Generation and Exploitation of Counterexamples in Stochastic Models

Chloé Capon

Mathematics Department
University of Mons

UMONS

Faculté
des Sciences

October 16, 2023
Masaryk University, Brno (Czech Republic)

Motivations

- Reactive systems are systems that continuously interact with their environment.
- We are interested in the correctness of critical reactive systems, e.g., ABS for cars.

Motivations

- Reactive systems are systems that continuously interact with their environment.
- We are interested in the correctness of critical reactive systems, e.g., ABS for cars.

Verification

Given a formal model of the system and a specification, the goal is to check that the system satisfies the specification.

Motivations

- Reactive systems are systems that continuously interact with their environment.
- We are interested in the correctness of critical reactive systems, e.g., ABS for cars.

Verification

Given a formal model of the system and a specification, the goal is to check that the system satisfies the specification.

Synthesis

Given a system to control trying to enforce some specification within an uncontrollable environment, it aims at the automated construction of provably-safe system controllers.

Motivations

- Synthesis algorithm permit to construct a suitable controller if one exists.
- Otherwise, they simply tell us that no such controller exists.

$\rightsquigarrow$ what happens in practice?

Motivations

- Synthesis algorithm permit to construct a suitable controller if one exists.
- Otherwise, they simply tell us that no such controller exists.

$\rightsquigarrow$ what happens in practice?

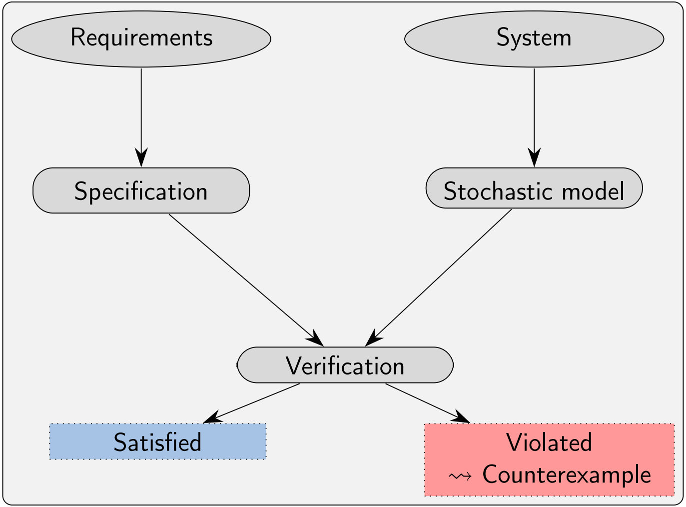
Idea

We need refinement mechanisms based on counterexamples that help practitioners understand:

1. why their attempt failed;
2. how they can patch the system – environment – specification triptych to make synthesis possible and obtain an adequate controller.

Verification

First, let us look at the verification of models containing only probabilistic transitions.

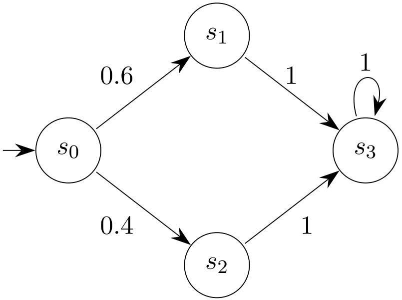


Markov Chains

Definition

A **Markov Chain** (MC) is a tuple $\mathcal{C} = (S, s_{\mathsf{init}}, \delta)$ where:

- S is a finite set of states;
- $s_{\mathsf{init}} \in S$ is an initial state;
- $\delta : S \times S \to [0, 1]$ is a transition probability function such that for all $s \in S$: $\sum_{s' \in S} \delta(s, s') \leq 1$.

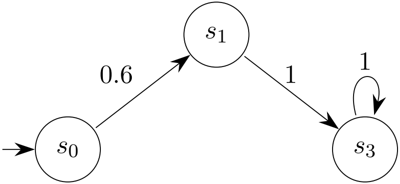


Markov Chains

Definition

A **Markov Chain** (MC) is a tuple $\mathcal{C} = (S, s_{\mathsf{init}}, \delta)$ where:

- S is a finite set of states;
- $s_{\mathsf{init}} \in S$ is an initial state;
- $\delta : S \times S \to [0, 1]$ is a transition probability function such that for all $s \in S$: $\sum_{s' \in S} \delta(s, s') \leq 1$.



Reachability properties

What is the probability to reach a set of states T when starting in state s?

Reachability properties

What is the probability to reach a set of states T when starting in state s?

Let us consider properties of the form: $\mathbb{P}_{\leq\lambda}(\Diamond T)$ for $\lambda \in \mathbb{Q} \cap [0, 1]$

Let $\mathcal{C} = (S, s_{\mathsf{init}}, \delta)$ be a Markov chain:

$$\mathcal{C} \models \mathbb{P}_{\leq\lambda}(\Diamond T) \text{ iff } \Pr(\{\mathsf{Paths}^{\mathcal{C}}_{\mathsf{fin}}(s_{\mathsf{init}}, T)\}) \leq \lambda.$$

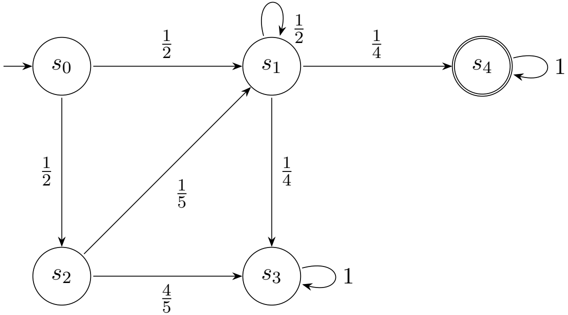
Reachability properties

What is the probability to reach a set of states T when starting in state s?

Let us consider properties of the form: $\mathbb{P}_{\leq\lambda}(\Diamond T)$ for $\lambda \in \mathbb{Q} \cap [0, 1]$

Let $\mathcal{C} = (S, s_{\text{init}}, \delta)$ be a Markov chain:

$$\mathcal{C} \models \mathbb{P}_{\leq\lambda}(\Diamond T) \text{ iff } \Pr(\{\text{Paths}^{\mathcal{C}}_{\text{fin}}(s_{\text{init}}, T)\}) \leq \lambda.$$



We have that $\Pr(\Diamond\{s_4\}) =$

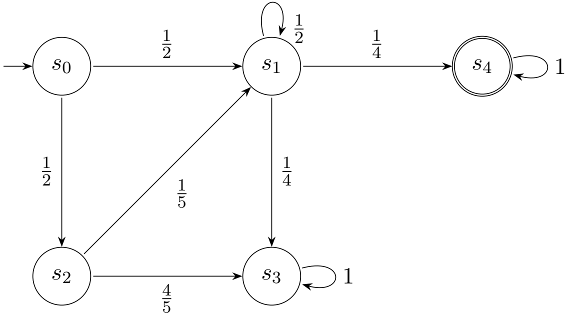
Reachability properties

What is the probability to reach a set of states T when starting in state s?

Let us consider properties of the form: $\mathbb{P}_{\leq\lambda}(\Diamond T)$ for $\lambda \in \mathbb{Q} \cap [0,1]$

Let $\mathcal{C} = (S,\ s_{\mathsf{init}},\ \delta)$ be a Markov chain:

$$\mathcal{C} \models \mathbb{P}_{\leq\lambda}(\Diamond T) \text{ iff } \Pr(\{\mathsf{Paths}^{\mathcal{C}}_{\mathsf{fin}}(s_{\mathsf{init}}, T)\}) \leq \lambda.$$



We have that $\Pr(\Diamond\{s_4\}) = \frac{3}{10}$

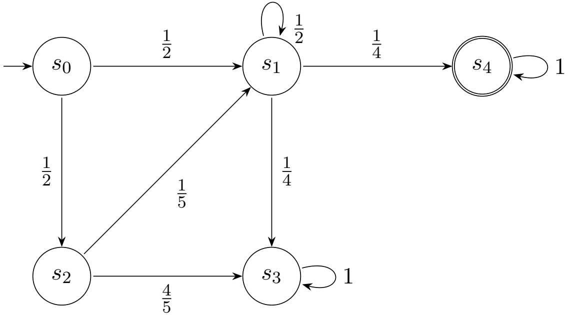
Reachability properties

What is the probability to reach a set of states T when starting in state s?

Let us consider properties of the form: $\mathbb{P}_{\leq \lambda}(\Diamond T)$ for $\lambda \in \mathbb{Q} \cap [0, 1]$

Let $\mathcal{C} = (S,\, s_{\mathsf{init}},\, \delta)$ be a Markov chain:

$$\mathcal{C} \models \mathbb{P}_{\leq \lambda}(\Diamond T) \text{ iff } \Pr(\{\mathsf{Paths}^{\mathcal{C}}_{\mathsf{fin}}(s_{\mathsf{init}}, T)\}) \leq \lambda.$$



We have that $\Pr(\Diamond\{s_4\}) = \frac{3}{10}$

$\rightsquigarrow \mathcal{C} \not\models \mathbb{P}_{\leq \frac{1}{5}}(\Diamond\{s_4\})$

Counterexamples for MCs

When a reachability property is not satisfied by an MC, we want to give an explanation of why the model violates this property.

Counterexamples for MCs

When a reachability property is not satisfied by an MC, we want to give an explanation of why the model violates this property.

This is the role of counterexamples.

Counterexamples for MCs

When a reachability property is not satisfied by an MC, we want to give an explanation of why the model violates this property.

This is the role of counterexamples.

A **counterexample** for $\mathbb{P}_{\leq\lambda}(\Diamond T)$ is a set of paths, each of them leading from the initial state to some target state in T, such that the probability of the path set is larger than λ.

Counterexamples for MCs

Different notions

Representations of counterexamples at different levels:

- At the level of paths:

The number of paths needed can be very large $\rightsquigarrow$ hard to understand and analyse.

Counterexamples for MCs

Different notions

Representations of counterexamples at different levels:

- At the level of paths:

The number of paths needed can be very large $\rightsquigarrow$ hard to understand and analyse.

- At the model level:

Counterexamples are parts of the model where the property is already violated.

Counterexamples for MCs

Different notions

Representations of counterexamples at different levels:

- At the level of paths:

The number of paths needed can be very large $\rightsquigarrow$ hard to understand and analyse.

- At the model level:

Counterexamples are parts of the model where the property is already violated.

$\hookrightarrow$ Notion of critical subsystem.

Definition

A subsystem $\mathcal{C}'$ of $\mathcal{C}$ is critical for $\mathbb{P}_{\leq\lambda}(\Diamond T)$ if $\mathcal{C}' \not\models \mathbb{P}_{\leq\lambda}(\Diamond T)$.

Critical subsystems for MCs

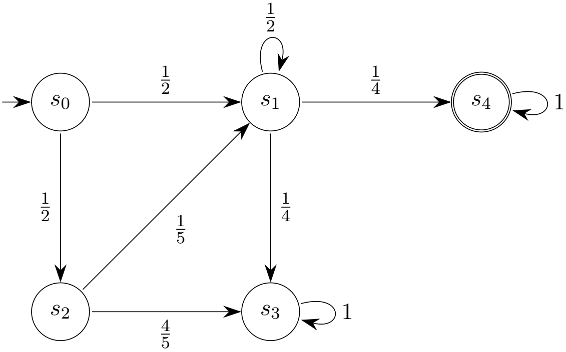
We look for counterexamples that are the most illustrative with regards to the violation of the property:

- A critical subsystem is minimal if it has a minimal set of states under all critical subsystems;
- A critical subsystem is a best critical subsystem if it has the largest probability to reach T among all minimal critical subsystems.

Critical subsystems for MCs

Example

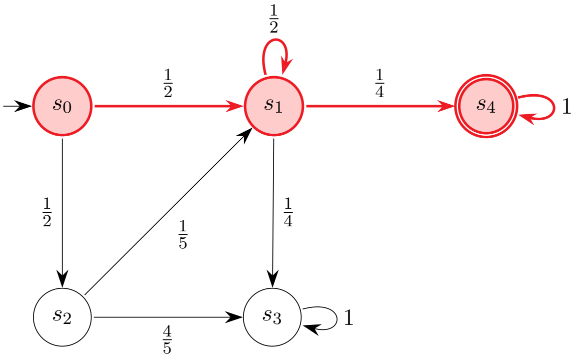
Let's consider this MC and $\mathbb{P}_{\leq \frac{1}{5}}(\Diamond\{s_4\})$:



Critical subsystems for MCs

Example

Let's consider this MC and $\mathbb{P}_{\leq \frac{1}{5}}(\Diamond\{s_4\})$:



This subsystem is a critical subsystem, since the probability to reach $\{s_4\}$ from s_0 is $\frac{1}{4} > \frac{1}{5}$.

Verification

Adding non-determinism

Now, we consider models with probabilistic transitions and non-deterministic choices: used to describe transitions for which we do not know the exact probabilities.

Verification

Adding non-determinism

Now, we consider models with probabilistic transitions and non-deterministic choices: used to describe transitions for which we do not know the exact probabilities.

Verification

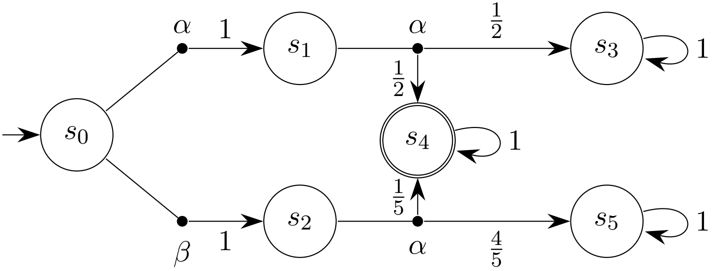
Verify that no matter how we resolve the non-determinism (i.e., no matter the strategy), the specification is verified.

Markov Decision Processes

Definition

A **Markov decision process** (MDP) is a tuple $\mathcal{M} = (S, s_{\mathsf{init}}, A, \delta)$ where:

- S is a finite set of states, s_{init} is an initial state, A is a set of actions;
- $\delta : S \times A \times S \to [0,1]$ is a transition probability function such that for all states $s \in S$ and actions $\alpha \in A(s)$: $\sum_{s' \in S} \delta(s, \alpha, s') \leq 1$.

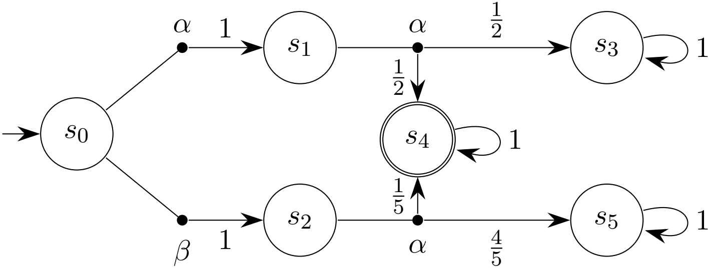


Markov Decision Processes

Strategies

Definition

A **strategy** for an MDP $\mathcal{M} = (S, s_{\text{init}}, A, \delta)$ is a function $\sigma : S \to A$. We denote the set of strategies on $\mathcal{M}$ by $\mathsf{Strat}(\mathcal{M})$.

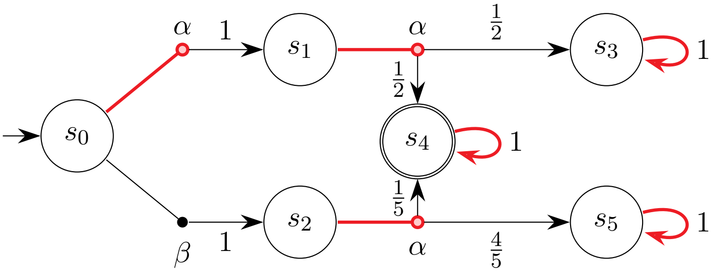


Markov Decision Processes

Strategies

Definition

A **strategy** for an MDP $\mathcal{M} = (S, s_{\mathsf{init}}, A, \delta)$ is a function $\sigma : S \to A$. We denote the set of strategies on $\mathcal{M}$ by $\mathsf{Strat}(\mathcal{M})$.



Universal reachability properties

For MDPs, $\mathcal{M} \models \mathbb{P}^{\forall}_{\leq\lambda}(\Diamond T)$ expresses that:

$$\forall \sigma \in \mathsf{Strat}(\mathcal{M}),\ \mathcal{M}^{\sigma} \models \mathbb{P}_{\leq\lambda}(\Diamond T).$$

Universal reachability properties

For MDPs, $\mathcal{M} \models \mathbb{P}^{\forall}_{\leq\lambda}(\Diamond T)$ expresses that:

$$\forall\sigma \in \mathsf{Strat}(\mathcal{M}),\ \mathcal{M}^{\sigma} \models \mathbb{P}_{\leq\lambda}(\Diamond T).$$

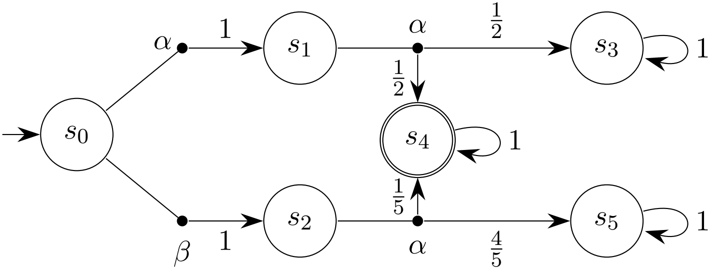
A counterexample consists in finding a strategy $\sigma \in \mathsf{Strat}(\mathcal{M})$ such that the MC induced by σ does not satisfy the corresponding reachability property and extract a critical subsystem.

Such strategies are called critical strategies.

Counterexamples for MDPs

Example

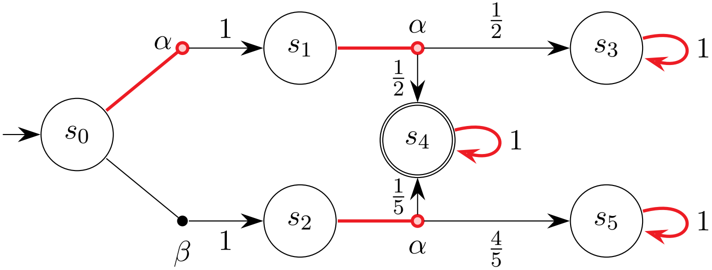
Let us consider the following MDP and $\mathbb{P}^{\forall}_{\leq \frac{2}{5}}(\Diamond\{s_4\})$.



Counterexamples for MDPs

Example

Let us consider the following MDP and $\mathbb{P}^{\forall}_{\leq \frac{2}{5}}(\Diamond\{s_4\})$.

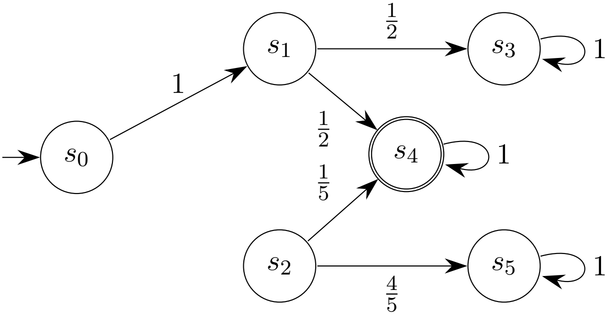


This strategy is critical.

Counterexamples for MDPs

Example

Let's consider the following MDP and $\mathbb{P}^{\forall}_{\leq \frac{2}{5}}(\Diamond\{s_4\})$.

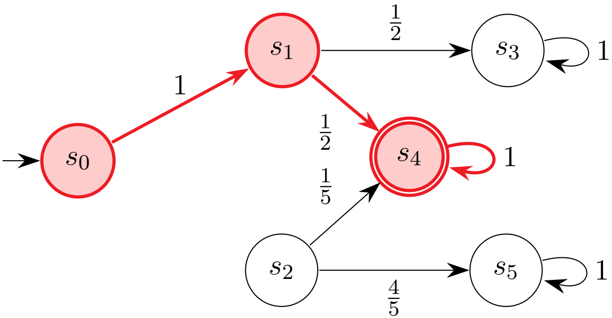


The probability to reach $\{s_4\}$ in the resulting MC is equal to $\frac{1}{2}$.

Counterexamples for MDPs

Example

Let's consider the following MDP and $\mathbb{P}^{\forall}_{\leq \frac{2}{5}}(\Diamond\{s_4\})$.



A best critical subsystem, where $\Pr(\Diamond\{s_4\}) = \frac{1}{2}$.

Generation of counterexamples

Existing works about counterexamples in stochastic models:

- In MCs[1] and MDPs[2]: path enumeration, critical subsystems;
- For omega-regular properties and for expected rewards[3];

All these works focus on the verification problem: proving the existence of a bad controller, but not the non-existence of a suitable one.

[1] Ábrahám et al., "Counterexample Generation for Discrete-Time Markov Models: An Introductory Survey".

[2] Wimmer et al., "Minimal counterexamples for linear-time probabilistic verification".

[3] Quatmann et al., "Counterexamples for Expected Rewards".

Generation of counterexamples

Existing works about counterexamples in stochastic models:

- In MCs[1] and MDPs[2]: path enumeration, critical subsystems;
- For omega-regular properties and for expected rewards[3];

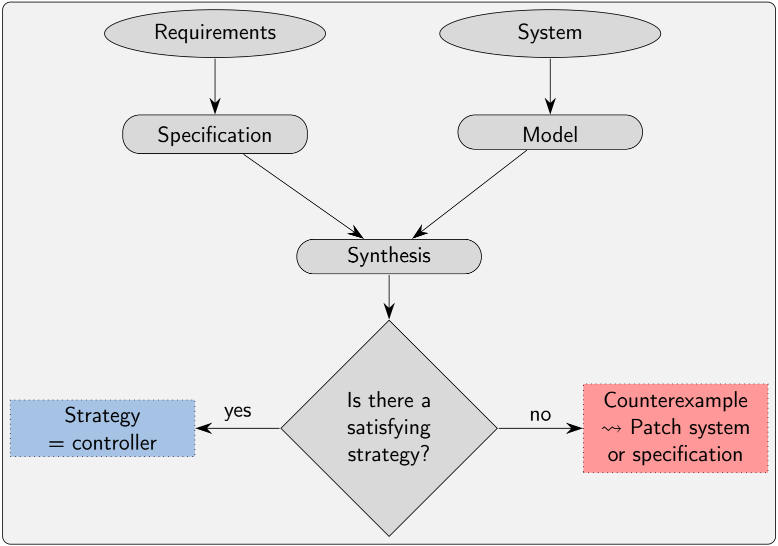
All these works focus on the verification problem: proving the existence of a bad controller, but not the non-existence of a suitable one.

What about synthesis?

In this case, the non-determinism is used to model actions of a system that can be controlled.
$\rightsquigarrow$ Find a controller (a suitable strategy) such that the induced MC satisfies the specification.

[1] Ábrahám et al., "Counterexample Generation for Discrete-Time Markov Models: An Introductory Survey".
[2] Wimmer et al., "Minimal counterexamples for linear-time probabilistic verification".
[3] Quatmann et al., "Counterexamples for Expected Rewards".

Synthesis


Requirements
System
Specification
Model
Synthesis
Is there a satisfying strategy?
yes
no
Strategy = controller
Counterexample ⇝ Patch system or specification

Generation of counterexamples

Synthesis

Verification

- Specification of the form $\mathbb{P}^{\forall}_{\leq\lambda}(\Diamond T)$: every strategy has a probability smaller than λ to reach T;
- If this specification is false, a counterexample needs to show a strategy that has a probability larger than λ to reach T.

Generation of counterexamples

Synthesis

Verification

- Specification of the form $\mathbb{P}^{\forall}_{\leq\lambda}(\Diamond T)$: every strategy has a probability smaller than λ to reach T;
- If this specification is false, a counterexample needs to show a strategy that has a probability larger than λ to reach T.

Synthesis

- Specification of the form $\mathbb{P}^{\exists}_{\leq\lambda}(\Diamond T)$: there exists a strategy with probability smaller than λ to reach T;
- If this specification is false, a counterexample needs to show that every strategy has a probability larger than λ to reach T.

Synthesis mindset

Critical subsystem

Let us consider an MDP $\mathcal{M} = (S, s_{\mathsf{init}}, A, \delta)$.

A **subsystem** $\mathcal{M}'$ of $\mathcal{M}$ is an MDP defined by a set of states $S' \subseteq S$ where for each state $s \in S'$, we have that every action of $A(s)$ is in $\mathcal{M}'$.

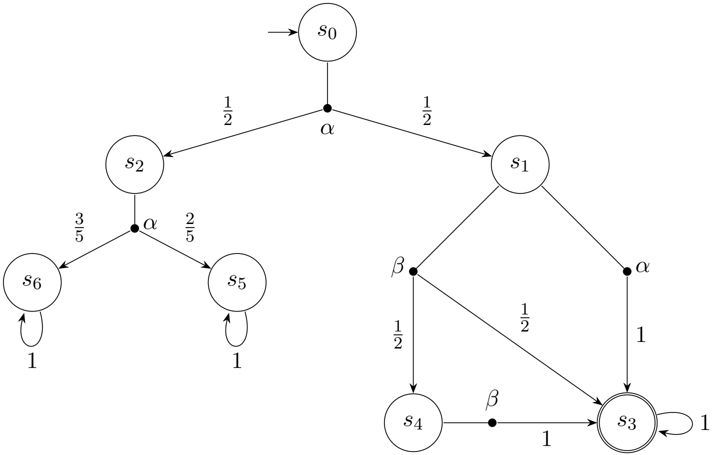
Definition

A subsystem of $\mathcal{M}$ is **critical** for $\mathbb{P}^{\exists}_{\leq\lambda}(\Diamond T)$ if for every strategy σ of $\mathcal{M}'$ we have that $\mathrm{Pr}^{\sigma}_{\mathcal{M}'}(\Diamond T) > \lambda$.

Synthesis mindset

Critical subsystem

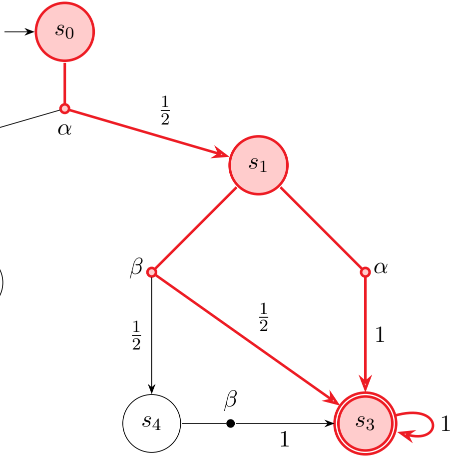
Let's consider an MDP $\mathcal{M}$ and $\mathbb{P}^{\exists}_{\leq \frac{1}{5}}(\Diamond\{s_3\})$.



Synthesis mindset

Critical subsystem

Let's consider an MDP $\mathcal{M}$ and $\mathbb{P}^{\exists}_{\leq \frac{1}{5}}(\Diamond\{s_3\})$.



How to generate them?

- For verification: MILP formulations that compute best critical subsystems for MCs[4] and MDPs[5] have been implemented.
- For synthesis: we developed an MILP formulation that computes best critical subsystems with the new mindset.

[4] Ábrahám et al., "Counterexample Generation for Discrete-Time Markov Models: An Introductory Survey".

[5] Wimmer et al., "Minimal counterexamples for linear-time probabilistic verification".

How to generate them?

- For verification: MILP formulations that compute best critical subsystems for MCs[4] and MDPs[5] have been implemented.
- For synthesis: we developed an MILP formulation that computes best critical subsystems with the new mindset.

The idea of our approach

- Only consider a minimising strategy of the subsystem;
- If the probability of this strategy to reach T is already too large, then every strategy will have a probability to reach T that is too large.

[4] Ábrahám et al., "Counterexample Generation for Discrete-Time Markov Models: An Introductory Survey".

[5] Wimmer et al., "Minimal counterexamples for linear-time probabilistic verification".

Exploitation of counterexamples

Two approaches that exploit the information given by counterexamples:

[6] Clarke et al., "Counterexample-guided abstraction refinement for symbolic model checking"; Hermanns, Wachter, and Zhang, "Probabilistic CEGAR".

[7] Ceska et al., "Counterexample-guided inductive synthesis for probabilistic systems".

Exploitation of counterexamples

Two approaches that exploit the information given by counterexamples:

1. Counterexample-guided abstraction refinement (CEGAR)[6] : verification of large systems through smaller abstractions.
   - If the abstraction satisfies the specification $\rightsquigarrow$ then so does the system.
   - Otherwise, a counterexample is used to refine the abstraction and try again.

[6] Clarke et al., "Counterexample-guided abstraction refinement for symbolic model checking"; Hermanns, Wachter, and Zhang, "Probabilistic CEGAR".

[7] Ceska et al., "Counterexample-guided inductive synthesis for probabilistic systems".

Exploitation of counterexamples

Two approaches that exploit the information given by counterexamples:

1. Counterexample-guided abstraction refinement (CEGAR)[6] : verification of large systems through smaller abstractions.
   - If the abstraction satisfies the specification $\rightsquigarrow$ then so does the system.
   - Otherwise, a counterexample is used to refine the abstraction and try again.
2. Counterexample-guided inductive synthesis (CEGIS)[7]: find a suitable controller when the answer to the synthesis problem is positive.
   - Generates a controller, if it turns out to be inadequate $\rightsquigarrow$ uses counterexamples to generate a new controller and try again.

[6] Clarke et al., "Counterexample-guided abstraction refinement for symbolic model checking"; Hermanns, Wachter, and Zhang, "Probabilistic CEGAR".

[7] Ceska et al., "Counterexample-guided inductive synthesis for probabilistic systems".

Exploitation of counterexamples

Two approaches that exploit the information given by counterexamples:

1. Counterexample-guided abstraction refinement (CEGAR)[6] : verification of large systems through smaller abstractions.
   - If the abstraction satisfies the specification $\rightsquigarrow$ then so does the system.
   - Otherwise, a counterexample is used to refine the abstraction and try again.
2. Counterexample-guided inductive synthesis (CEGIS)[7]: find a suitable controller when the answer to the synthesis problem is positive.
   - Generates a controller, if it turns out to be inadequate $\rightsquigarrow$ uses counterexamples to generate a new controller and try again.

$\rightsquigarrow$ It does not help when no suitable controller exists.

[6] Clarke et al., "Counterexample-guided abstraction refinement for symbolic model checking"; Hermanns, Wachter, and Zhang, "Probabilistic CEGAR".

[7] Ceska et al., "Counterexample-guided inductive synthesis for probabilistic systems".

Conclusion

- Overview of existing works on the generation of counterexamples for verification stochastic models;
- Introduction of a new notion of counterexamples for synthesis of MDPs.

Ongoing work

- Assess the performance of our method in practice;
- Use the new notion of counterexamples introduced in this talk for when the synthesis process fails.

Thank you for your attention!

Bibliography I

Ábrahám, Erika et al. "Counterexample Generation for Discrete-Time Markov Models: An Introductory Survey". In: *Formal Methods for Executable Software Models - 14th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2014, Bertinoro, Italy, June 16-20, 2014, Advanced Lectures*. Ed. by Marco Bernardo et al. Vol. 8483. Lecture Notes in Computer Science. Springer, 2014, pp. 65–121. DOI: 10.1007/978-3-319-07317-0_3. URL: https://doi.org/10.1007/978-3-319-07317-0_3.

Ceska, Milan et al. "Counterexample-guided inductive synthesis for probabilistic systems". In: *Formal Aspects Comput.* 33.4-5 (2021), pp. 637–667. DOI: 10.1007/s00165-021-00547-2. URL: https://doi.org/10.1007/s00165-021-00547-2.

Bibliography II

- Clarke, Edmund M. et al. "Counterexample-guided abstraction refinement for symbolic model checking". In: *J. ACM* 50.5 (2003), pp. 752–794. DOI: 10.1145/876638.876643. URL: https://doi.org/10.1145/876638.876643.
- Hermanns, Holger, Björn Wachter, and Lijun Zhang. "Probabilistic CEGAR". In: *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, NJ, USA, July 7-14, 2008, Proceedings*. Ed. by Aarti Gupta and Sharad Malik. Vol. 5123. Lecture Notes in Computer Science. Springer, 2008, pp. 162–175. DOI: 10.1007/978-3-540-70545-1_16. URL: https://doi.org/10.1007/978-3-540-70545-1_16.

Bibliography III

Quatmann, Tim et al. "Counterexamples for Expected Rewards". In: *FM 2015: Formal Methods - 20th International Symposium, Oslo, Norway, June 24-26, 2015, Proceedings*. Ed. by Nikolaj S. Bjørner and Frank S. de Boer. Vol. 9109. Lecture Notes in Computer Science. Springer, 2015, pp. 435–452. DOI: 10.1007/978-3-319-19249-9_27. URL: https://doi.org/10.1007/978-3-319-19249-9_27.

Wimmer, Ralf et al. "Minimal counterexamples for linear-time probabilistic verification". In: *Theor. Comput. Sci.* 549 (2014), pp. 61–100. DOI: 10.1016/j.tcs.2014.06.020. URL: https://doi.org/10.1016/j.tcs.2014.06.020.