
- if \( g \models^{\text{must}} p \) then all concretisations of \( g \) also must-satisfy \( p \)

Verification & refutation via games:

- to verify \( m \models p \) find a game \( g \in G \) such that \( g \) abstracts \( m \) (i.e. \( g \sqsubseteq_p \text{emb}(m) \)) and \( g \) must-satisfies \( p \) (i.e. \( g \models^{\text{must}} p \))
- to refute \( m \models p \) find a game \( g \in G \) such that \( g \) abstracts \( m \) (i.e. \( g \sqsubseteq_p \text{emb}(m) \)) and \( g \) does not may-satisfy \( p \) (i.e. \( g \not\models^{\text{may}} p \))
Overview

Concrete:

- M are models (Markov Decision Processes)
- P are specifications (Probabilistic CTL, [HJ94])
- $\models$ is a satisfaction relation ([BdA95])

Abstract:

- G are abstract models (Games, [KNP06])
- $emb : M \rightarrow G$ is an embedding function (this paper)
- $\sqsubseteq_p \subseteq G \times G$ is an abstraction relation (this paper)
- $\models^{\text{may}}, \models^{\text{must}} \subseteq G \times P$ are PCTL semantics (this paper)
Models & specifications

Markov decision processes

- non-deterministic choice
- probabilistic choice
- path
- strategy
Models & specifications

Markov decision processes

- non-deterministic choice
- probabilistic choice
- path
- strategy
Models & specifications

Markov decision processes

- non-deterministic choice
- probabilistic choice

- path
- strategy
Models & specifications

Markov decision processes

- non-deterministic choice
- probabilistic choice
- path
- strategy
Models & specifications

Markov decision processes

- non-deterministic choice
- probabilistic choice

- path
- strategy
Models & specifications

Markov decision processes

- non-deterministic choice
- probabilistic choice
- path
- strategy

PCTL formulas

- for all strategies the probability of reaching $s_3$ is $\leq 0.5$
- for some strategy the probability of reaching $s_3$ is $> 0.2$
Overview

Concrete:
- $M$ are models (Markov Decision Processes)
- $P$ are specifications (Probabilistic CTL, [HJ94])
- $\models$ is a satisfaction relation ([BdA95])

Abstract:
- $G$ are abstract models (Games, [KNP06])
- $\text{emb} : M \to G$ is an embedding function (this paper)
- $\sqsubseteq_p \subseteq G \times G$ is an abstraction relation (this paper)
- $\models^\text{may}, \models^\text{must} \subseteq G \times P$ are PCTL semantics (this paper)
Abstraction: stochastic games \((G)\)

- player 1 choice
- player 2 choice
- probabilistic choice
- play
- player 1 strategy
- player 2 strategy
Abstraction: stochastic games (G)

- player 1 choice
- player 2 choice
- probabilistic choice

- play
- player 1 strategy
- player 2 strategy
Abstraction: stochastic games (G)

- player 1 choice
- player 2 choice
- probabilistic choice

- play
- player 1 strategy
- player 2 strategy
Abstraction: stochastic games \((G)\)

- player 1 choice
- player 2 choice
- probabilistic choice

\[s_1 \xrightarrow{1} s_4 \quad \xrightarrow{1/2} s_5\]
\[s_4 \xrightarrow{1/2} s_5\]
\[s_2 \xrightarrow{6/10} s_3\]
\[s_3 \xrightarrow{4/10} s_4\]

- play
- player 1 strategy
- player 2 strategy
Abstraction: stochastic games (G)

- player 1 choice
- player 2 choice
- probabilistic choice

- play
- player 1 strategy
- player 2 strategy
Abstraction: stochastic games \((G)\)

- player 1 choice
- player 2 choice
- probabilistic choice

- play
- player 1 strategy
- player 2 strategy
Abstraction: stochastic games (G)

- player 1 choice
- player 2 choice
- probabilistic choice

- play
- player 1 strategy
- player 2 strategy
Games as abstractions of MDPs

- player 1 “picks a concretisation”
- player 2 controls the non-determinism in the MDPs
Games as abstractions of MDPs

- player 1 “picks a concretisation”
- player 2 controls the non-determinism in the MDPs
Overview

Concrete:

- M are models (Markov Decision Processes)
- P are specifications (Probabilistic CTL, [HJ94])
- |= is a satisfaction relation ([BdA95])

Abstract:

- G are abstract models (Games, [KNP06])
- emb : M → G is an embedding function (this paper)
- ⊑p ⊆ G × G is an abstraction relation (this paper)
- |=^may, |=^must ⊆ G × P are PCTL semantics (this paper)
Embedding function $\text{emb} : M \rightarrow G$

- i.e. add a trivial player 1 transitions everywhere
Overview

Concrete:

- M are models (Markov Decision Processes)
- P are specifications (Probabilistic CTL, [HJ94])
- |= is a satisfaction relation (BdA95)

Abstract:

- G are abstract models (Games, [KNP06])
- emb : M \to G is an embedding function (this paper)
- \sqsubseteq_p \subseteq G \times G is an abstraction relation (this paper)
- \models_{\text{may}}, \models_{\text{must}} \subseteq G \times P are PCTL semantics (this paper)
Abstraction relation $\sqsubseteq$

“Abstract” game

“Concrete” game

\[ \frac{1}{2} \quad \frac{1}{2} \quad \frac{1}{2} \quad \frac{1}{2} \quad \frac{1}{2} \]

iff

- player 1 in $\bigcirc$ can over-approximate player 1 in $\bullet$:
  \[
  \forall \quad \exists \quad \forall \quad \exists \quad \sqsubseteq
  \]

- player 1 in $\bigcirc$ can under-approximate player 1 in $\bullet$:
  \[
  \forall \quad \exists \quad \forall \quad \exists \quad \sqsubseteq
  \]

(here $\sqsubseteq$ is lifted to distributions with standard methods [JL91])
Abstraction relation $\sqsubseteq$

\[ a_{1/2} \sqsubseteq c_{1/2} \iff \forall c : \exists a : \forall c : \exists a : a \sqsubseteq c \quad \text{(here } \sqsubseteq \text{ is lifted to distributions with standard methods [JL91])} \]
Abstraction relation \( \sqsubseteq \)

\[ \begin{align*}
\text{“Abstract” game} & \quad \sqsubseteq \\
\text{“Concrete” game} \\
\end{align*} \]

iff

- player 1 in \( \bullet \) can **over-approximate** player 1 in \( \bigcirc \):
  \[ \forall \; \exists : \forall \; \exists \; \subseteq \]

- player 1 in \( \bullet \) can **under-approximate** player 1 in \( \bigcirc \):
  \[ \forall \; \exists : \forall \; \exists \; \subseteq \]

(here \( \sqsubseteq \) is lifted to distributions with standard methods [JL91])
Combined transitions (c.f. [Seg95])
Combined transitions (c.f. [Seg95])
Combined transitions (c.f. [Seg95])
We now have $\bigtriangledown_p$ iff

- $\forall \bigtriangledown \exists C : \bigtriangledown C \bigtriangledown : \exists C : \bigtriangledown C\bigtriangledown$
Overview

Concrete:

- M are models \((\text{Markov Decision Processes})\)
- P are specifications \((\text{Probabilistic CTL, [HJ94]})\)
- \(\models\) is a satisfaction relation \((\text{[BdA95]})\)

Abstract:

- G are abstract models \((\text{Games, [KNP06]})\)
- \(\text{emb}: M \to G\) is an embedding function \((\text{this paper})\)
- \(\sqsubseteq_p \subseteq G \times G\) is an abstraction relation \((\text{this paper})\)
- \(\models^{\text{may}}, \models^{\text{must}} \subseteq G \times P\) are PCTL semantics \((\text{this paper})\)
PCTL semantics $\models^\text{may}$, $\not\models^\text{must}$ for games

Player 1 & 2 revisited:
- player 1 strategies quantify over concretisations
- player 2 strategies correspond to MDP-strategies

Informal PCTL semantics:

<table>
<thead>
<tr>
<th>Specification</th>
<th>must-satisfied</th>
<th>may-satisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>for all strategies ...</td>
<td>for all player 1 strategies and for all player 2 strategies ...</td>
<td>for some player 1 strategies and for all player 2 strategies ...</td>
</tr>
<tr>
<td>for some strategy ...</td>
<td>for all player 1 strategies and for some player 2 strategy ...</td>
<td>for some player 1 strategies and for some player 2 strategy ...</td>
</tr>
</tbody>
</table>
Games as abstractions (revisited)

Currently:

- identifies one game abstraction

This paper:

- identifies many game abstractions
Finding better games (examples)

Without losing precision:

- remove non-extreme player 2 transitions

- remove non-extreme player 1 transitions
Finding better games (cont’d)

- a smaller abstraction is obtained by removing non-extreme transitions

- both abstractions are equivalent!
Finding better games (examples)

By losing precision:

- over-approximate probabilistic choice

- use most abstract game over a partition
Finding better games (cont’d)

- a smaller abstraction is obtained by replacing probabilistic choice with player 1 non-determinism

- the left-hand game is more abstract (but still “good enough”)

\[
\begin{align*}
&\subseteq p \\
&\not\supseteq p
\end{align*}
\]
Summary

Summary:

- identified the need for a more fine-grained abstraction for games
- developed an abstraction framework \((\text{emb}, \sqsubseteq_p \text{ and } \models \text{may}, \models \text{must})\)
- enabled, via \(\sqsubseteq_p\), a whole hierarchy of game abstractions with varying precisions (even for fixed predicates)
- developed a notion of combined player 1 and player 2 transitions for games
- developed 4-valued semantics for arbitrary pctl specifications over games

Future work

- automatically find good abstractions
Questions
Questions

- how can you find good games?
- what was the running example?
Abstraction refinement with games

- the machinery developed in this paper facilitates the inner refinement loop
void main()
{
    uchar x, y, z;
    x = nondet();
    y = x*x;
    z = prob();

    if (y < z && z < x)
    {
        TARGET;
    }
}

- does main satisfy $P \leq 0.98 \langle F \text{TARGET} \rangle$? (hint: take $x=255$)
Running example (cont’d)

refutation via \( \text{emb}, \models_p \) and \( \models^\text{may} \)

- the abstraction does not \( \text{may} \)-satisfy \( P \leq 0.98 \langle F \text{ TARGET} \rangle \)


M. Kattenbelt, M. Kwiatkowska, G. Norman, and David Parker. Abstraction refinement for probabilistic programs.


R. Segala. 