May and Must Abstractions for Markov Decision Processes

Anvesh Komuravelli, David Henriques, Fei He and Edmund Clarke

Carnegie Mellon

NSF Expeditions in Computing

Introduction

- Markov decision processes (MDPs) are popular for modeling systems that exhibit both probabilistic and non-deterministic behavior.
- Useful quantitative properties over MDPs can be automatically verified with probabilistic model checking (PMC), a popular formal verification technique.
- Unfortunately, PMC suffers from the state explosion problem. Abstraction-Refinement techniques can be used to tackle this problem.
- Abstraction-Refinement methods are lacking in this context, mainly developed for small classes of properties.

Background, Objectives & Claims

Background

- Abstractions usually have smaller state spaces than the original model.
- A useful abstraction preserves some properties, that is, if the property holds in the abstraction, it holds in the original model.
- Abstraction for PMC has been limited to reachability, an important but restrictive class of properties.
- Two tools have been developed for abstraction for reachability properties: PRISM uses game based abstraction and PASS uses predicate abstraction.

Objectives & Claims

- We aim at extending the classes of properties for which abstraction methods can be used.

  Our method **preserves both the safety and the liveness** fragment of Probabilistic CTL (PCTL).
- Counterexamples and counterexample analysis in PMC are complex. We want a technique that does not rely on counterexamples.

  Our technique uses **two abstractions**: an over- and an under-approximation, allowing for preservation of both validity and refutation.

Methodology

Safety and Liveness

$\varphi_S := \neg p \mid \varphi_S \wedge \varphi_S \mid \varphi_S \vee \varphi_S \mid \mathbf{A}_{\rhd x}(\mathbf{X}\varphi_S) \mid \mathbf{A}_{\rhd x}(\varphi_S \mathbf{W} \varphi_S)$

$\varphi_L := p \mid \varphi_L \wedge \varphi_L \mid \varphi_L \vee \varphi_L \mid \mathbf{E}_{\rhd x}(\mathbf{X}\varphi_L) \mid \mathbf{E}_{\rhd x}(\varphi_L \mathbf{U} \varphi_L)$

Simulation

MDP $\mathcal{M}'$ simulates $\mathcal{M}$ if there is $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}'$ s.t.

- $s\mathcal{R}s'$ then $\mathcal{L}(s) \subseteq \mathcal{L}(s')$
- if $s \to \mu$, $s\mathcal{R}s'$ there is μ' s.t. $s' \to \mu'$ and $\mu \sqsubseteq_{\mathcal{R}} \mu'$
- $s_i \mathcal{R} s_i'$

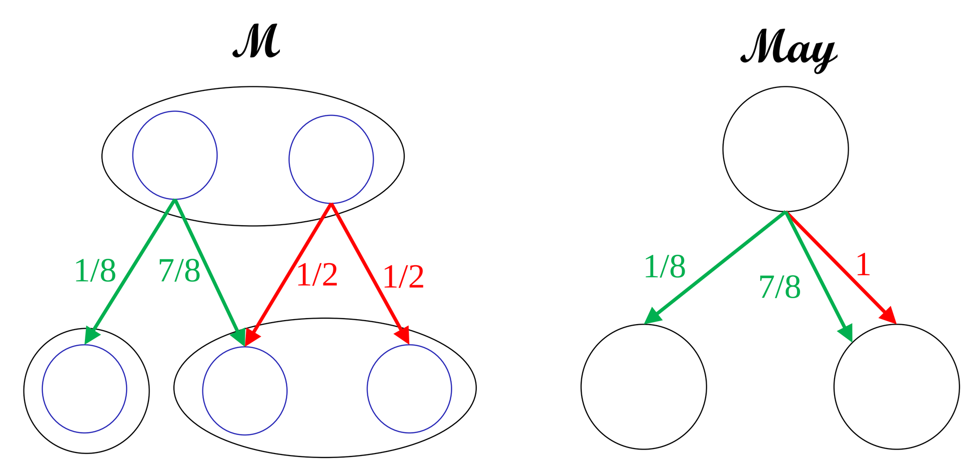
PreservationTheorem

Let $\mathcal{M}'$ simulate $\mathcal{M}$. If $\mathcal{M}' \vDash \varphi_S$ then $\mathcal{M} \vDash \varphi_S$. If $\mathcal{M} \vDash \varphi_L$ then $\mathcal{M}' \vDash \varphi_L$.

May Abstractions

Consider a partition $\mathcal{P}$ of $\mathcal{S}$,

- states of the abstraction are elements of $\mathcal{P}$
- for each transition in $\mathcal{M}$, its lifting is in $\mathcal{M}'$



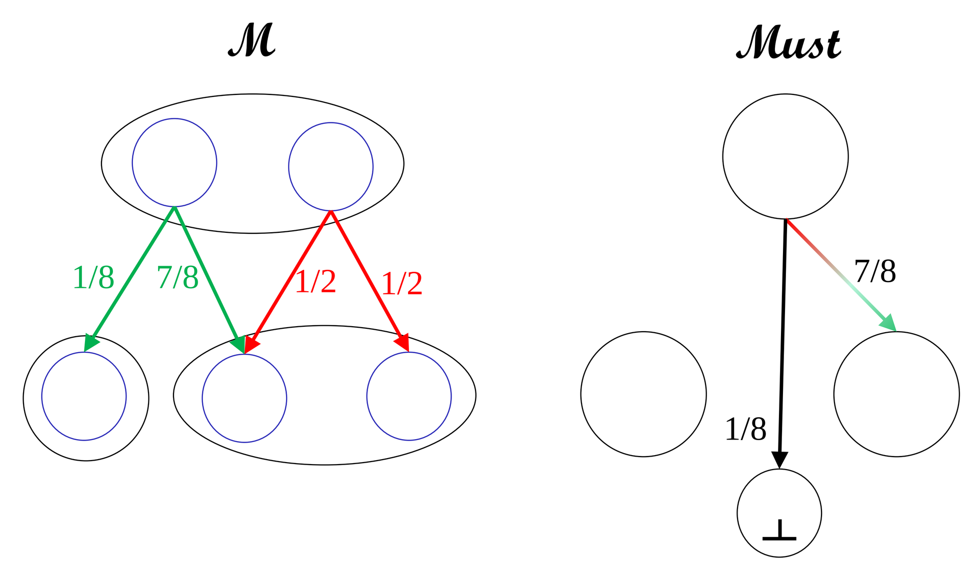
$\mathcal{M}$ is simulated by $\mathcal{May}$

Methodology

Must Abstractions

Consider a partition $\mathcal{P}$ of $\mathcal{S}$,

- states of the abstraction are elements of $\mathcal{P}$
- transitions in the abstraction are the common parts of liftings of distributions among concrete states in each partition class



$\mathcal{M}$ simulates $\mathcal{Must}$

Refinement

- Strategy based refinement: check which distributions are responsible for max and min probabilities. Refine partition to take out all states with those distributions.
- Spurious state refinement: check distributions responsible for max and min probabilities. Refine partition taking out states responsible for directing probability to spurious state.
- Propositional symbol refinement: Refine partition taking out states that expose some propositional symbol in the property.

Methodology

Monotonic Refinement

We introduce temporary states that record information while needed to guarantee monotonicity. However, this is an expensive operation that is not used in all steps.

Abstraction Refinement Loop

1. Initial partition is the trivial partition $\mathcal{S}$;
2. If $\mathcal{May} \vDash \varphi_S$ return **true**;
3. If $\neg$ ($\mathcal{Must} \vDash \varphi_S$) return **false**;
4. Pick a refinement method;
5. Refine $\mathcal{P}$ according to 4;
6. Goto 2;

This is guaranteed to stop because eventually the partition becomes the diagonal relation.

Results

Model	Property	Concrete Model (# states)	Game-based (# states)	May/Must (# may states*)	Game-based (s)	May/Must (s)	Game-based (# iters)	May/Must (# iters)
Zeroconf	Reachability	1065569	966	1808	525	3391	127	245
Zeroconf	Safety	26121	-	940	-	50	-	205
Wlan	Reachability	28480	258	525	5.5	7	70	179
GridWorld	Nested Safety	10201	-	431	-	13	-	86
Non-det GridWorld	Nested Safety	15555	-	2011	-	1518	-	434

*# must states = 2(# may states)-1

Conclusions & Future Work

- May and Must abstraction extends the class of properties amenable to abstraction in practice to the safe and live fragments of PCTL; for the reachability subset of properties, its efficiency is comparable to other abstraction techniques, although not as good.
- We believe that with minimal modifications, the method can be extended to Probabilistic Automata and used in compositional settings.
- From may and must literature for classical systems, it seems likely that we can further extend the class of verifiable properties to richer logics.

References

R. Chadha and M. Viswanathan. A Counterexample-Guided Abstraction-Refinement Framework for Markov Decision Processes. TOCL, 12(1):1-49, November 2010.

M. Kattenbelt, M. Kwiatkowska, G. Norman and D. Parker. A Game-based Abstraction-Refinement Framework for Markov Decision Processes.Formal Methods in System Design, 36(3), pages 246-280, Springer.September 2010.

E.M. Hahn, H. Hermanns, B. Wachter and L. Zhang. PASS: Abstraction Refinement for Infinite Probabilistic Models. TACAS, pages 353-357, 2010