UNIVERSITY OF OXFORD

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:

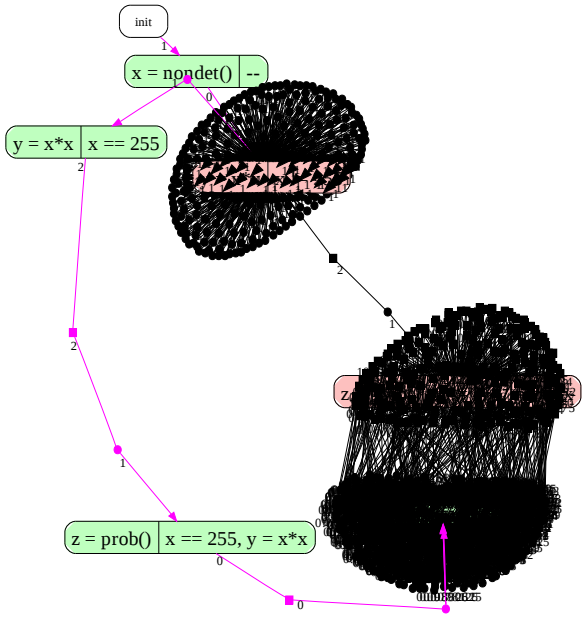
- game abstractions and abstraction refinement have yielded good results in verifying probabilistic systems such as: PRISM models [KKNP08], ANSI-C programs [KKNP09] and PTA [KNP09]
- for a fixed model & predicates [KNP09] only considers one game abstraction, which is typically expensive to construct/analyse

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])

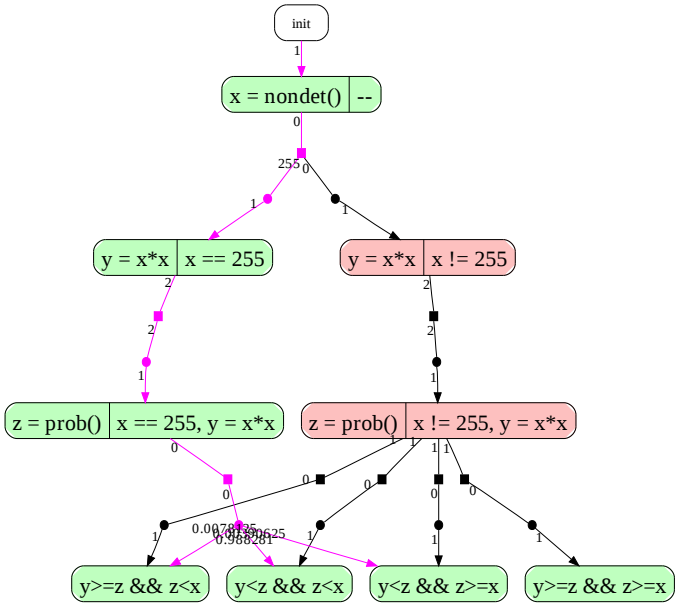
Games as abstractions (cont'd)

Currently:



– identifies one game abstraction

This paper:



– identifies many game abstractions

Overview (definitions follow later!)

Concrete:

- M are models (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*)

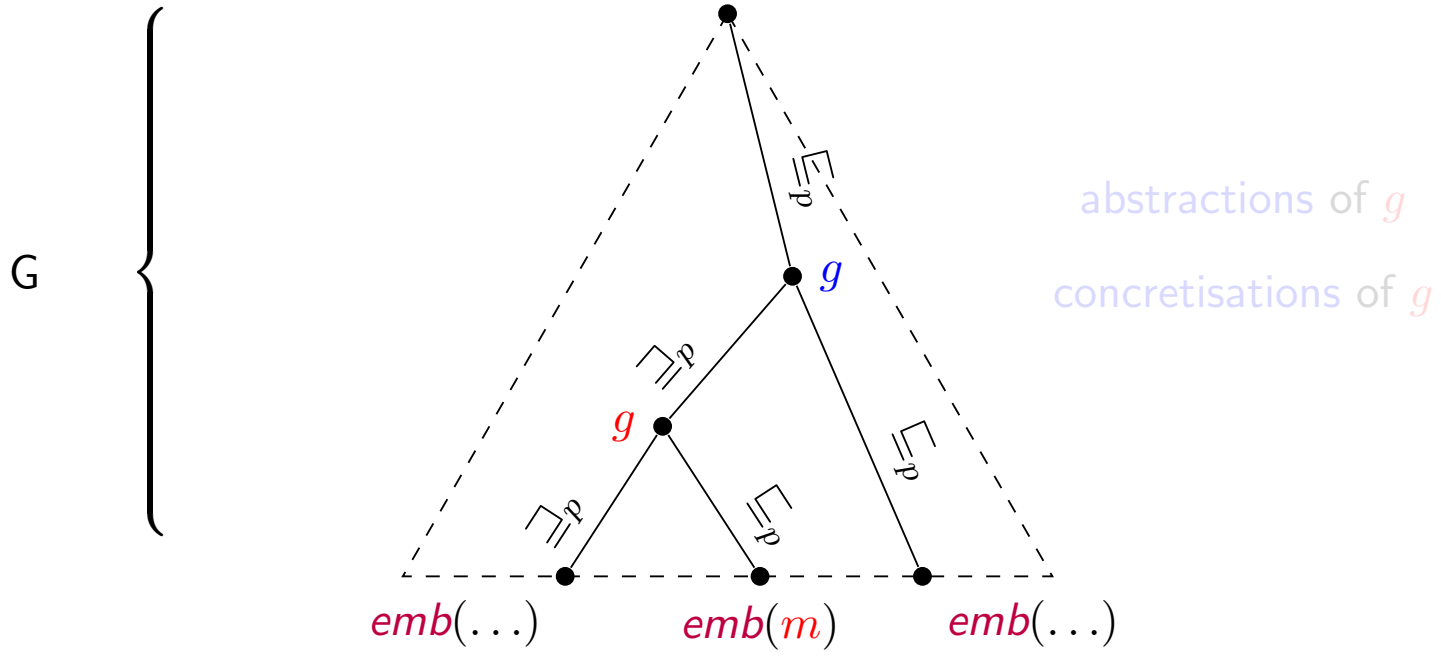
General idea:

- for $m \in$ M and $p \in$ P, verify or refute $m \models p$ via model checks on games $g \in$ G using properties of *emb*, $\sqsubseteq_p$ and $\models^{\text{may}}, \models^{\text{must}}$

Overview (definitions follow later!)

Intuition of emb, $\sqsubseteq_p$:

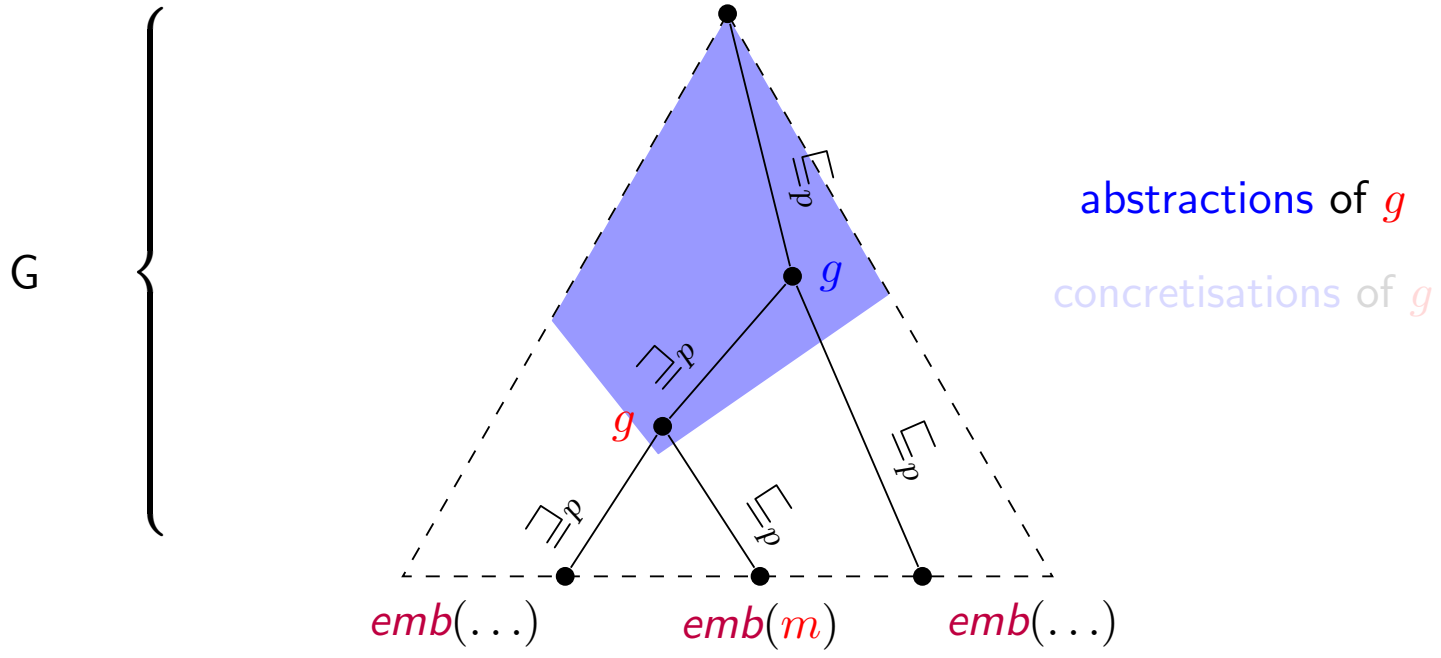
- $emb(m)$ is an exact representation of an MDP m in G
- $g \sqsubseteq_p g$ means g abstracts g
- $g \sqsubseteq_p emb(m)$ means the game g abstracts the MDP $emb(m)$



Overview (definitions follow later!)

Intuition of emb, $\sqsubseteq_p$:

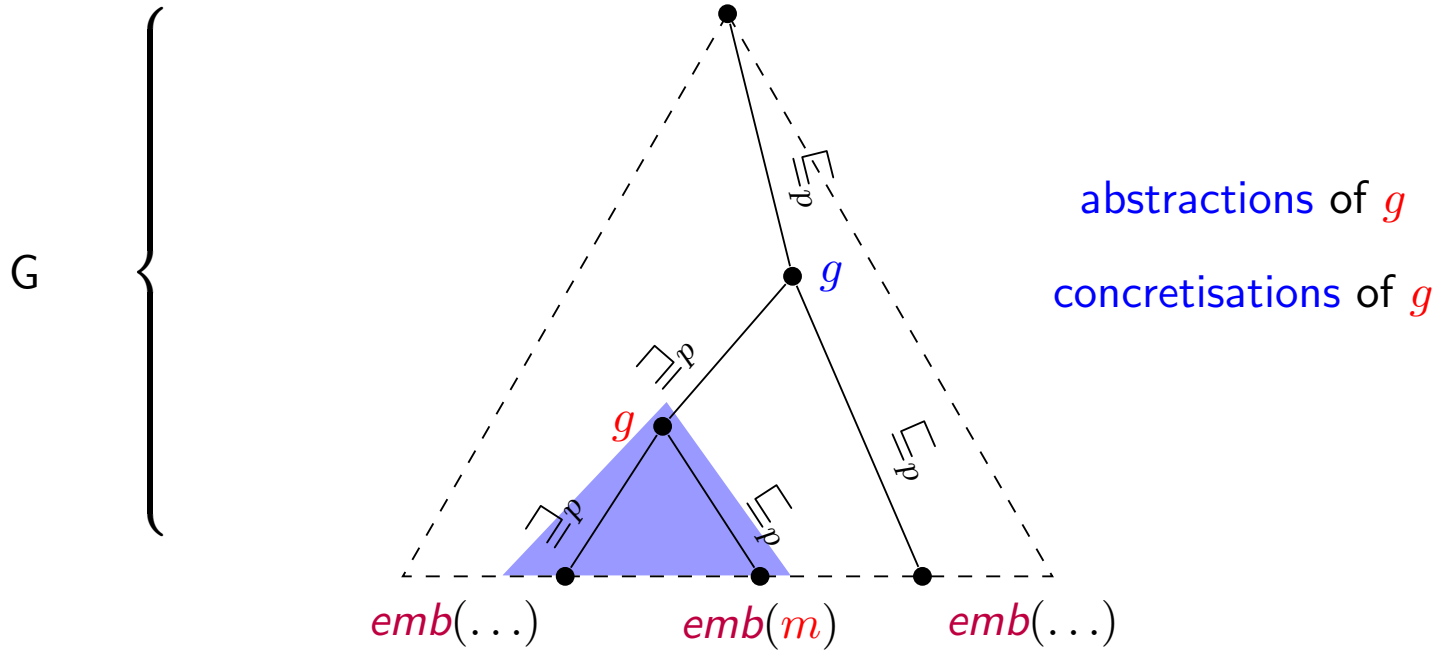
- $emb(m)$ is an exact representation of an MDP m in G
- $g \sqsubseteq_p g$ means g abstracts g
- $g \sqsubseteq_p emb(m)$ means the game g abstracts the MDP $emb(m)$



Overview (definitions follow later!)

Intuition of emb, $\sqsubseteq_p$:

- $emb(m)$ is an exact representation of an MDP m in G
- $g \sqsubseteq_p g$ means g abstracts g
- $g \sqsubseteq_p emb(m)$ means the game g abstracts the MDP $emb(m)$



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

Overview

Concrete:

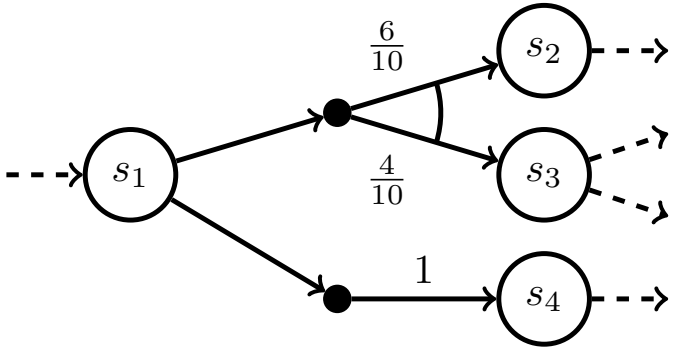
- M are models (Markov Decision Processes)
- P are specifications (Probabilistic CTL, [HJ94])
- $\models$ is a satisfaction relation ([BdA95])

Abstract:

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

Models & specifications

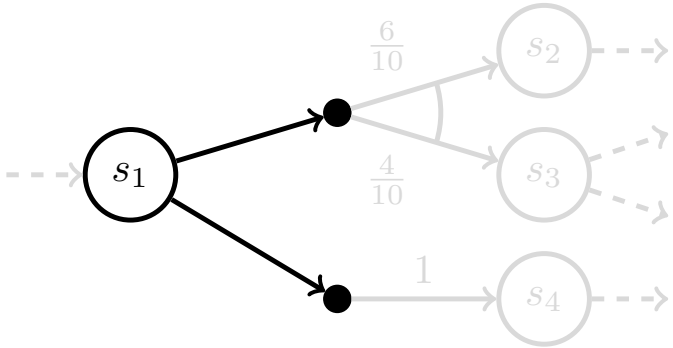
Markov decision processes



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

Models & specifications

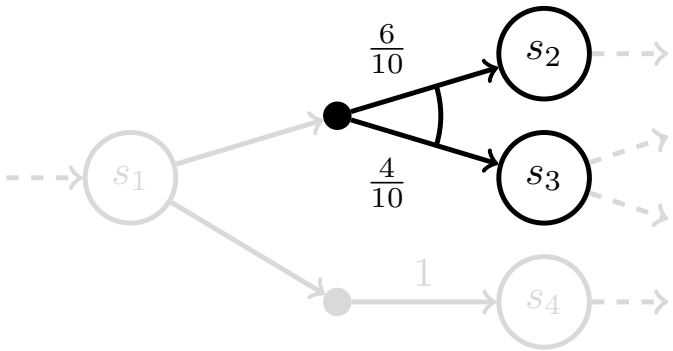
Markov decision processes



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

Models & specifications

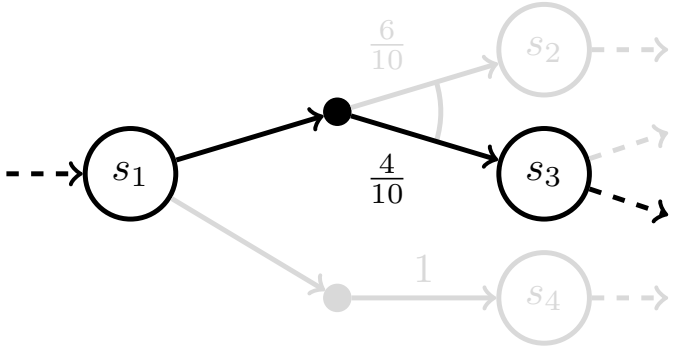
Markov decision processes



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

Models & specifications

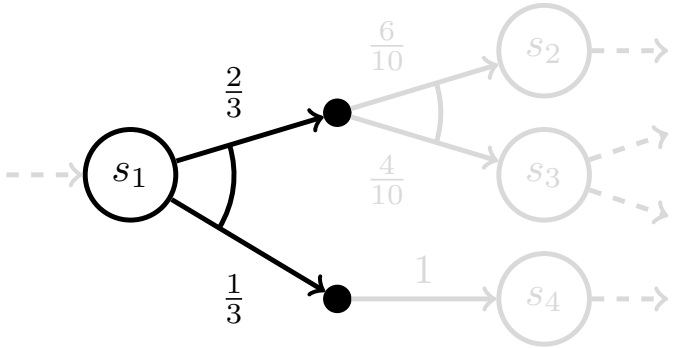
Markov decision processes



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

Models & specifications

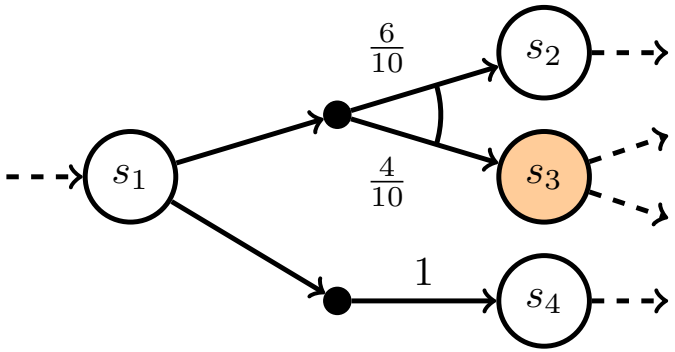
Markov decision processes



- 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 ≤ 0.5
- for some strategy the probability of reaching s_3 is > 0.2

Overview

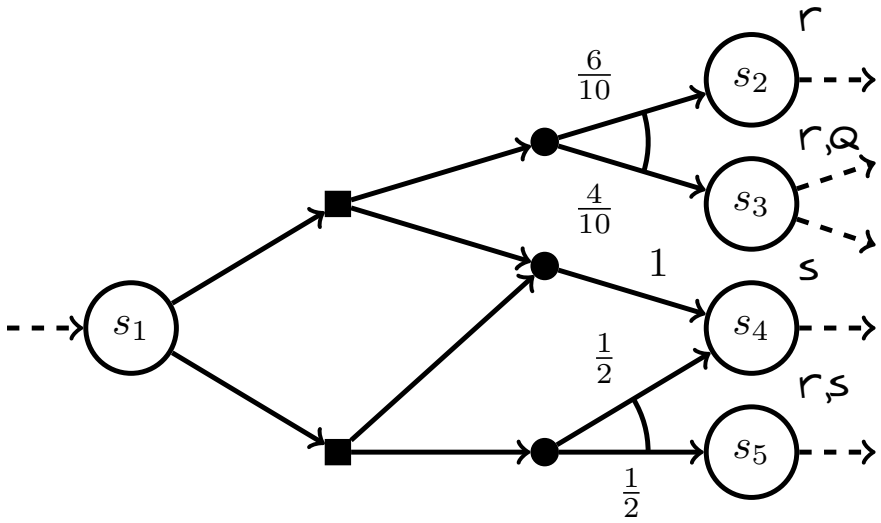
Concrete:

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

Abstract:

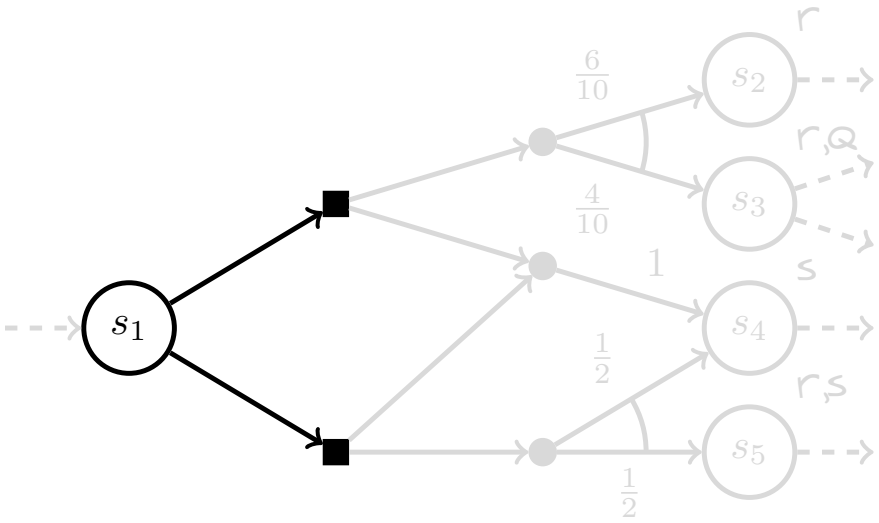
- G are abstract models (Games, [KNP06])
- *emb* : M $\rightarrow$ G is an embedding function (*this paper*)
- $\sqsubseteq_p \subseteq$ G $\times$ G is an abstraction relation (*this paper*)
- $\models^{\text{may}}, \models^{\text{must}} \subseteq$ G $\times$ P are PCTL semantics (*this paper*)

Abstraction: stochastic games (G)



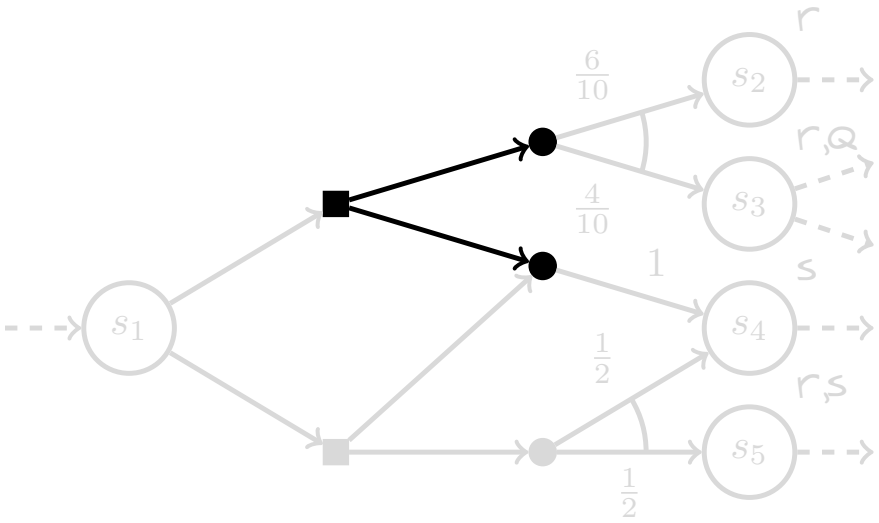
- player 1 choice
- player 2 choice
- probabilistic choice
- play
- player 1 strategy
- player 2 strategy

Abstraction: stochastic games (G)



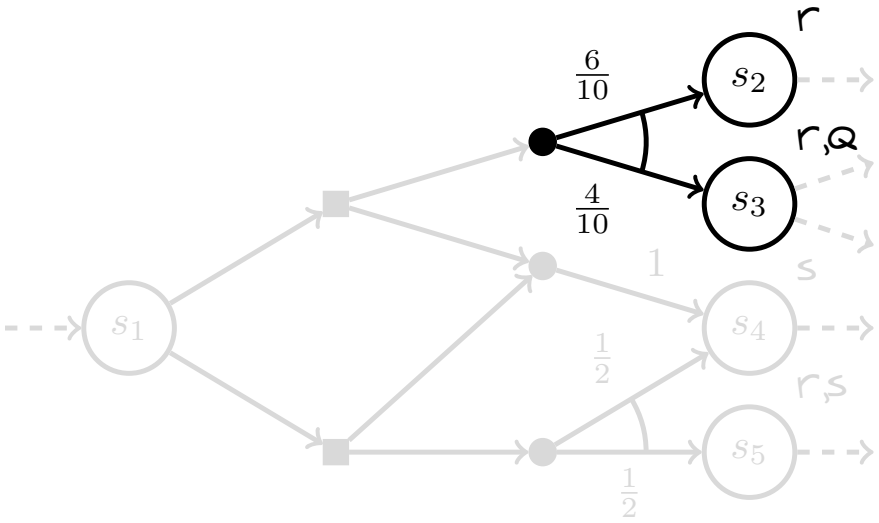
- player 1 choice
- player 2 choice
- probabilistic choice
- play
- player 1 strategy
- player 2 strategy

Abstraction: stochastic games (G)



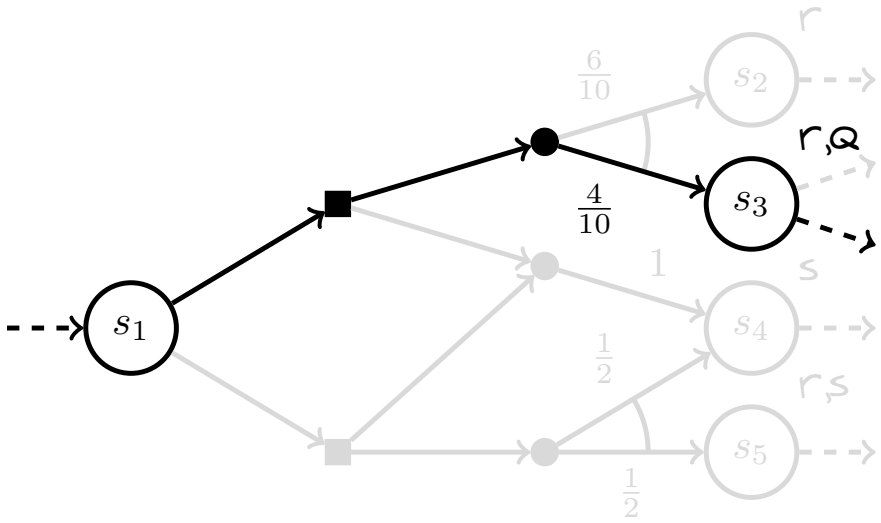
- player 1 choice
- player 2 choice
- probabilistic choice
- play
- player 1 strategy
- player 2 strategy

Abstraction: stochastic games (G)



- player 1 choice
- player 2 choice
- probabilistic choice
- play
- player 1 strategy
- player 2 strategy

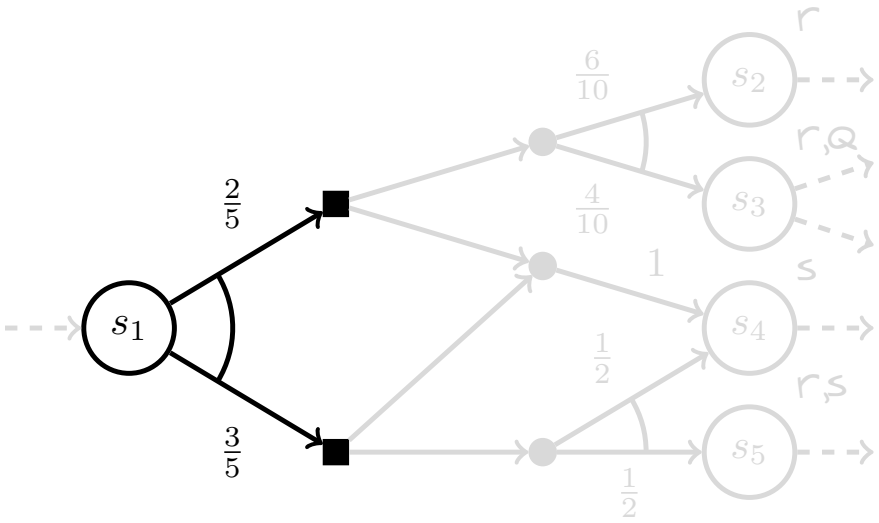
Abstraction: stochastic games (G)



- player 1 choice
- player 2 choice
- probabilistic choice

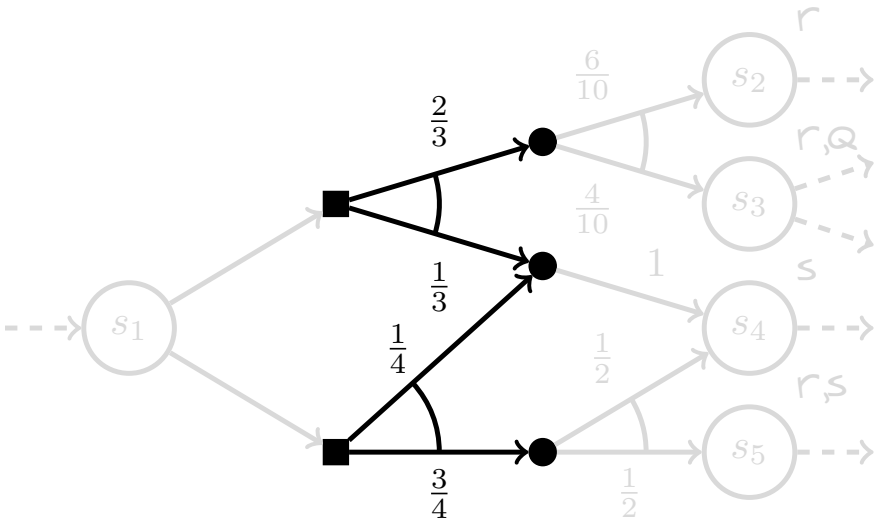
- play
- player 1 strategy
- player 2 strategy

Abstraction: stochastic games (G)



- player 1 choice
- player 2 choice
- probabilistic choice
- play
- player 1 strategy
- player 2 strategy

Abstraction: stochastic games (G)

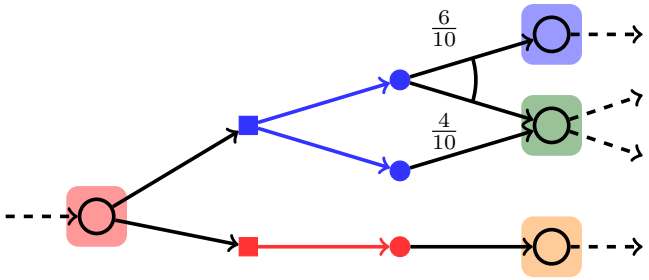


- player 1 choice
- player 2 choice
- probabilistic choice

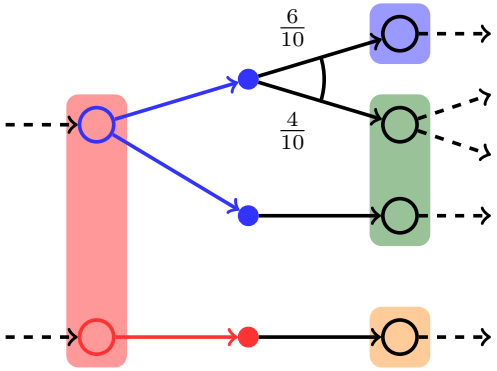
- play
- player 1 strategy
- player 2 strategy

Games as abstractions of MDPs

Game



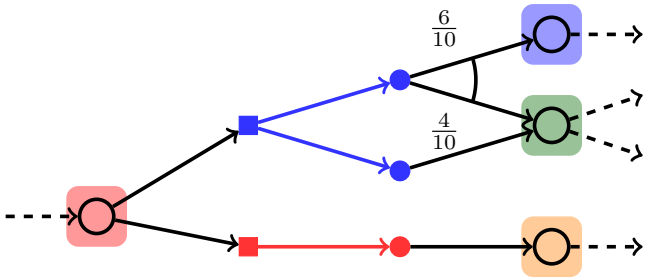
MDP



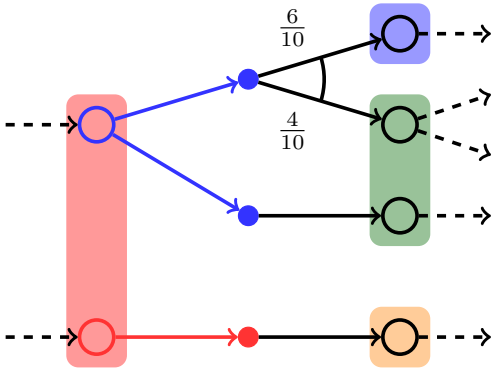
- player 1 "picks a concretisation"
- player 2 controls the non-determinism in the MDPs

Games as abstractions of MDPs

Game



MDP



- player 1 "picks a concretisation"
- player 2 controls the non-determinism in the MDPs

Overview

Concrete:

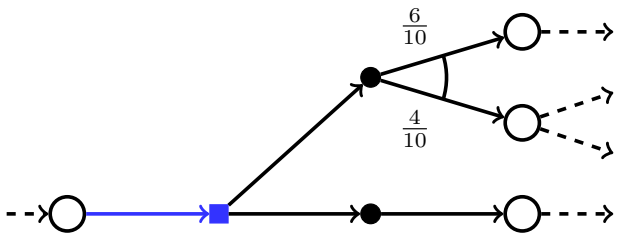
- M are models (Markov Decision Processes)
- P are specifications (Probabilistic CTL, [HJ94])
- $\models$ is a satisfaction relation ([BdA95])

Abstract:

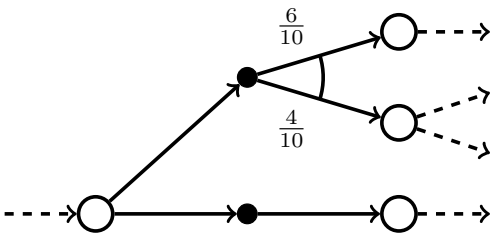
- G are abstract models (Games, [KNP06])
- $emb : \mathsf{M} \to \mathsf{G}$ is an embedding function (*this paper*)
- $\sqsubseteq_p \subseteq \mathsf{G} \times \mathsf{G}$ is an abstraction relation (*this paper*)
- $\models^{\mathsf{may}}, \models^{\mathsf{must}} \subseteq \mathsf{G} \times \mathsf{P}$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

Embedding function emb : M → G

Game emb(m)



MDP m



– i.e. add a trivial player 1 transitions everywhere

Overview

Concrete:

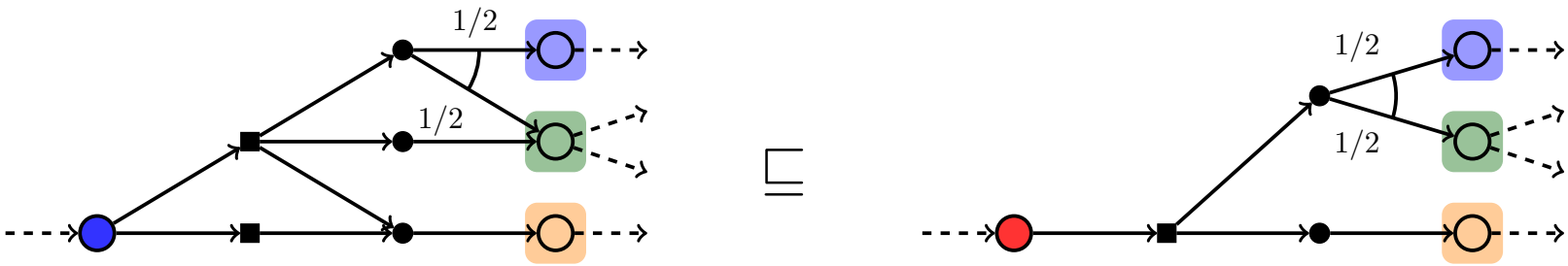
- M are models (Markov Decision Processes)
- P are specifications (Probabilistic CTL, [HJ94])
- $\models$ is a satisfaction relation ([BdA95])

Abstract:

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

Abstraction relation $\sqsubseteq$

"Abstract" game **"Concrete" game**



iff

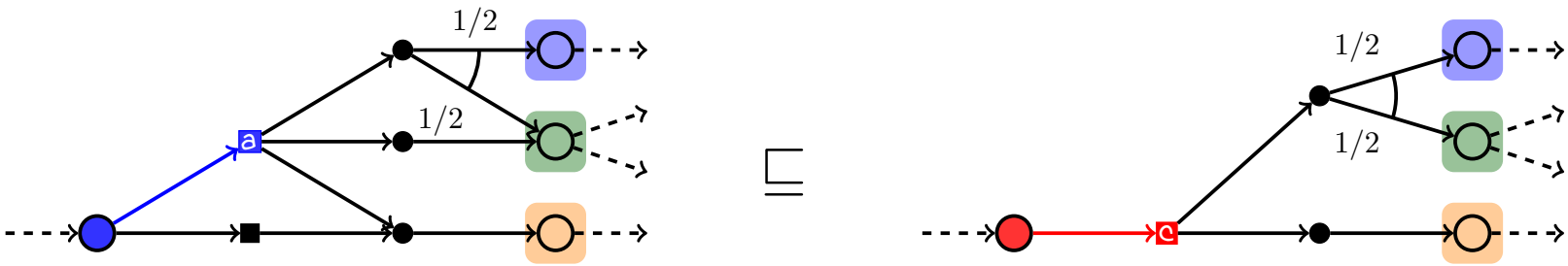
– player 1 in ● can over-approximate player 1 in ●:
$\forall$ ●→c : $\exists$ ●→a : $\forall$ c→c : $\exists$ a→a : a $\sqsubseteq$ c

– player 1 in ● can under-approximate player 1 in ●:
$\forall$ ●→c : $\exists$ ●→a : $\forall$ a→a : $\exists$ c→c : a $\sqsubseteq$ c

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

Abstraction relation $\sqsubseteq$

"Abstract" game **"Concrete" game**



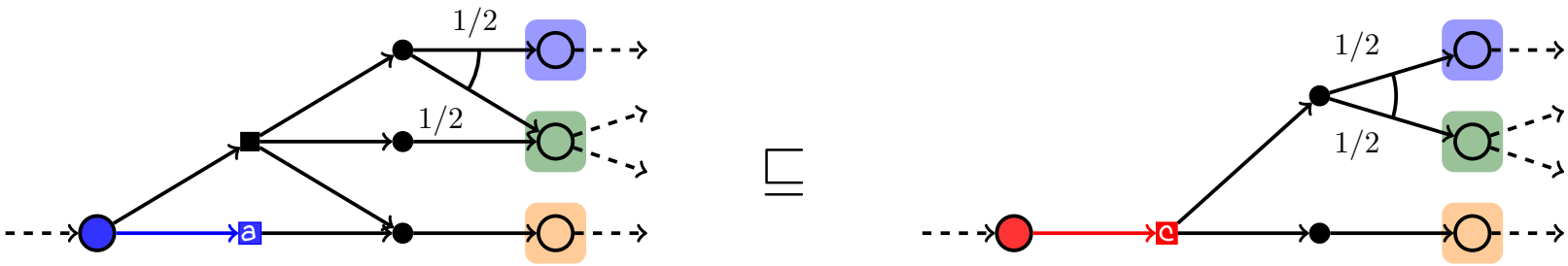
iff

- player 1 in ● can over-approximate player 1 in ●:
  $\forall$ ●→c : $\exists$ ●→a : $\forall$ c→c : $\exists$ a→a : a $\sqsubseteq$ c
- player 1 in ● can under-approximate player 1 in ●:
  $\forall$ ●→c : $\exists$ ●→a : $\forall$ a→a : $\exists$ c→c : a $\sqsubseteq$ c

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

Abstraction relation $\sqsubseteq$

"Abstract" game **"Concrete" game**



iff

– player 1 in ● can over-approximate player 1 in ●:

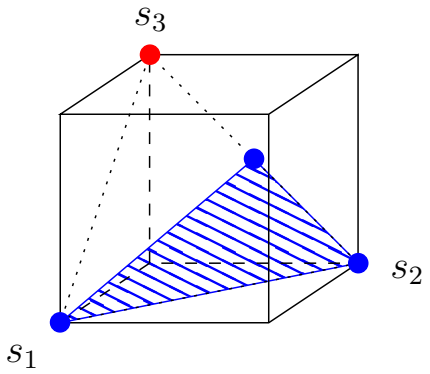
$\forall$ ●$\xrightarrow{c}$: $\exists$ ●$\xrightarrow{a}$: $\forall$ $\xrightarrow{c}$: $\exists$ $\xrightarrow{a}$: a $\sqsubseteq$ c

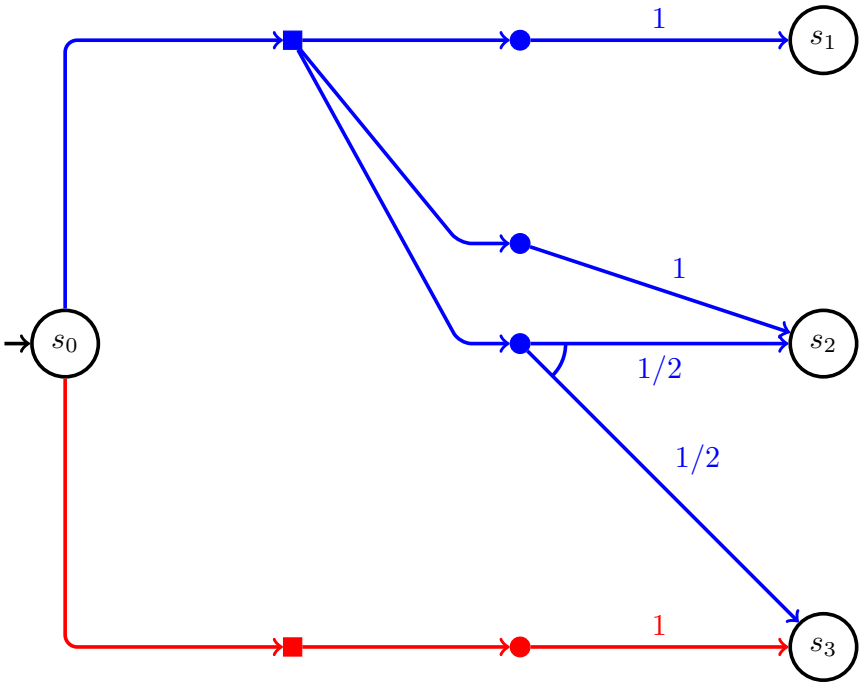
– player 1 in ● can under-approximate player 1 in ●:

$\forall$ ●$\xrightarrow{c}$: $\exists$ ●$\xrightarrow{a}$: $\forall$ $\xrightarrow{a}$: $\exists$ $\xrightarrow{c}$: a $\sqsubseteq$ c

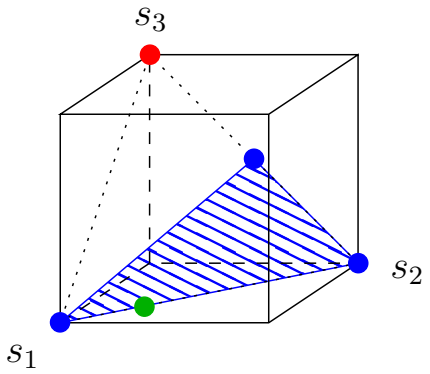
(here $\sqsubseteq$ is lifted to distributions with standard methods [JL91])

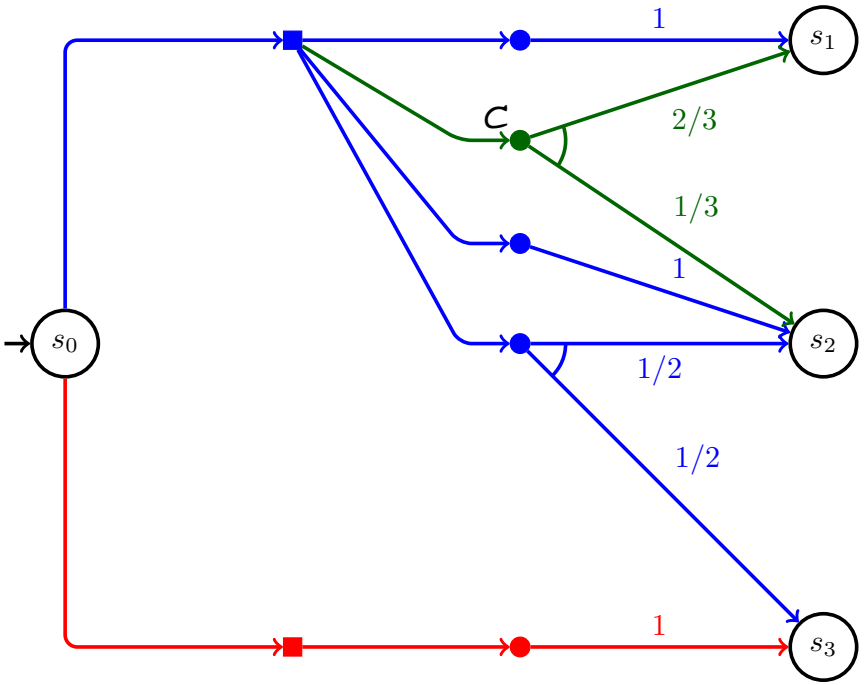
Combined transitions (c.f. [Seg95])



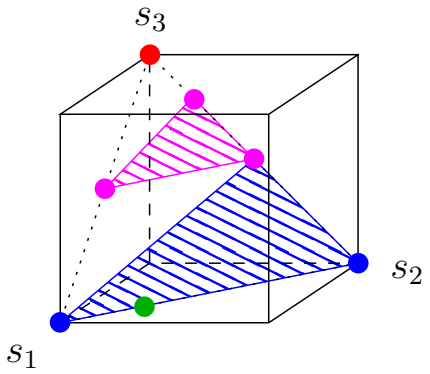


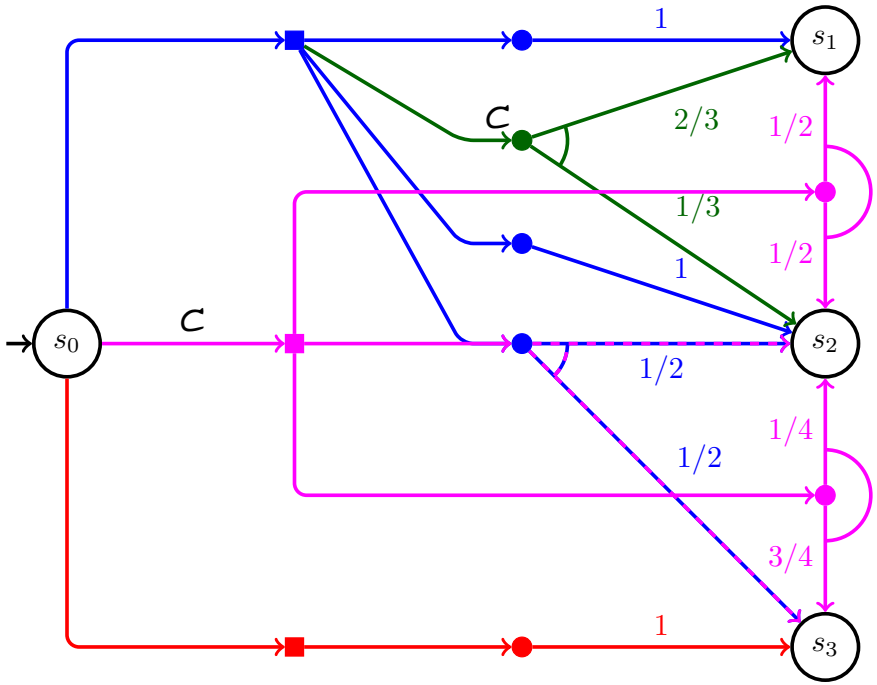
Combined transitions (c.f. [Seg95])



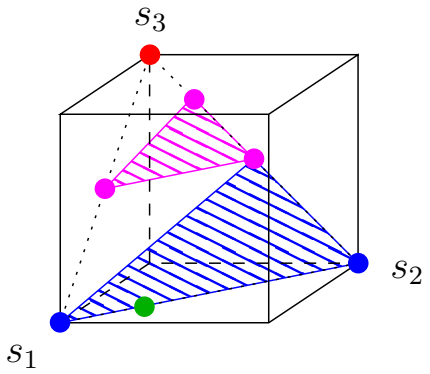


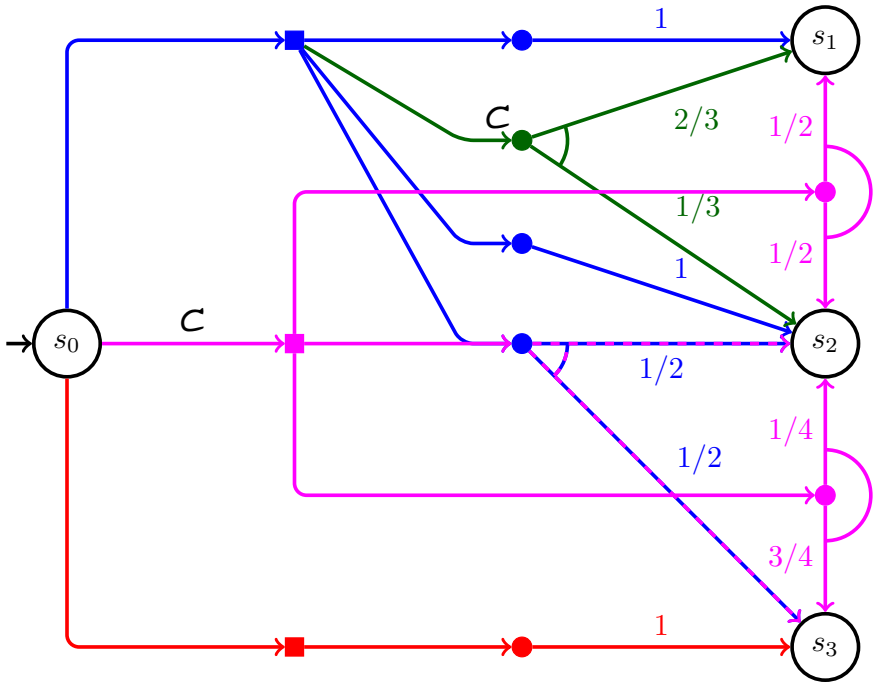
Combined transitions (c.f. [Seg95])





Combined transitions (c.f. [Seg95])





We now have ● $\sqsubseteq_p$ ● iff

– ∀ ● → c : ∃ ● —c→ a : ∀ c → c : ∃ a —c→ a : a ⊑ c

– ∀ ● → c : ∃ ● —c→ a : ∀ a → a : ∃ c —c→ c : a ⊑ c

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])
- $emb : \mathsf{M} \to \mathsf{G}$ is an embedding function (*this paper*)
- $\sqsubseteq_p \subseteq \mathsf{G} \times \mathsf{G}$ is an abstraction relation (*this paper*)
- $\models^{\mathsf{may}}, \models^{\mathsf{must}} \subseteq \mathsf{G} \times \mathsf{P}$ are PCTL semantics (*this paper*)

PCTL semantics $\models^{may}$, $\models^{must}$ for games

Player 1 & 2 revisited:

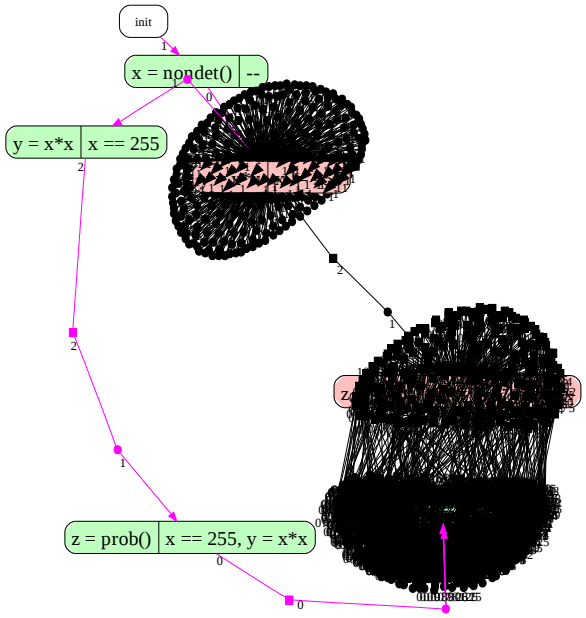
- player 1 strategies quantify over concretisations
- player 2 strategies correspond to MDP-strategies

Informal PCTL semantics:

Specification	must-satisfied	may-satisfied
for all strategies ...	for all player 1 strategies and for all player 2 strategies ...	for some player 1 strategies and for all player 2 strategies ...
for some strategy ...	for all player 1 strategies and for some player 2 strategy ...	for some player 1 strategies and for some player 2 strategy ...

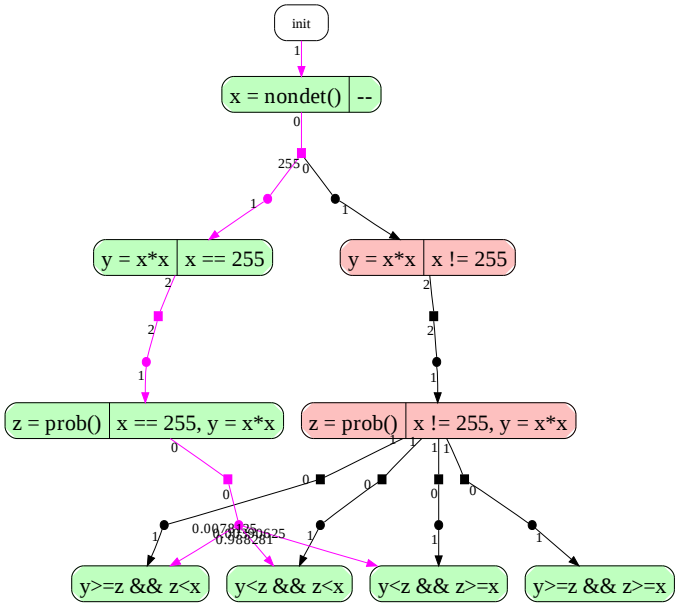
Games as abstractions (revisited)

Currently:



– identifies one game abstraction

This paper:

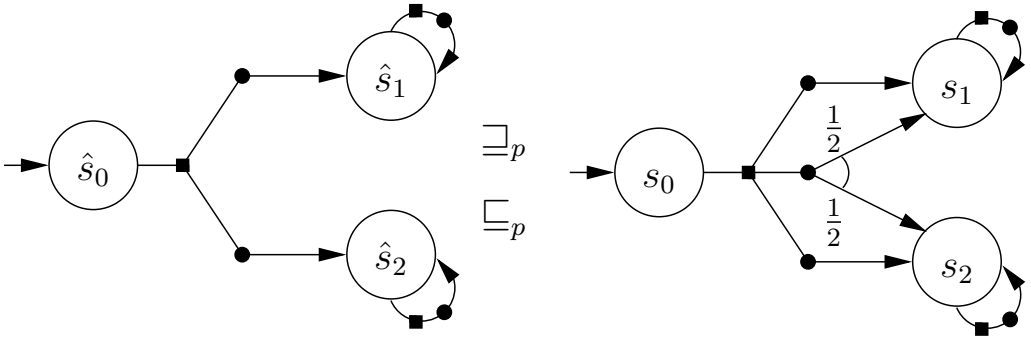


– identifies many game abstractions

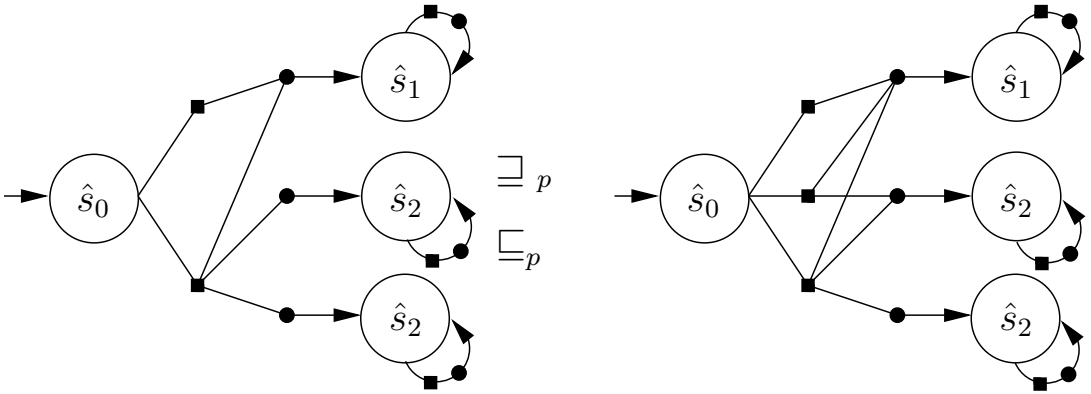
Finding better games (examples)

Without losing precision:

- remove non-extreme player 2 transitions

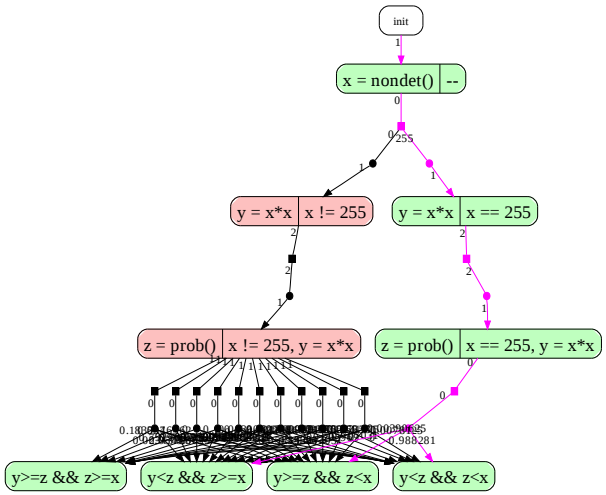


- remove non-extreme player 1 transitions



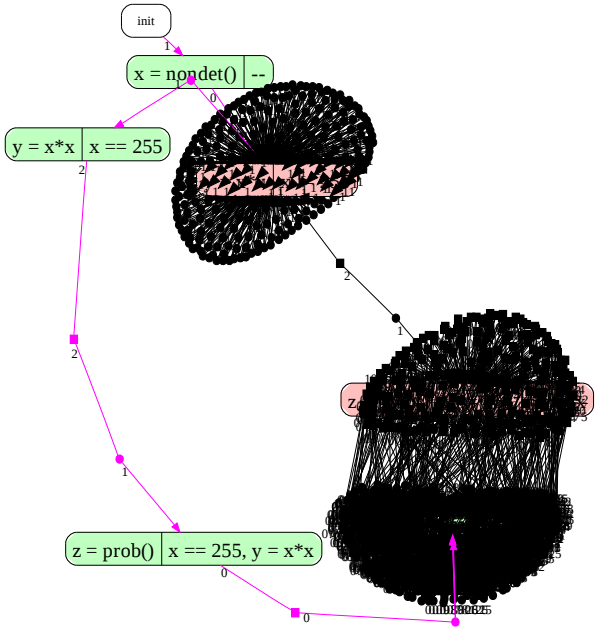
Finding better games (cont'd)

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



$\sqsubseteq_p$

$\sqsupseteq_p$

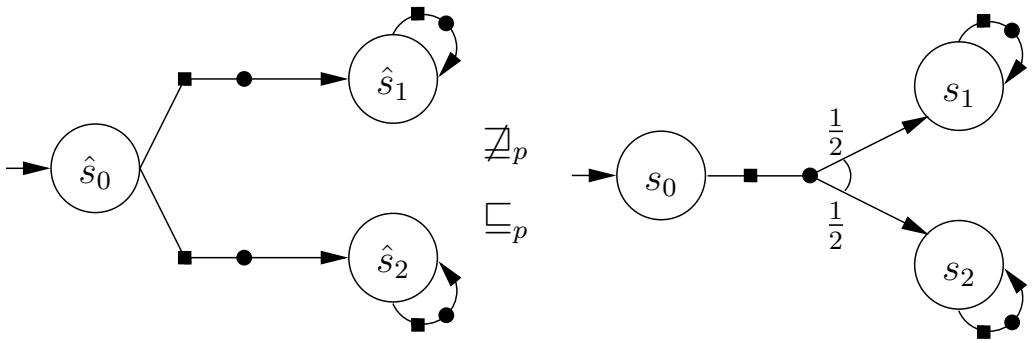


– both abstractions are equivalent!

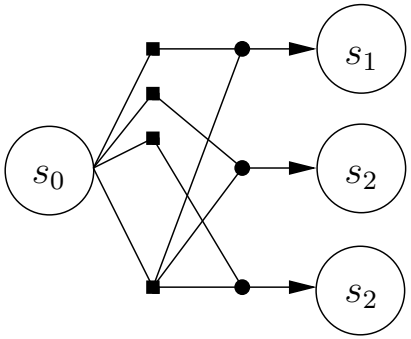
Finding better games (examples)

By losing precision:

- over-approximate probabilistic choice

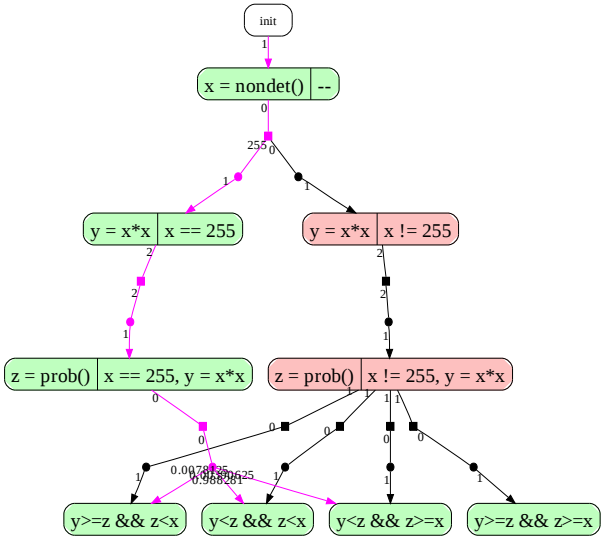


- use most abstract game over a partition



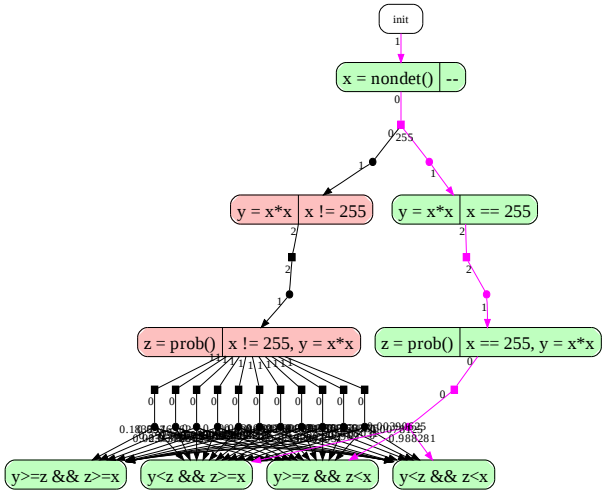
Finding better games (cont'd)

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



$\sqsubseteq_p$

$\not\sqsubseteq_p$



– the left-hand game is more abstract (but still "good enough")

Summary

Summary:

- identified the need for a more fine-grained abstraction for games
- developed an abstraction framework (*emb*, $\sqsubseteq_p$ 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

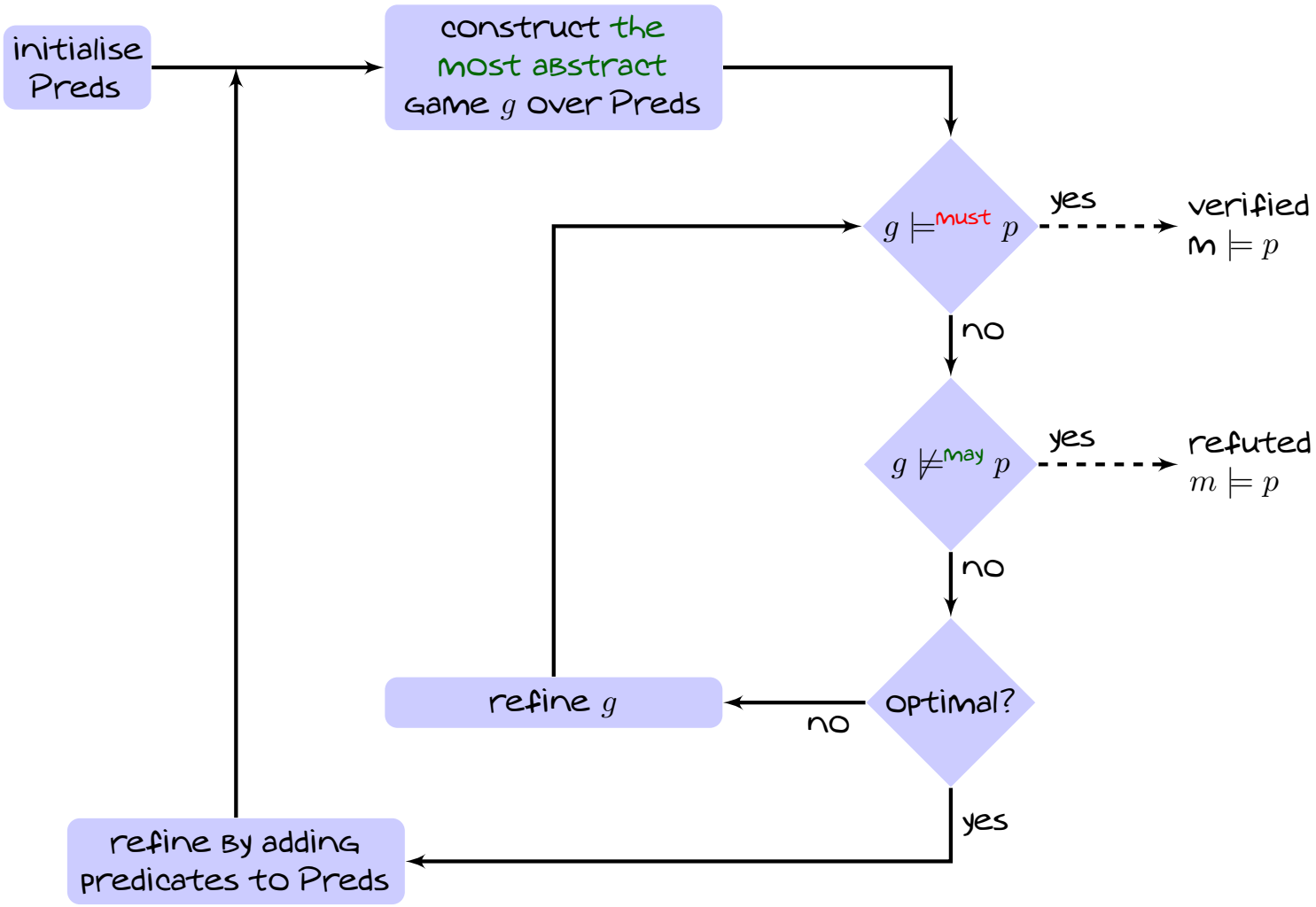
- automatically find good abstractions

Questions

Questions

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

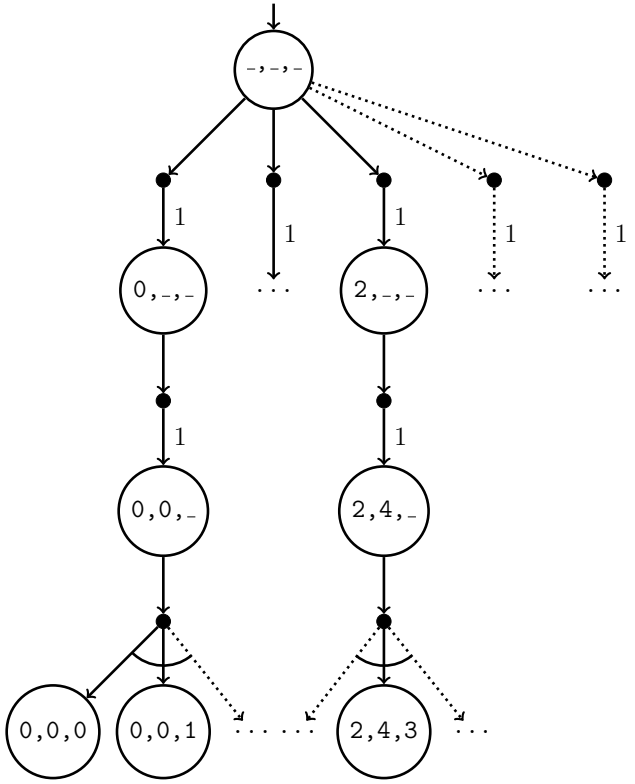
Abstraction refinement with games



– the machinery developed in this paper facilitates the inner refinement loop

Running example

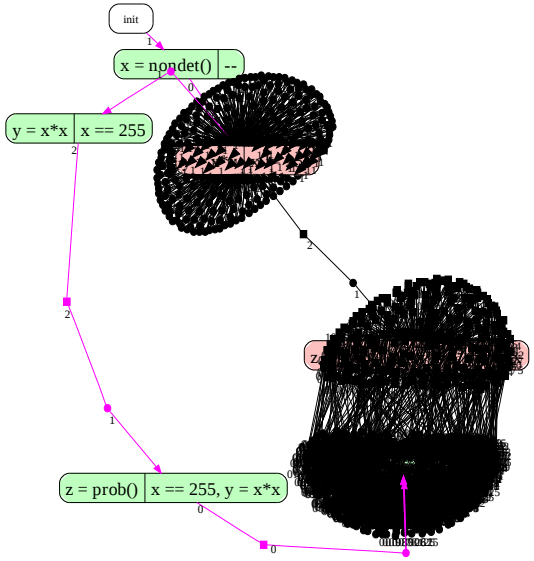
```
void main()
{
  uchar x,y,z;

  x = nondet();
  y = x*x;
  z = prob();

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



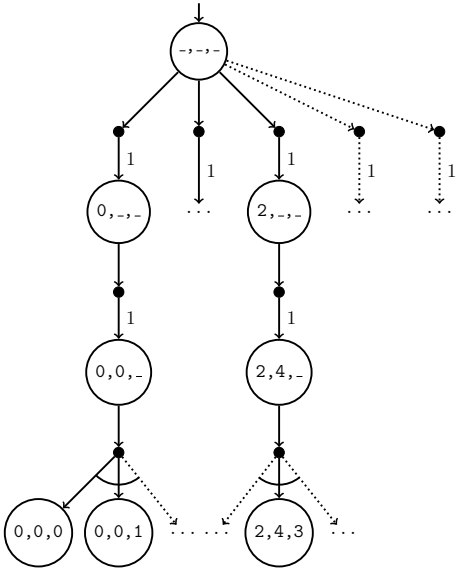
– does `main` satisfy $\mathsf{P}_{\leq 0.98}\langle \mathsf{F}\ \texttt{TARGET}\rangle$? (hint: take `x`=255)

Running example (cont'd)

refutation via emb, $\sqsubseteq_p$ and $\models^{\text{may}}$



$\sqsubseteq_p$ *emb*(



)

– the abstraction does not may-satisfy $\mathsf{P}_{\leq 0.98}\langle \mathsf{F}\ \mathtt{TARGET}\rangle$

Andrea Bianco and Luca de Alfaro.
Model checking of probabilistic and nondeterministic systems.
In P. S. Thiagarajan, editor, *Proceedings of the 15th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTC '95)*, volume 1026 of *Lecture Notes in Computer Science*, pages 499–513. Springer, 1995.

Hans Hansson and Bengt Jonsson.
A logic for reasoning about time and reliability.
Formal Aspects of Computing, 6:512–535, 1994.

Bengt Jonsson and Kim G. Larsen.
Specification and refinement of probabilistic processes.
In *Proceedings of Sixth Annual IEEE Symposium on Logic in Computer Science (LICS '91)*, pages 266–277. IEEE, 1991.

M. Kattenbelt, M. Kwiatkowska, G. Norman, and D. Parker.
A game-based abstraction-refinement framework for Markov decision processes.
Technical Report RR-08-06, Oxford University Computing Laboratory, April 2008.

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

In Neil D. Jones and Markus Müller-Olm, editors, *Proceedings of the 10th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI '09)*, volume 5403 of *Lecture Notes in Computer Science*, pages 182–197. Springer, 2009.

M. Kwiatkowska, G. Norman, and D. Parker.
Game-based abstraction for Markov decision processes.
In *Proceedings of the 3rd International Conference on Quantitative Evaluation of Systems (QEST '06)*, pages 157–166. IEEE Computer Society, 2006.

M. Kwiatkowska, G. Norman, and D. Parker.
Stochastic games for verification of probabilistic timed automata.

In J. Ouaknine and F. Vaandrager, editors, *Proc. 7th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'09)*, volume 5813 of *LNCS*, pages 212–227. Springer, 2009.

Kim Guldstrand Larsen.
Modal specifications.
In Joseph Sifakis, editor, *Proceedings of the International Workshop on Automatic Verification Methods for Finite State*

Systems, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 1990.

R. Segala.
Modelling and Verification of Randomized Distributed Protocols.
PhD thesis, Massachusetts Institute of Technology, 1995.