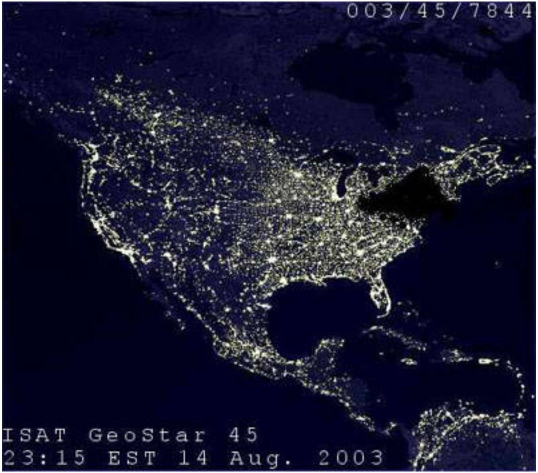
Refined Probabilistic Abstraction

Björn Wachter

UNIVERSITÄT DES SAARLANDES

December 8, 2010



- Bug in control software of power network

$\Rightarrow$ 50 million people without electricity

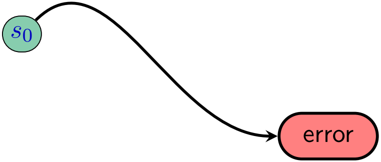
Model checking:

"does a computing system behave as intended?"

- mathematical model M of system
- specification φ
- automatic proof or refutation of:

$$M \models \varphi$$

- Example: $\varphi =$ no arithmetic overflow



e.g., arithmetic overflow

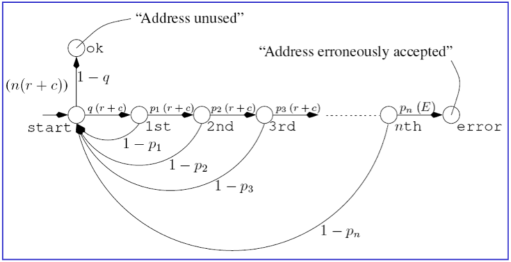
Probabilities are Important

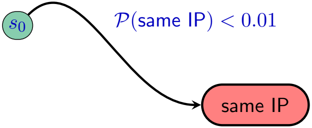
- computer networks
  - performance: P(message loss) $= 2\%$
  - reliability: P(node failure) $= 3\%$
- randomized algorithms
  - network protocols
  - sorting algorithms
  - ...

Probabilistic Model checking

- models: Markov chains
- properties: PCTL

Probabilistic Model checking

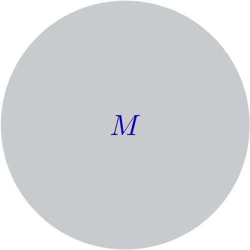




- models: Markov chains
- properties: PCTL
- Zeroconf protocol
- IP for new member picked probabilistically
- bad: two members have the same IP!

Why Abstraction?

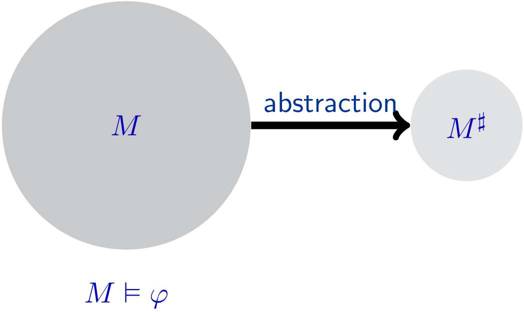
- Limitations of probabilistic model checking
  - based on state-space exploration
  - state-space explosion problem



$M \models \varphi$

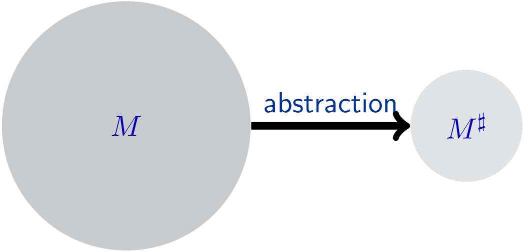
Why Abstraction?

- Limitations of probabilistic model checking
  - based on state-space exploration
  - state-space explosion problem
- Abstraction very successful



Why Abstraction?

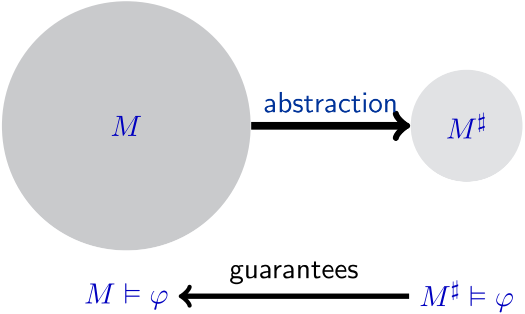
- Limitations of probabilistic model checking
  - based on state-space exploration
  - state-space explosion problem
- Abstraction very successful



$M \models \varphi$ $M^\sharp \models \varphi$

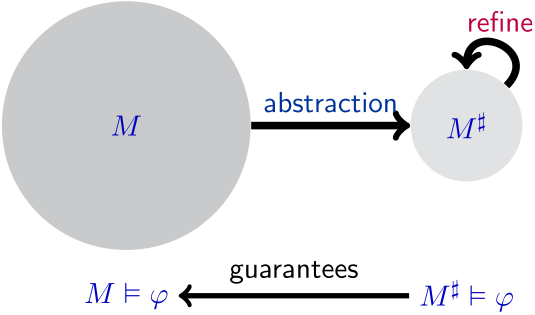
Why Abstraction?

- Limitations of probabilistic model checking
  - based on state-space exploration
  - state-space explosion problem
- Abstraction very successful



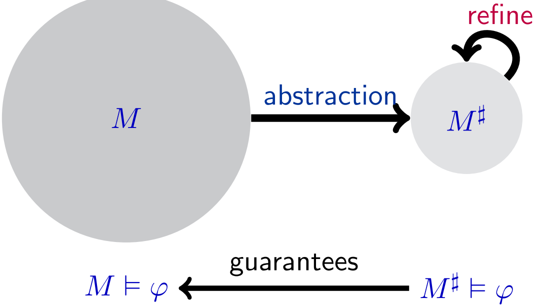
Why Abstraction?

- Limitations of probabilistic model checking
  - based on state-space exploration
  - state-space explosion problem
- Abstraction very successful



Why Abstraction?

- Limitations of probabilistic model checking
  - based on state-space exploration
  - state-space explosion problem
- Abstraction very successful
  - Refinement fits the abstraction to the property



Contribution

Abstraction refinement for very large probabilistic models

- ... even infinite ones!

- implementation in PASS tool
- successful on various network protocols
  - Wireless LAN
  - IPV4
  - BRP
  - ...

Background

Probabilistic Programs

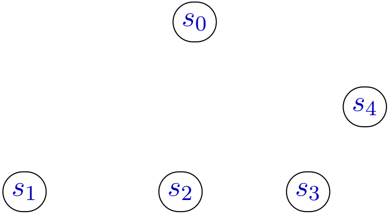
```
// parallel composition of modules
module sender
i : int; // variable definition
...
endmodule

module channelK
[aF] (k=0) -> 0.98 : (k'=1)  // probabilistic
            + 0.02 : (k'=2); // guarded command
endmodule

init T=false & ... endinit // initial states
```

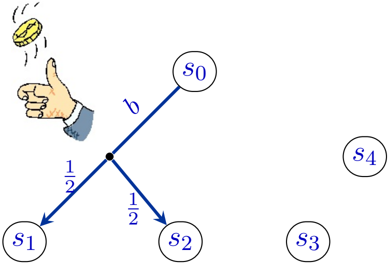
Semantics: **M**arkov **D**ecision **P**rocess (MDP)

states S $\cong$ assignments to program variables



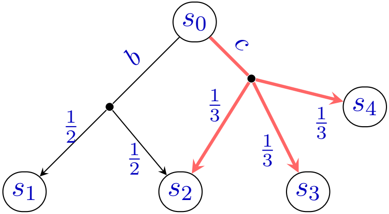
Semantics: **M**arkov **D**ecision **P**rocess (MDP)

states S	$\cong$	assignments to program variables
probabilistic transitions	$\dots$	probabilistic choice



Semantics: **M**arkov **D**ecision **P**rocess (MDP)

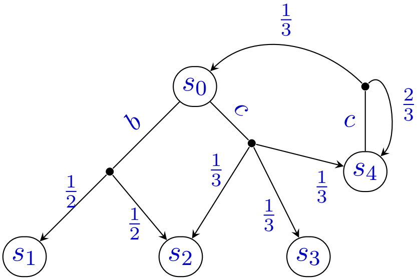
states S	$\cong$	assignments to program variables
probabilistic transitions	...	probabilistic choice
non-deterministic choice	...	concurreny



- Markov chain $\cong$ deterministic MDP

Semantics: **M**arkov **D**ecision **P**rocess (MDP)

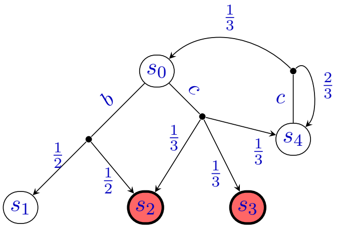
states S	$\cong$	assignments to program variables
probabilistic transitions	...	probabilistic choice
non-deterministic choice	...	concurreny



- Markov chain $\cong$ deterministic MDP

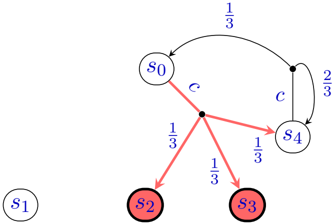
Properties: Probabilistic Reachability

- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



Properties: Probabilistic Reachability

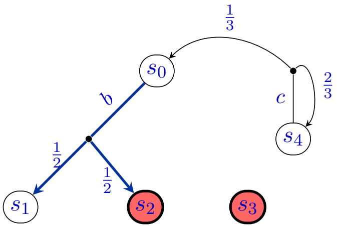
- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



reachability probability: 1

Properties: Probabilistic Reachability

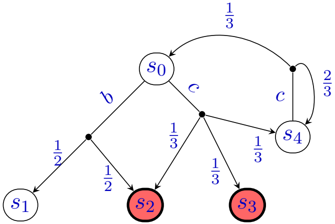
- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



reachability probability: $\frac{1}{2}$

Properties: Probabilistic Reachability

- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



- reachability probability p_F^η
  - depends on adversary η

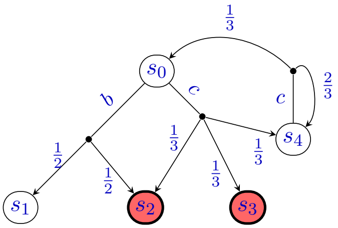
adversary $\eta : Paths \to A$

- resolves non-determinism

$\Rightarrow$ induces a Markov chain

Properties: Probabilistic Reachability

- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



adversary $\eta : Paths \to A$

- resolves non-determinism

$\Rightarrow$ induces a Markov chain

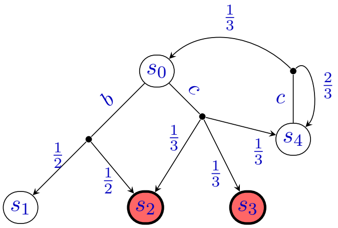
- reachability probability p_F^η
  - depends on adversary η
- minimal/maximal

$$p^{min}{}_F \quad = \quad \inf_\eta \quad p_F^\eta$$

$$p^{max}{}_F \quad = \quad \sup_\eta \quad p_F^\eta$$

Properties: Probabilistic Reachability

- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



adversary $\eta : Paths \to A$

- resolves non-determinism

$\Rightarrow$ induces a Markov chain

- reachability probability p_F^η
  - depends on adversary η
- minimal/maximal

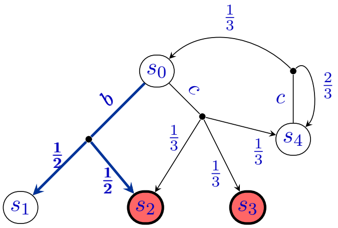
$$p^{min}{}_F = \inf_\eta p_F^\eta$$
$$p^{max}{}_F = \sup_\eta p_F^\eta$$

least fixpoint of $Pre^{min} : [0,1]^S \to [0,1]^S$

$$w \mapsto \lambda s. \begin{cases} 1 & ; s \in F \\ 0 & ; s \in F_0 \\ \min\limits_{a \in A(s)} \sum\limits_{(u,t) \in U \times S} R(s,u,t) \cdot w(t) & ;\ \text{ow.} \end{cases}$$

Properties: Probabilistic Reachability

- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



adversary $\eta : Paths \to A$

- resolves non-determinism

$\Rightarrow$ induces a Markov chain

- reachability probability p_F^η
  - depends on adversary η
- minimal/maximal

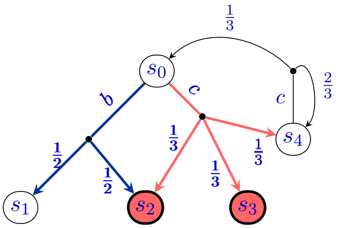
$$p^{min}{}_F = \inf_\eta \; p_F^\eta$$
$$p^{max}{}_F = \sup_\eta \; p_F^\eta$$

least fixpoint of $Pre^{min} : [0,1]^S \to [0,1]^S$

$$w \mapsto \lambda s. \begin{cases} 1 & ; s \in F \\ 0 & ; s \in F_0 \\ \min_{a \in A(s)} \sum_{(u,t) \in U \times S} R(s,u,t) \cdot w(t) & ; \text{ ow.} \end{cases}$$

Properties: Probabilistic Reachability

- probabilities to reach states $F \subseteq S$
- valuations $[0,1]^S \quad \cong \quad S \to [0,1]$



adversary $\eta : Paths \to A$

- resolves non-determinism

$\Rightarrow$ induces a Markov chain

- reachability probability p_F^η
  - depends on adversary η
- minimal/maximal

$$p^{min}{}_F = \inf_\eta p_F^\eta$$
$$p^{max}{}_F = \sup_\eta p_F^\eta$$

least fixpoint of $Pre^{max} : [0,1]^S \to [0,1]^S$

$$w \mapsto \lambda s. \begin{cases} 1 & ; s \in F \\ 0 & ; s \in F_0 \\ \max\limits_{a \in A(s)} \sum\limits_{(u,t) \in U \times S} R(s,u,t) \cdot w(t) & ; \text{ ow.} \end{cases}$$

Abstraction

Abstraction for Probabilistic Reachability

- Problem: many states S

reachability probability

1.0	0.5	1.0	1.0
0.2	0.1	1.0	1.0
0.3	0.7	0.4	0.1
0.0	0.8	0.3	0.2

in general, hard to compute

Abstraction for Probabilistic Reachability

- Problem: many states S

❶ merge states to blocks Q

reachability probability

in general, hard to compute

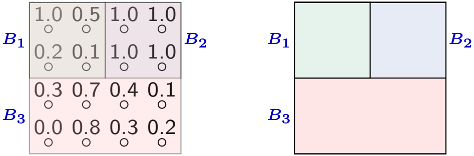
Abstraction for Probabilistic Reachability

- Problem: many states S

❶ merge states to blocks Q

  - in example, **16** states but only **3** blocks $Q = \{B_1, B_2, B_3\}$.

reachability probability



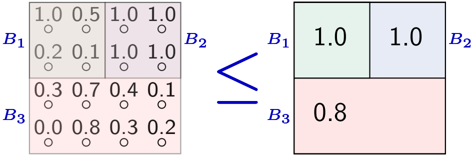
in general, hard to compute

Abstraction for Probabilistic Reachability

- Problem: many states S

1. merge states to blocks Q
   - in example, **16** states but only **3** blocks $Q = \{B_1, B_2, B_3\}$.
2. compute abstract valuations $[0, 1]^Q$

reachability probability



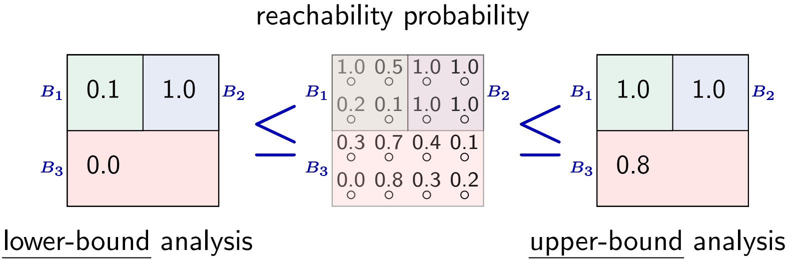
Abstraction for Probabilistic Reachability

- Problem: many states S

❶ merge states to blocks Q

- in example, **16** states but only **3** blocks $Q = \{B_1, B_2, B_3\}$.

❷ compute abstract valuations $[0,1]^Q$



Challenge of Analysis Design

complex interplay

uncertainty from abstraction

probabilism — concurrency

- Open Question:

  what does an optimal analysis look like?

Challenge of Analysis Design

complex interplay

uncertainty from abstraction

probabilism —— concurrency

- Open Question:

  what does an optimal analysis look like?

- Our solution:

  - Recipe: Abstract Interpretation [Cousot77]

Challenge of Analysis Design

uncertainty from abstraction

complex interplay

probabilism —— concurrency

- Open Question:

  what does an optimal analysis look like?

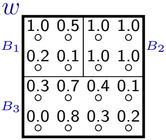
- Our solution:

- Recipe: Abstract Interpretation [Cousot77]
- Ingredients:
  - abstraction functions

Abstraction & concretization

1. abstraction functions:
   - mappings
     $[0,1]^S \mapsto [0,1]^Q$

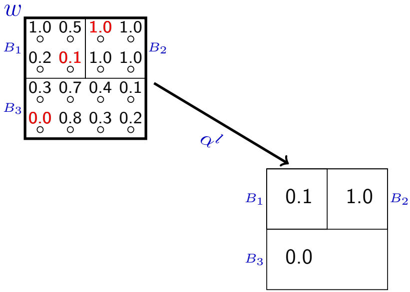
$[0,1]^S$ $[0,1]^Q$



Abstraction & concretization

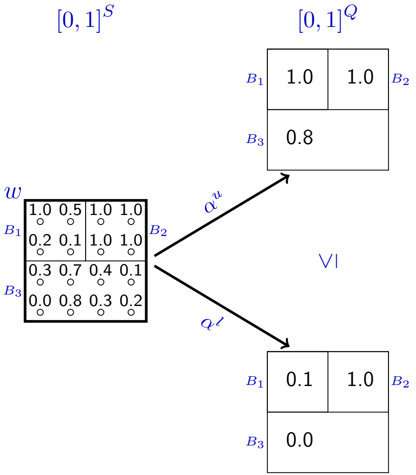
1. abstraction functions:
   - mappings $[0,1]^S \mapsto [0,1]^Q$
   - lower bound: $\alpha^l(w) = \lambda B.\ \inf_{s \in B} w(s)$

$[0,1]^S$ $[0,1]^Q$



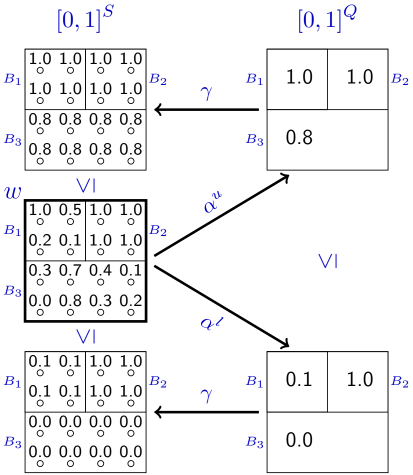
Abstraction & concretization

1. abstraction functions:
   - mappings
     $[0,1]^S \mapsto [0,1]^Q$
   - lower bound:
     $\alpha^l(w) = \lambda B.\ \inf_{s \in B} w(s)$
   - upper bound:
     $\alpha^u(w) = \lambda B.\ \sup_{s \in B} w(s)$



Abstraction & concretization

1. abstraction functions:
   - mappings
     $[0,1]^S \mapsto [0,1]^Q$
   - lower bound:
     $\alpha^l(w) = \lambda B.\ \inf_{s \in B} w(s)$
   - upper bound:
     $\alpha^u(w) = \lambda B.\ \sup_{s \in B} w(s)$
2. concretiztion function
   (or meaning function)
   - $[0,1]^Q \mapsto [0,1]^S$
   - $\gamma(w^\sharp) = \lambda s.\ w^\sharp([s])$.



How do we get an Abstract Analysis?

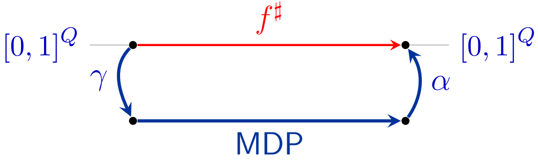
- abstract analysis
  - function $f^\sharp : [0,1]^Q \to [0,1]^Q$

  $\Rightarrow$ lower/upper bound = fixpoint of $f^\sharp$
- Best-transformer paradigm [Cousot 2002]

How do we get an Abstract Analysis?

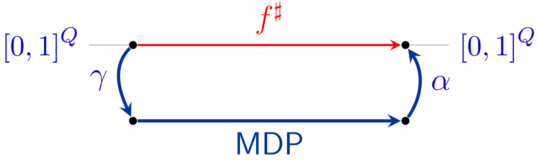
- abstract analysis
  - function $f^\sharp : [0,1]^Q \to [0,1]^Q$

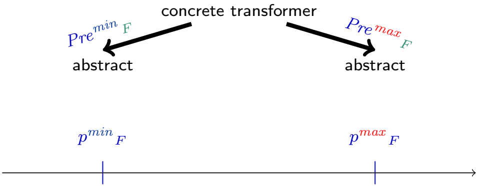
  $\Rightarrow$ lower/upper bound = fixpoint of $f^\sharp$
- Best-transformer paradigm [Cousot 2002]



How do we get an Abstract Analysis?

- abstract analysis
  - function $f^\sharp : [0,1]^Q \to [0,1]^Q$
  - $\Rightarrow$ lower/upper bound = fixpoint of $f^\sharp$
- Best-transformer paradigm [Cousot 2002]

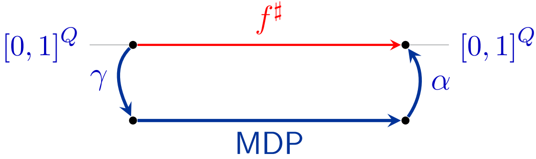


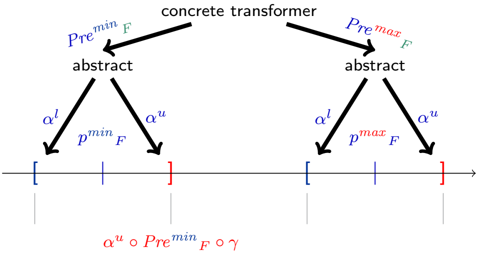


How do we get an Abstract Analysis?

- abstract analysis
  - function $f^\sharp : [0,1]^Q \to [0,1]^Q$

  $\Rightarrow$ lower/upper bound = fixpoint of $f^\sharp$
- Best-transformer paradigm [Cousot 2002]



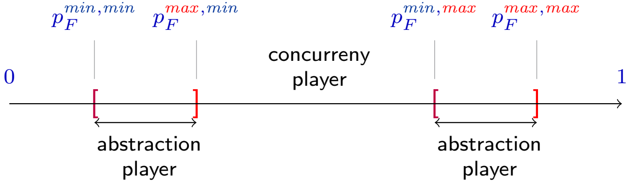


Abstract Transformers = Stochastic Games

$$(\alpha^u \circ Pre_F^{min} \circ \gamma(w^\sharp))(B) = \sup_{s \in B} Pre_F^{min}(\gamma(w^\sharp))(s)$$

$$= \sup_{s \in B} \min_{a \in A(s)} \sum_{s' \in S} R(s,a)(s') \cdot (\gamma(w^\sharp))(s')$$

- Markov models
  - Markov chain = $\frac{1}{2}$ player
  - MDP = $1\frac{1}{2}$ player
  - stochastic game = $2\frac{1}{2}$ player
- minimum / maximum over all strategies *for both players*

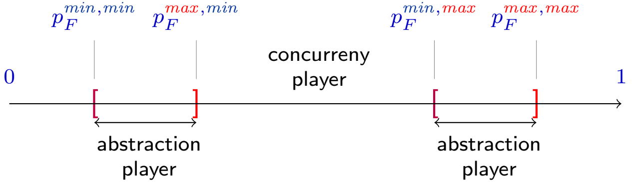


Abstract Transformers = Stochastic Games

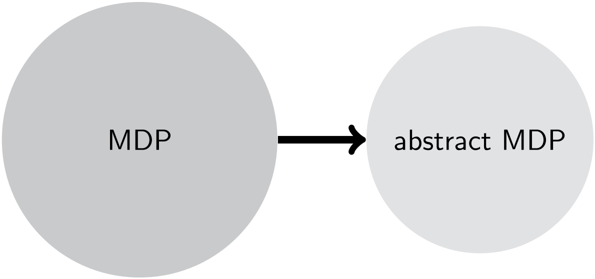
$$(\alpha^u \circ Pre_F^{min} \circ \gamma(w^\sharp))(B) = \sup_{s \in B} Pre_F^{min}(\gamma(w^\sharp))(s)$$

$$= \sup_{s \in B} \min_{a \in A(s)} \sum_{s' \in S} R(s,a)(s') \cdot (\gamma(w^\sharp))(s')$$

- Markov models
  - Markov chain = $\frac{1}{2}$ player
  - MDP = $1\frac{1}{2}$ player
  - stochastic game = $2\frac{1}{2}$ player
- minimum / maximum over all strategies *for both players*

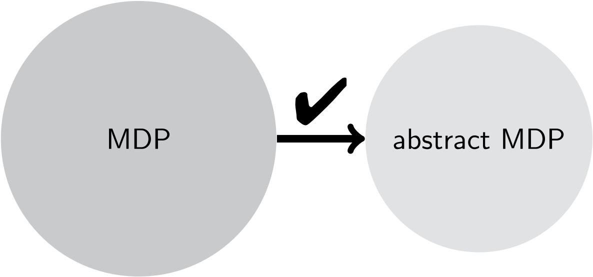


Wrap-up



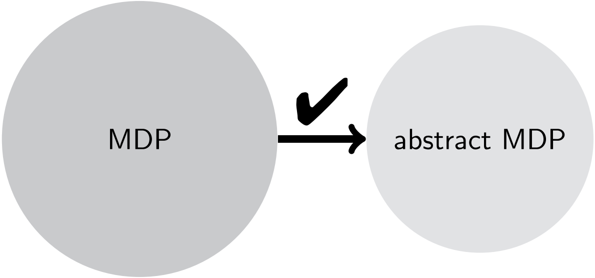
reachability probability

Wrap-up



reachability probability

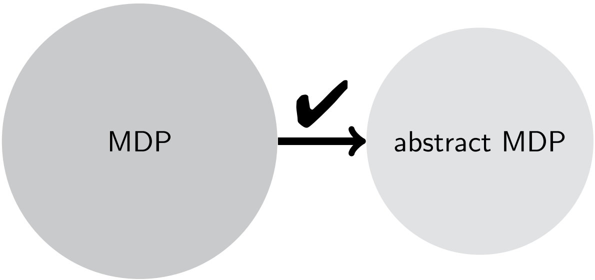
Wrap-up



reachability probability

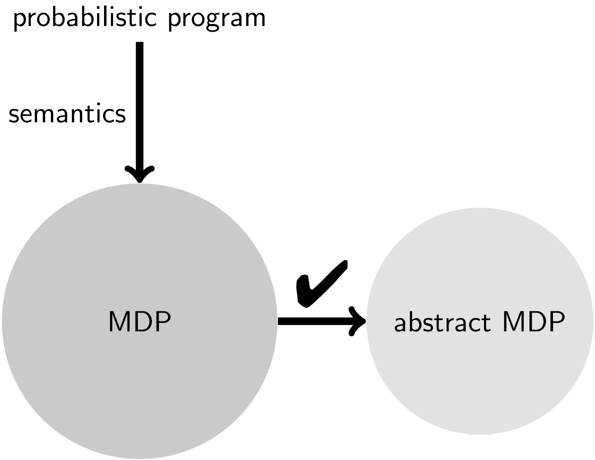
$[lb, ub]$

Wrap-up



reachability probability $\in$ $[lb, ub]$

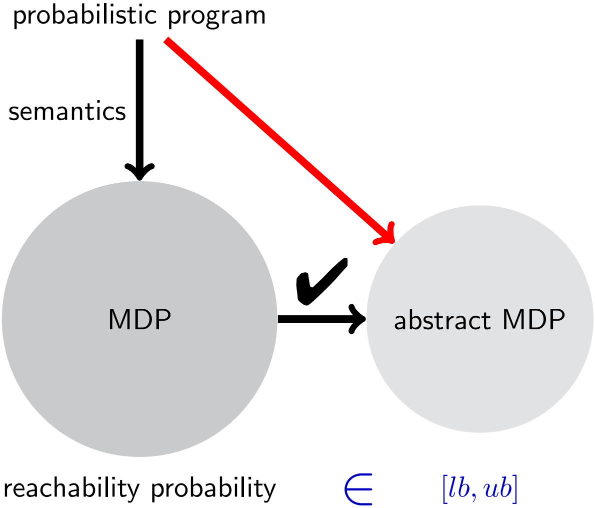
Wrap-up



reachability probability $\in$ $[lb, ub]$

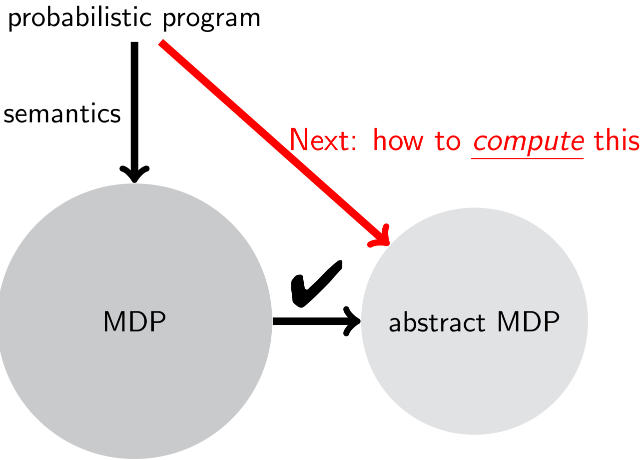
Wrap-up

- diagram *defines* an abstract semantics (mathematics)



Wrap-up

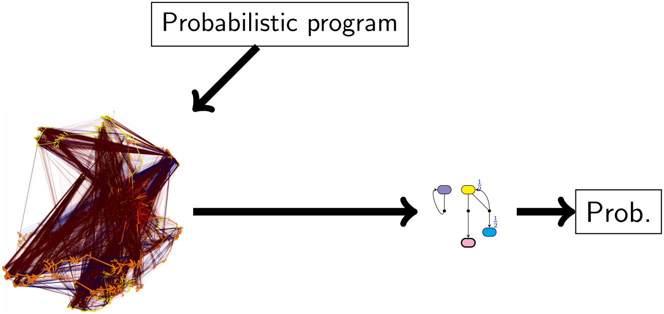
- diagram *defines* an abstract semantics (mathematics)



reachability probability $\in$ $[lb, ub]$

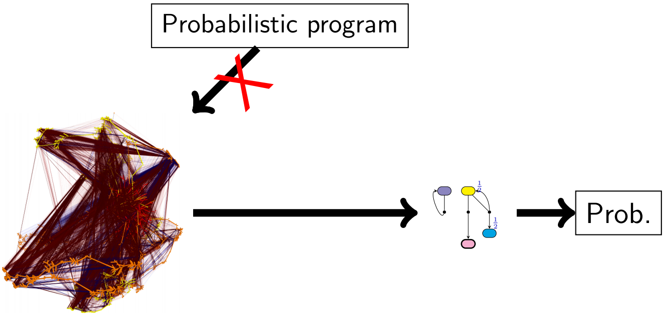
State of the Art before Thesis

- de Alfaro, Roy. CAV 2007
  Magnifying-Lens Abstraction for Markov Decision Processes..
- Chatterjee, Henzinger, Jhala, Majumdar. UAI 2005
  Counterexample Guided Planning.
- D'Argenio, Jeannet, Jensen, Larsen. PAPM-PROBMIV 2002
  Reduction and Refinement Strategies for Probabilistic Analysis.
- ... build full semantics



State of the Art before Thesis

- de Alfaro, Roy. CAV 2007
  Magnifying-Lens Abstraction for Markov Decision Processes..
- Chatterjee, Henzinger, Jhala, Majumdar. UAI 2005
  Counterexample Guided Planning.
- D'Argenio, Jeannet, Jensen, Larsen. PAPM-PROBMIV 2002
  Reduction and Refinement Strategies for Probabilistic Analysis.
- ... build full semantics
  ⇒ expensive or even impossible

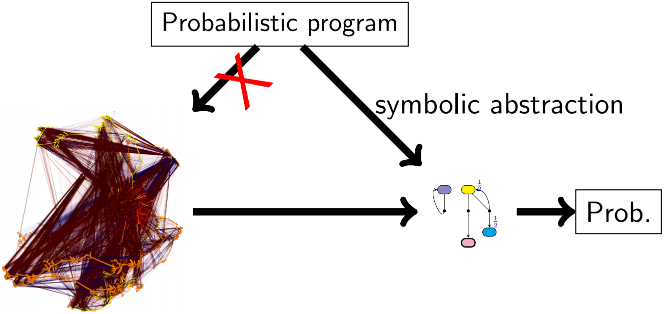


State of the Art before Thesis

- de Alfaro, Roy. CAV 2007
  Magnifying-Lens Abstraction for Markov Decision Processes..
- Chatterjee, Henzinger, Jhala, Majumdar. UAI 2005
  Counterexample Guided Planning.
- D'Argenio, Jeannet, Jensen, Larsen. PAPM-PROBMIV 2002
  Reduction and Refinement Strategies for Probabilistic Analysis.
- ... build full semantics
  ⇒ expensive or even impossible

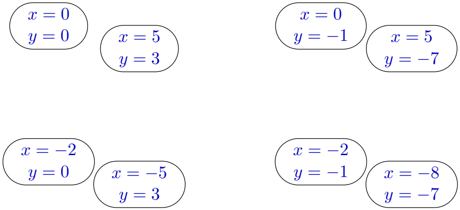
Premiere: 1st symbolic abstraction for probabilistic programs

- abstraction at the language level



Predicate Abstraction

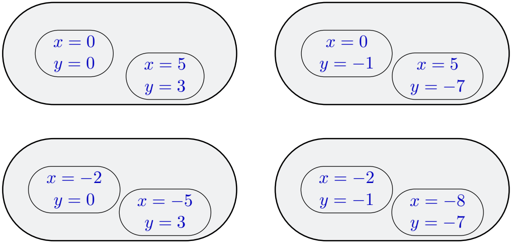
- Predicates ≅ expressions over program variables
  - e.g., $x > 0$, $x < y$



Predicate Abstraction

- Predicates $\cong$ expressions over program variables
  - e.g., $x > 0$, $x < y$

$\overset{\text{define}}{\Rightarrow}$ partition

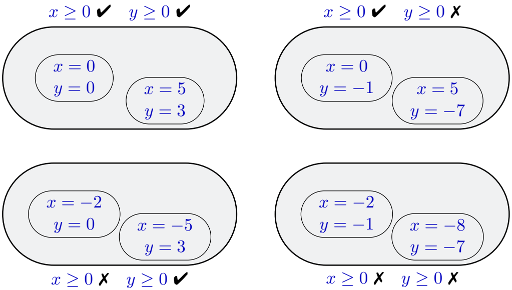


Predicate Abstraction

- Predicates $\cong$ expressions over program variables
  - e.g., $x > 0$, $x < y$

$\stackrel{\text{define}}{\Rightarrow}$ partition

- Blocks



Predicate Abstraction

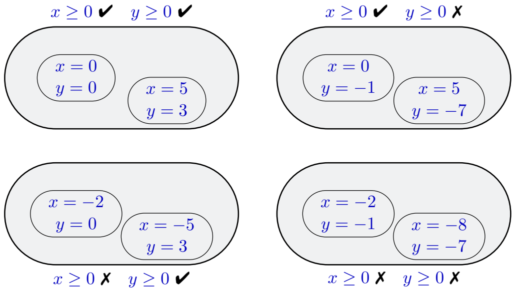
- Predicates $\cong$ expressions over program variables
  - e.g., $x > 0$, $x < y$

$\overset{\text{define}}{\Rightarrow}$ partition

- Blocks

$\overset{\text{define}}{\Rightarrow}$ abstract model

- stochastic game



Predicate Abstraction

- Predicates $\cong$ expressions over program variables
  - e.g., $x > 0$, $x < y$

$\overset{\text{define}}{\Rightarrow}$ partition
  - Blocks

$\overset{\text{define}}{\Rightarrow}$ abstract model
  - stochastic game

- reduce abstraction to satisfiability of logical formulas

$\Rightarrow$ implemented by SMT solver

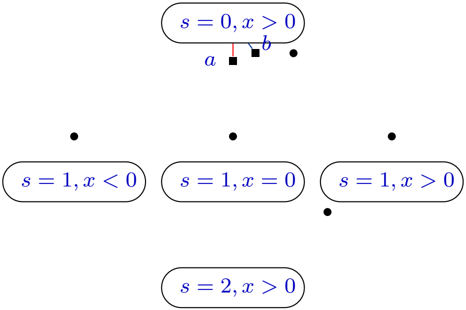
- SMT = Satisfiability Modulo Theories

Example

- Consider program

```
module main
  s   : [0..2];    // control flow
  x,y : int;       // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s = 0$,$s = 1$,$s = 2$,$x = 0$,$x > 0$,$x < 0$

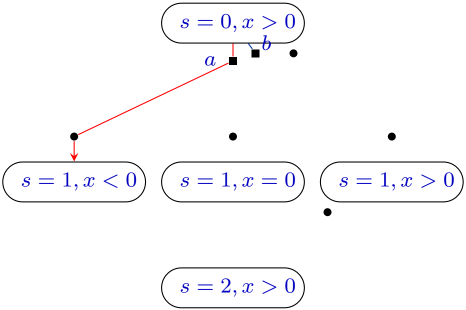


Example

- Consider program

```
module main
  s   : [0..2];     // control flow
  x,y : int;        // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s = 0$,$s = 1$,$s = 2$,$x = 0$,$x > 0$,$x < 0$

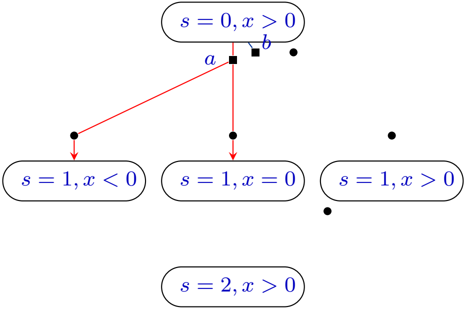


Example

- Consider program

```
module main
  s    : [0..2];    // control flow
  x,y  : int;       // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s = 0$,$s = 1$,$s = 2$,$x = 0$,$x > 0$,$x < 0$

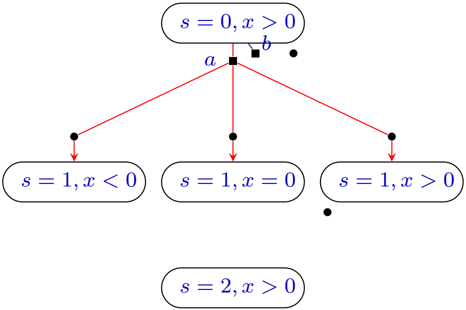


Example

- Consider program

```
module main
  s   : [0..2];    // control flow
  x,y : int;       // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s = 0$,$s = 1$,$s = 2$,$x = 0$,$x > 0$,$x < 0$

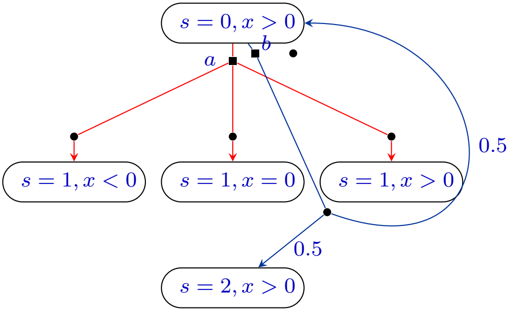


Example

- Consider program

```
module main
  s   : [0..2];     // control flow
  x,y : int;        // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s = 0$,$s = 1$,$s = 2$,$x = 0$,$x > 0$,$x < 0$

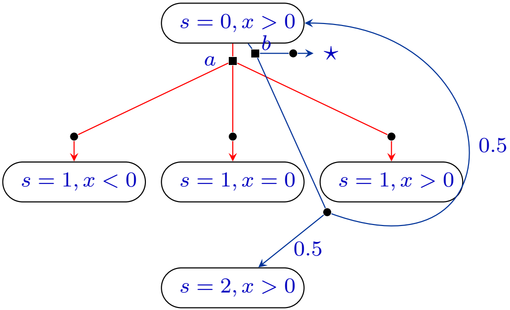


Example

- Consider program

```
module main
  s   : [0..2];     // control flow
  x,y : int;        // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s = 0$,$s = 1$,$s = 2$,$x = 0$,$x > 0$,$x < 0$

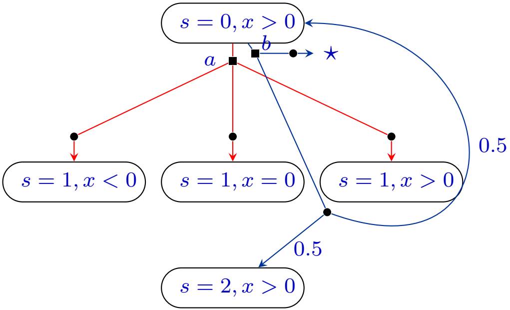


Example

- Consider program

```
module main
  s   : [0..2];    // control flow
  x,y : int;       // integer variables
  [a] s=0 -> 1.0:(s'=1) & (x'=y);
  [b] s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s=0$,$s=1$,$s=2$,$x=0$,$x>0$,$x<0$

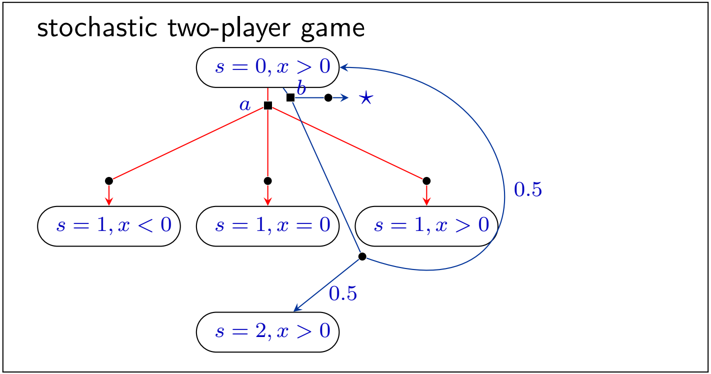


Example

- Consider program

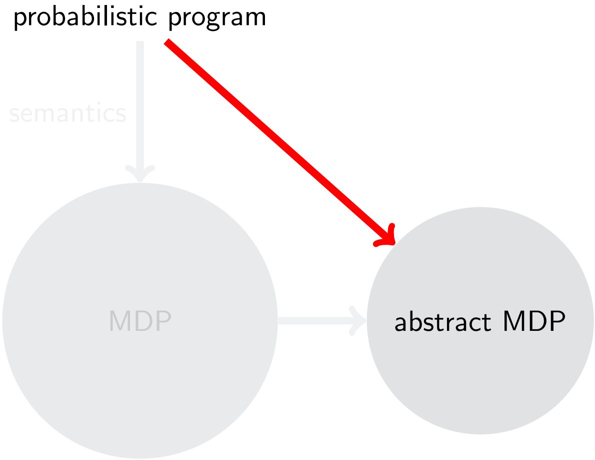
```
module main
  s   : [0..2];    // control flow
  x,y : int;       // integer variables
   s=0 -> 1.0:(s'=1) & (x'=y);
   s=0 & x>10 -> 0.5:(s'=0)+ 0.5:(s'=2);
endmodule
```

- predicates $s=0$,$s=1$,$s=2$,$x=0$,$x>0$,$x<0$



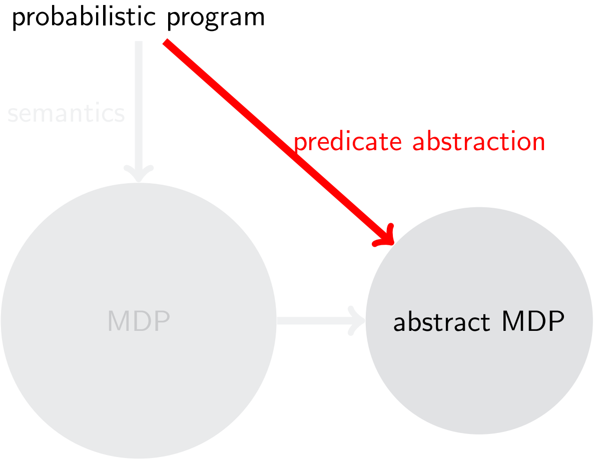
Wrap-up

- fully automatic and symbolic abstraction
  - for given **predicate** set



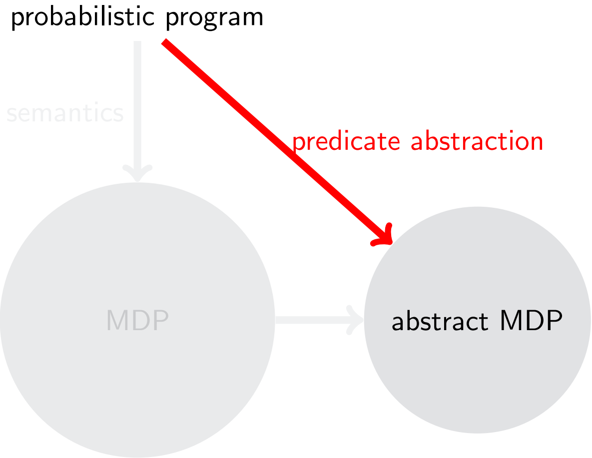
Wrap-up

- fully automatic and symbolic abstraction
  - for given **predicate** set

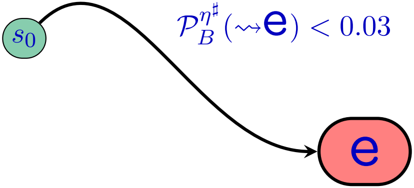


Wrap-up

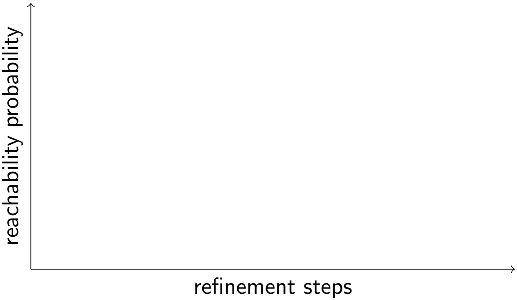
- fully automatic and symbolic abstraction
  - for given **predicate** set
- ... but where do predicates come from?



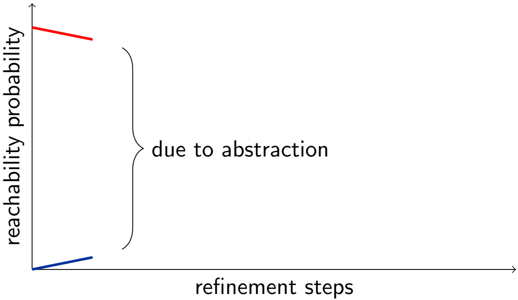
Reachability Properties



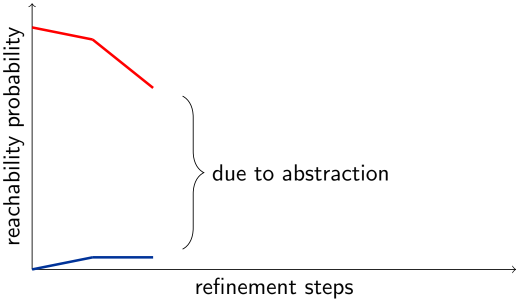
Abstraction Refinement



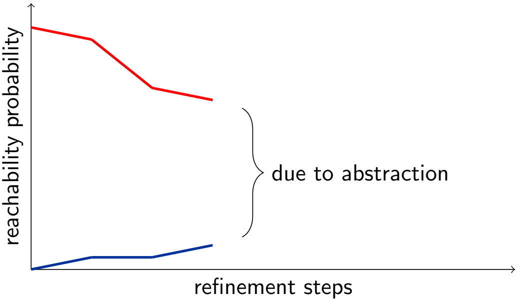
Abstraction Refinement



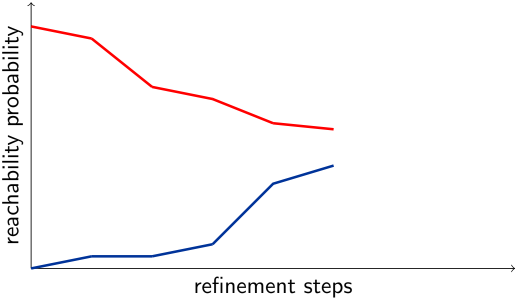
Abstraction Refinement



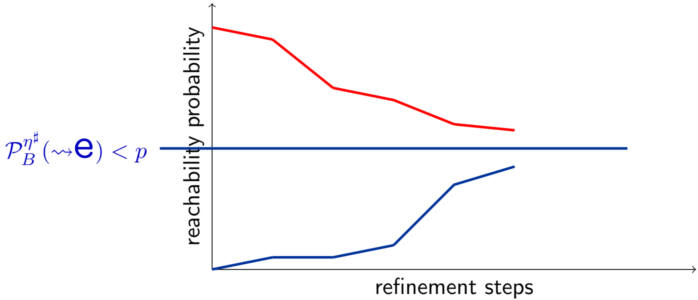
Abstraction Refinement



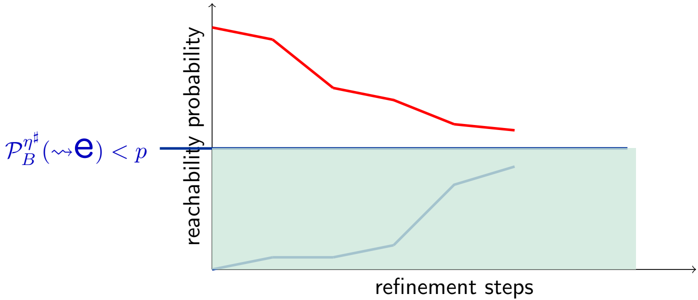
Abstraction Refinement



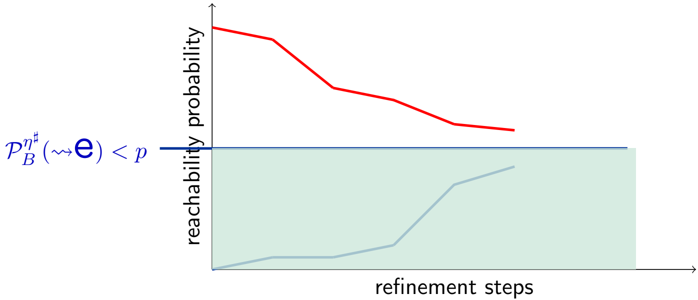
Abstraction Refinement



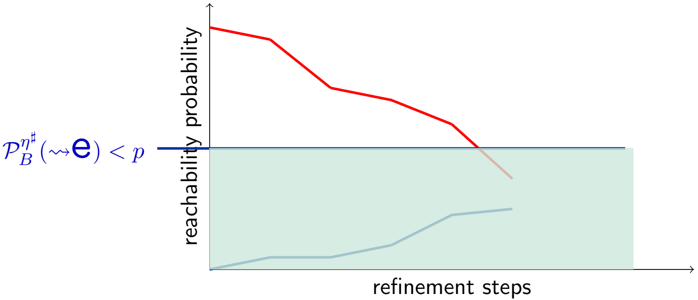
Abstraction Refinement



Abstraction Refinement

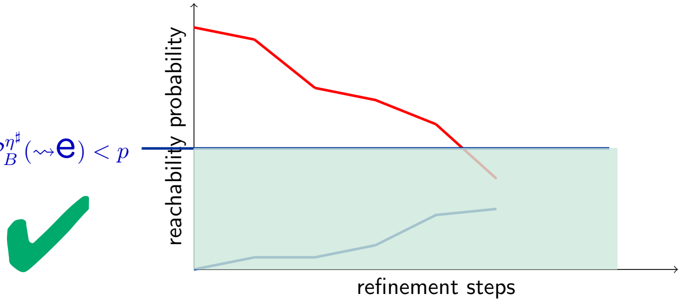


Property shown

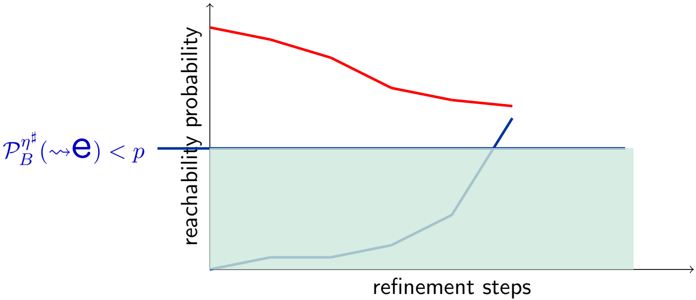


Property shown

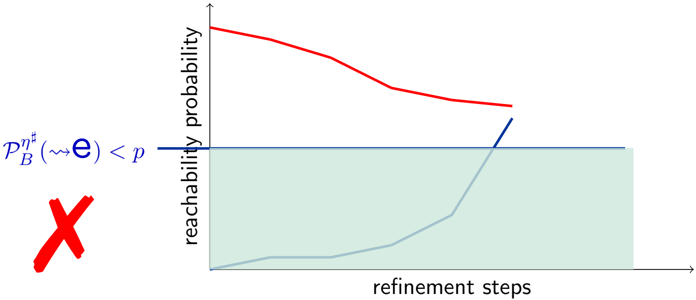
$\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e}) < p$



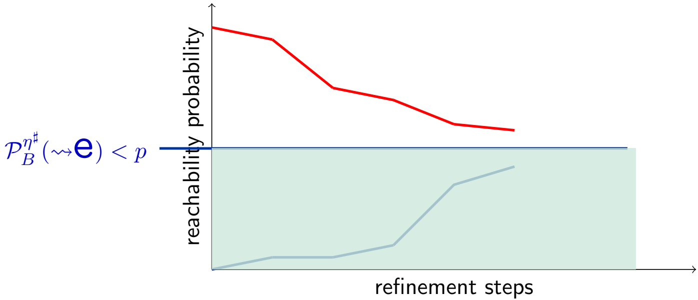
Property refuted



Property refuted



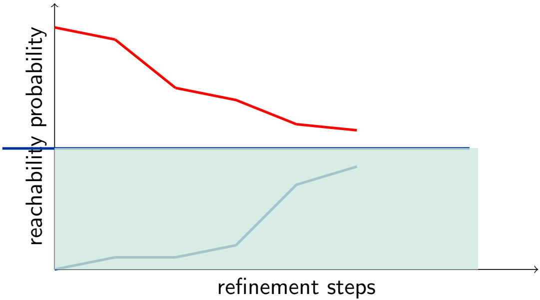
Inconclusive: refinement due



Inconclusive: refinement due

$\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e}) < p$

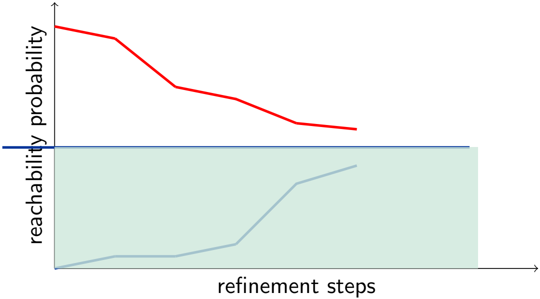
?



Inconclusive: refinement due

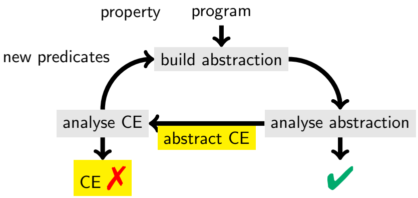
$\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e}) < p$

?



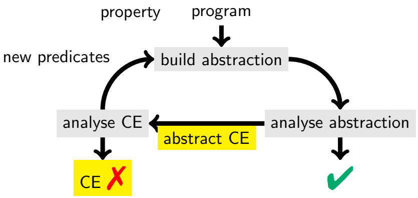
CEGAR: Counterexample-Guided Abstraction Refinement

- refinement technique in software model checking
  - SLAM project at Microsoft [Ball/Rajamani 2002,...]
  - Blast
  - ...



CEGAR: Counterexample-Guided Abstraction Refinement

- refinement technique in software model checking
  - SLAM project at Microsoft [Ball/Rajamani 2002,...]
  - Blast
  - ...



What **is** an abstract CE?

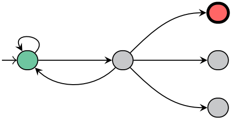
- pioneering work in probabilistic verification

Counterexamples

Safety: Error State Unreachable

Probabilistic Reachability

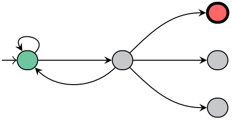
Transition System



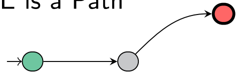
Counterexamples

Safety: Error State Unreachable

Transition System



CE is a Path



"error state is reachable"

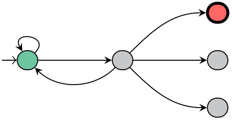
Probabilistic Reachability

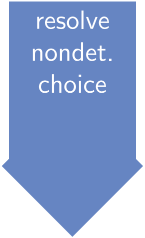
Counterexamples

Safety: Error State Unreachable

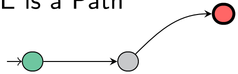
Probabilistic Reachability

Transition System





CE is a Path

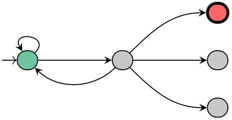


"error state is reachable"

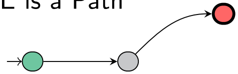
Counterexamples

Safety: Error State Unreachable

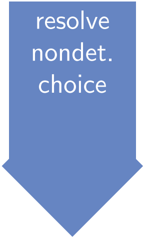
Transition System



CE is a Path

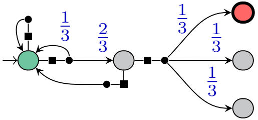


“error state is reachable”



Probabilistic Reachability

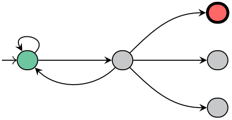
Stochastic Game



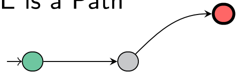
Counterexamples

Safety: Error State Unreachable

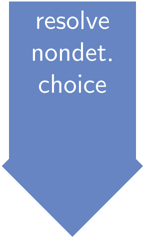
Transition System



CE is a Path

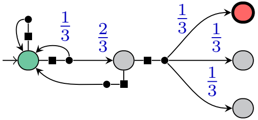


"error state is reachable"

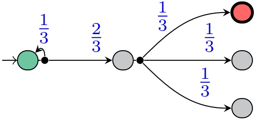


Probabilistic Reachability

Stochastic Game



Markov Chain

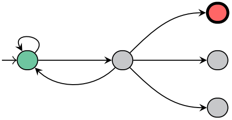


"probability to reach error state $= \frac{1}{3}$"

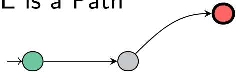
Counterexamples

Safety: Error State Unreachable

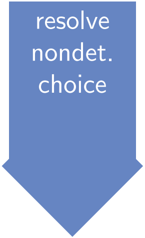
Transition System



CE is a Path

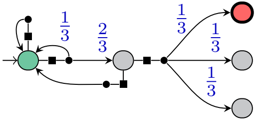


"error state is reachable"

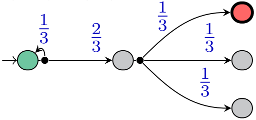


Probabilistic Reachability

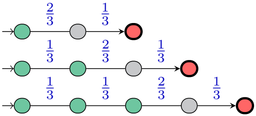
Stochastic Game



Markov Chain



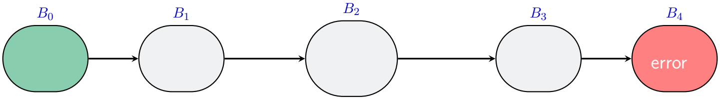
"probability to reach error state $= \frac{1}{3}$"



...

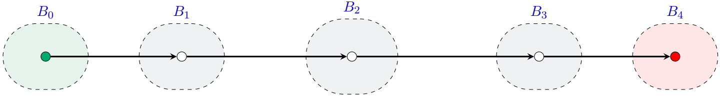
Conventional counterexample analysis

- check if the abstract counterexample is realisable …



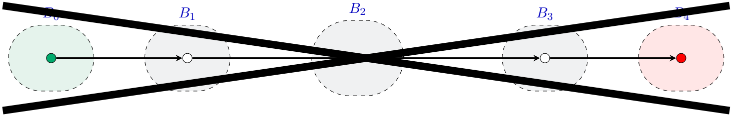
Conventional counterexample analysis

- check if the abstract counterexample is realisable ...
  - if so, we've found a **bug**

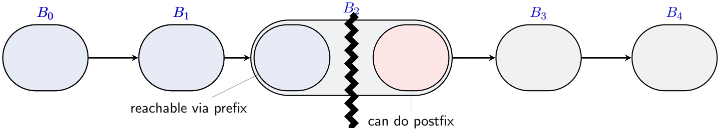


Conventional counterexample analysis

- check if the abstract counterexample is realisable ...
  - if so, we've found a **bug**

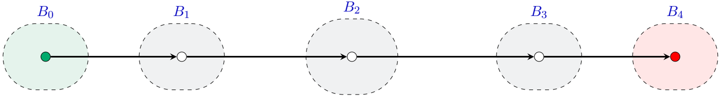


- ... or spurious
  - abstraction too coarse, i.e., needs **refinement**

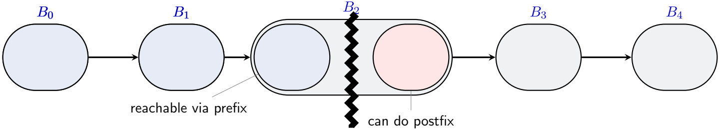


Conventional counterexample analysis

- check if the abstract counterexample is realisable ...
  - if so, we've found a **bug**



- ... or spurious
  - abstraction too coarse, i.e., needs **refinement**

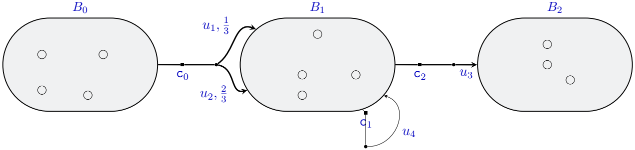


- Implementation with SMT solver:
  - convert path to formula
  - path realisable $\iff$ formula satisfiable
  - generate splitting predicate, e.g., by interpolation

Analysis of Probabilistic CE

- Is there a matching real counterexample?
  - replay decisions of abstract counterexample

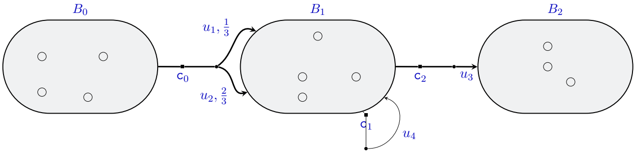
abstract CE



Analysis of Probabilistic CE

- Is there a matching real counterexample?
  - replay decisions of abstract counterexample

abstract CE



- Challenge: CE correspond to many paths
  - Markov chain: cyclic & has probabilistic branching
- Goal 1: Leverage conventional counterexample analysis
  - path analysis based on SMT and interpolation
- Goal 2: avoid exploring too many paths

Probabilistic CEGAR

- enumerate paths of CE Markov chain
- visit paths with highest probability first [Han&Katoen 2007]
  - path $\sigma_1^\sharp$
    - if spurious generate predicate (interpolation)

$\longleftarrow$ abstract probability mass $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathtt{e})$ $\longrightarrow$

$\sigma_1^\sharp$

Probabilistic CEGAR

- enumerate paths of CE Markov chain
- visit paths with highest probability first [Han&Katoen 2007]
  - path $\sigma_1^\sharp$
    - if spurious generate predicate (interpolation)
  - path $\sigma_2^\sharp$

⟵ abstract probability mass $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e})$ ⟶

$\sigma_1^\sharp$	$\sigma_2^\sharp$

Probabilistic CEGAR

- enumerate paths of CE Markov chain
- visit paths with highest probability first [Han&Katoen 2007]
  - path $\sigma_1^\sharp$
    - if spurious generate predicate (interpolation)
  - path $\sigma_2^\sharp$
  - ...

$\longleftarrow$ abstract probability mass $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e})$ $\longrightarrow$

$\sigma_1^\sharp$	$\sigma_2^\sharp$	$\sigma_3^\sharp$

Probabilistic CEGAR

- enumerate paths of CE Markov chain
- visit paths with highest probability first [Han&Katoen 2007]
  - path $\sigma_1^\sharp$
    - if spurious generate predicate (interpolation)
  - path $\sigma_2^\sharp$
  - ...

$\longleftarrow$ abstract probability mass $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e})$ $\longrightarrow$

$\sigma_1^\sharp$	$\sigma_2^\sharp$	$\sigma_3^\sharp$	$\sigma_4^\sharp$

Probabilistic CEGAR

- enumerate paths of CE Markov chain
- visit paths with highest probability first [Han&Katoen 2007]
  - path $\sigma_1^\sharp$
    - if spurious generate predicate (interpolation)
  - path $\sigma_2^\sharp$
  - ...

$\longleftarrow$ abstract probability mass $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e})$ $\longrightarrow$

$\sigma_1^\sharp$	$\sigma_2^\sharp$	$\sigma_3^\sharp$	$\sigma_4^\sharp$	$\sigma_5^\sharp$

Probabilistic CEGAR

- enumerate paths of CE Markov chain
- visit paths with highest probability first [Han&Katoen 2007]
  - path $\sigma_1^\sharp$
    - if spurious generate predicate (interpolation)
  - path $\sigma_2^\sharp$
  - ...
- realisable probability mass $> p$?

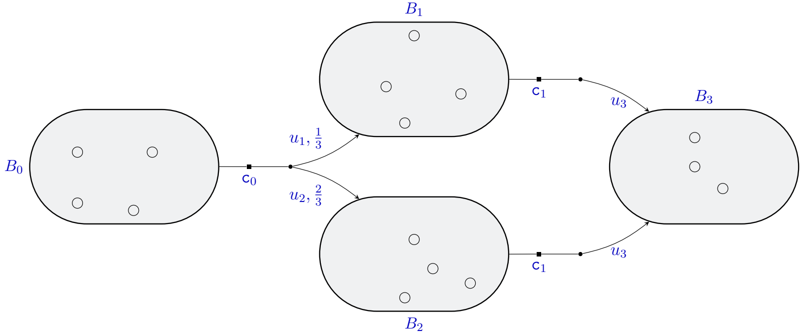
$\longleftarrow$ abstract probability mass $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e})$ $\longrightarrow$

$\sigma_1^\sharp$	$\sigma_2^\sharp$	$\sigma_3^\sharp$	$\sigma_4^\sharp$	$\sigma_5^\sharp$

$\leftarrow$ $\mathcal{P}_B^{\eta^\sharp}$(realisable paths) $> p$? $\rightarrow$

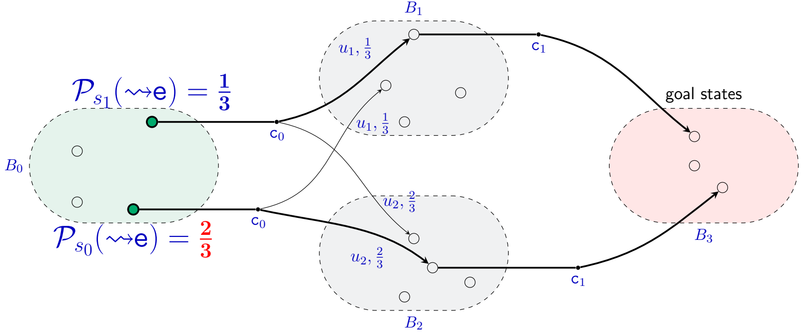
Computing realisable probability is harder than it seems

realisable probability $\neq \sum\{\mathcal{P}(\sigma^\sharp) \mid \text{realizable path}\} = \mathbf{1}$



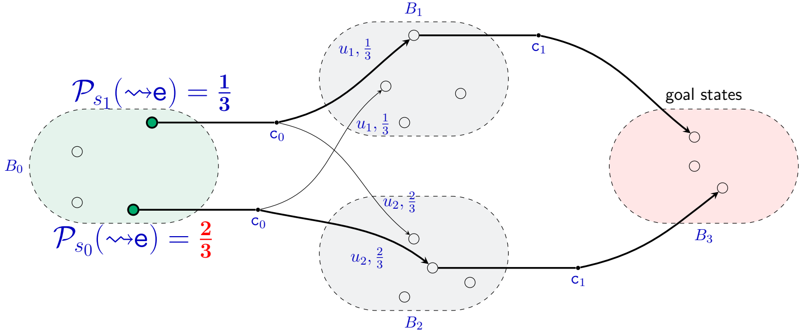
Computing realisable probability is harder than it seems

$\frac{2}{3}$ = **realisable probability** $\neq \sum\{\mathcal{P}(\sigma^\sharp) \mid \text{realizable path}\} = \mathbf{1}$



Computing realisable probability is harder than it seems

$\frac{2}{3}$ = **realisable probability** $\neq \sum\{\mathcal{P}(\sigma^\sharp) \mid \text{realizable path}\} = \mathbf{1}$



Theorem (Realisable probability)

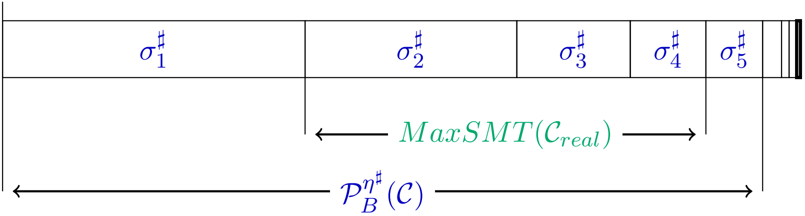
Realisable probability as optimisation problem:

$MaxSmt(exp_1, \ldots, exp_n) = \max\{\sum_{i=1}^{n} [\![exp_i]\!]_s \cdot p_i \mid s \in [\![I \wedge F(B)]\!]\}$

where exp_i *characterizes path* σ_i

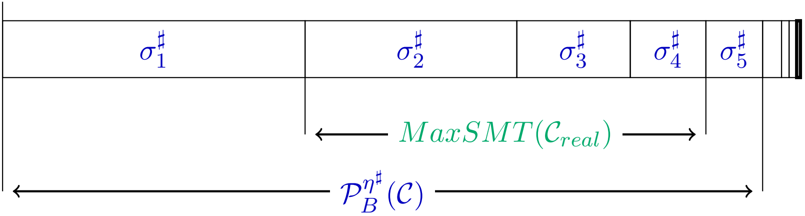
Probabilistic CEGAR

- analyse paths in Markov chain with decreasing probability
  - spurious paths give predicates
  - realizable paths can improve lower bound
    - MaxSMT



Probabilistic CEGAR

- analyse paths in Markov chain with decreasing probability
  - spurious paths give predicates
  - realizable paths can improve lower bound
    - MaxSMT



- Semi-decision procedure for probabilistic CE analysis
  - always terminates returning either
    - ✗ CE realizable
    - ✔ CE spurious and predicate
    - ? don't know and predicate
  - incomplete to ensure termination
    - limit on number of spurious paths

Justification for Incompleteness: Undecidability

- A CE analysis problem consists of
  - probabilistic program M
  - threshold p
  - abstraction G of M
  - CE $(B, \eta^\sharp)$ in G, i.e., $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e}) > p$
- **decide** if the CE is real
  - there is a corresponding concrete CE (s, η) with $\mathcal{P}_s^{\eta}(\rightsquigarrow \mathsf{e}) > p$

Justification for Incompleteness: Undecidability

- A CE analysis problem consists of
  - probabilistic program M
  - threshold p
  - abstraction G of M
  - CE $(B, \eta^\sharp)$ in G, i.e., $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathtt{e}) > p$
- **decide** if the CE is real
  - there is a corresponding concrete CE (s, η) with $\mathcal{P}_s^{\eta}(\rightsquigarrow \mathtt{e}) > p$
- **assume**: expressions language = linear arithmetic over the integers

  $\Rightarrow$ conventional counterexample analysis decidable

Justification for Incompleteness: Undecidability

- A CE analysis problem consists of
  - probabilistic program M
  - threshold p
  - abstraction G of M
  - CE $(B, \eta^\sharp)$ in G, i.e., $\mathcal{P}_B^{\eta^\sharp}(\rightsquigarrow \mathsf{e}) > p$
- **decide** if the CE is real
  - there is a corresponding concrete CE (s, η) with $\mathcal{P}_s^{\eta}(\rightsquigarrow \mathsf{e}) > p$
- **assume**: expressions language = linear arithmetic over the integers

  $\Rightarrow$ conventional counterexample analysis decidable

Theorem (Undecidability of Counterexample Analysis)

Counterexample analysis for probabilistic programs is undecidable

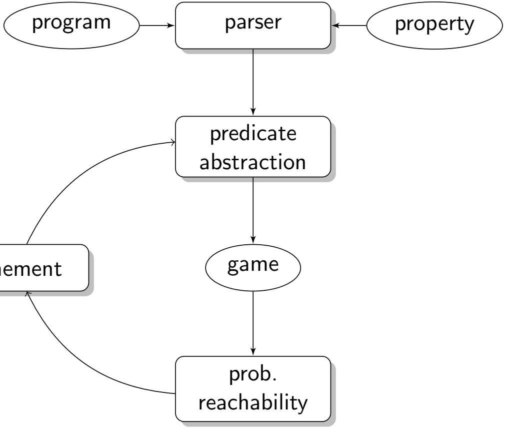
Proof.

Halting problem for counter machines can be reduced to CE analysis □

Backward Refinement (VMCAI 2010)

- Refinement if no threshold is available
- Target uncertainty from abstraction (leverage game strategies)

The PASS tool



Case Study: BRP

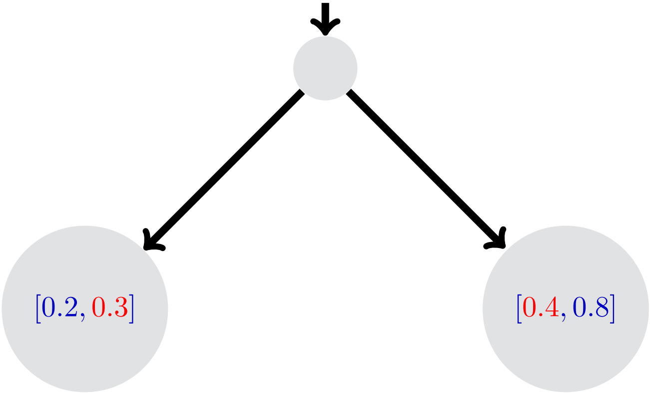
- Probability to reach:

  “receiver does not receive any chunk and sender tried to send a chunk”
- Can be analyzed for an infinite parameter range with PASS
  - for any file size $\geq$ 16, probability is $1.6E-7$
- PASS provides proof for arbitrary file size
- BRP is just one case study:

Case study (parameters)			Property	Conventional states	Conventional trans	Conventional time	Abstraction states	Abstraction trans	Abstraction refs	Abstraction preds	Abstraction paths	Abstraction time
WLAN (BOFF T)	5	315	k=3	5,195K	11,377K	93	34K	36K	9	120	604	72
	6	315	k=3	12,616K	28,137K	302	34K	42K	9	116	604	88
	6	315	k=6	12,616K	28,137K	2024	771K	113K	9	182	582	306
	6	9500	k=6	–	–	**TO**	771K	113K	9	182	582	311
CSMA/CD (BOFF)	3		p1	41K	52K	10	1K	2K	8	58	28	9
	4		p1	124K	161K	56	6K	9K	14	100	56	38
	3		p2	41K	52K	10	0.5K	0.9K	12	41	28	10
	4		p2	124K	161K	21	0.5K	1.5K	12	41	44	11
BRP (N MAX)	16	3	p1	2K	3K	5.4	2K	3K	9	46	41	9
	32	5	p1	5K	7K	12	5K	7K	9	64	111	21
	64	5	p1	10K	14K	26	10K	14K	8	95	585	91
	$>$16	3	p4	∞	–	–	0.5K	0.9K	7	26	17	3
	$>$16	4	p4	∞	–	–	0.6K	1K	7	27	17	4
	$>$16	5	p4	∞	–	–	0.7K	1K	8	28	18	5
SW			goodput	∞	–	–	5K	11K	3	40	7	87
			timeout	∞	–	–	27K	44K	3	49	6	89

Why abstraction works.

- absolute values do matter but only, e.g., differences between variables
- cannot contribute to optimum



Contributions

a novel analysis method for probabilistic programs:

- symbolic abstraction to tackle large state spaces
  - **Première**: predicate abstraction in probabilistic verification
    - prior work in qualitative software verification
  - **Challenge**

    complex interplay

    uncertainty from abstraction

    probabilism — concurrency

- refinement to achieve full automation
  - **Première**: Probabilistic CEGAR
    - prior work: CEGAR in qualitative software verification
  - **Challenge**
    - counterexamples are Markov chains
- implemented in PASS tool

Limitations

- Abstraction is not a panacea / silver bullet
  - can be less efficient for certain finite-state models
- Probabilistic CEGAR:
  - lower thresholds for minimal reachability?
- No support for state-dependent probabilities:

[] m=1 & x>0 -> $\frac{1}{x}$: (x'=x-1) + $\frac{x-1}{x}$: (m'=3);

Avenues for Future Work

- Beyond probabilistic reachability for MDPs
  - rewards and expectations
  - Exponential distributions
  - Support for full PCTL
- Richer input language
  - Modest

Thesis Statement

Abstraction enables automatic verification of probabilistic programs with large and, for the first time, infinite state spaces.