Tutorial:
Probabilistic Model Checking

Christel Baier
Technische Universität Dresden

Tutorial: Probabilistic Model Checking

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles
* mean-payoff
* expected accumulated weights

Markov decision processes (MDP)

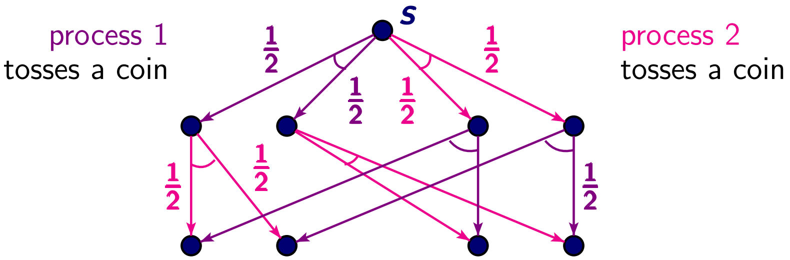
Markov decision processes (MDP)

extend Markov chains by nondeterminism

Markov decision processes (MDP)

extend Markov chains by nondeterminism

- modeling asynchronous distributed systems by interleaving

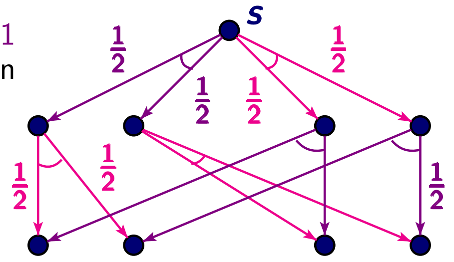


Markov decision processes (MDP)

extend Markov chains by nondeterminism

- modeling asynchronous distributed systems by interleaving
- useful for abstraction purposes
- representation of the interface with an unpredictable environment, e.g., human user

process 1
tosses a coin



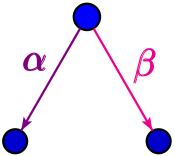
process 2
tosses a coin

From TS and MC to MDP

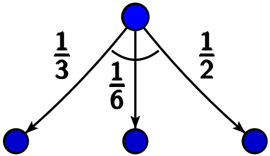
TS:	transition system
MC:	Markov chain
MDP:	Markov decision process

From TS and MC to MDP

transition system
purely nondeterministic



Markov chain
purely probabilistic



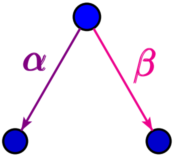
α, β are action names

TS: transition system
MC: Markov chain
MDP: Markov decision process

From TS and MC to MDP

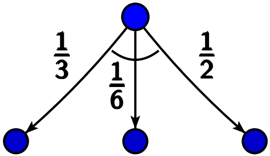
transition system

purely nondeterministic

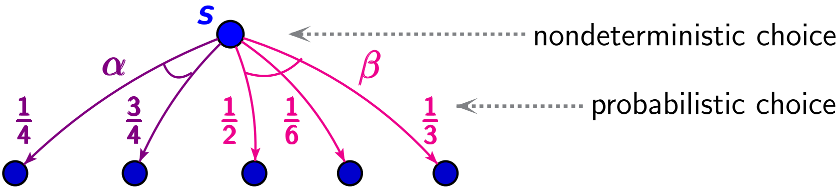


Markov chain

purely probabilistic

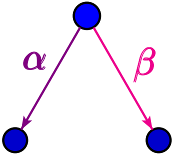


Markov decision process (MDP)

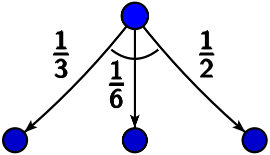


From TS and MC to MDP

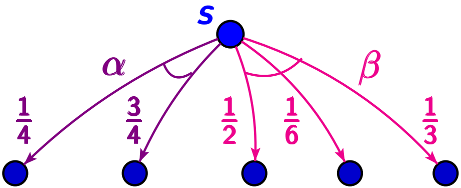
transition system
purely nondeterministic



Markov chain
purely probabilistic



Markov decision process (MDP)

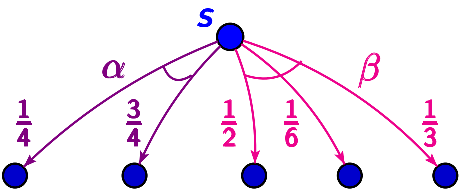


integer weights
$wgt(s, \alpha) \in \mathbb{Z}$

Markov decision process (MDP)

$\mathcal{M} = (S, Act, P, \ldots)$

- finite state space S
- Act finite set of actions



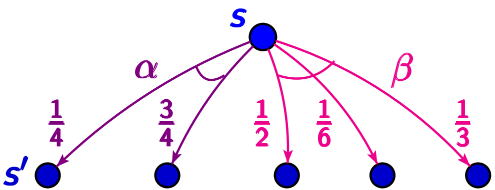
Markov decision process (MDP)

$\mathcal{M} = (S, Act, P, \ldots)$

- finite state space S
- Act finite set of actions
- transition probability fct. $P : S \times Act \times S \to [0,1]$

$$\forall s \in S \;\; \forall \alpha \in Act. \;\; \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$$

$\alpha \notin Act(s)$ ↗ $\quad$ ↖ $\alpha \in Act(s)$



nondeterministic choice between enabled actions

$Act(s) = \{\alpha, \beta\}$

Markov decision process (MDP)

$$\mathcal{M} = (S, Act, P, rew_1, rew_2, \ldots)$$

- finite state space S
- Act finite set of actions
- transition probability fct. $P : S \times Act \times S \to [0,1]$

$$\forall s \in S \ \ \forall \alpha \in Act. \ \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$$

- reward functions rew_1, rew_2, $\ldots : S \times Act \to \mathbb{N}$

energy (rew_1) utility (rew_2)

Weighted MDP

$$\mathcal{M} = (S, Act, P, wgt_1, wgt_2, \ldots)$$

- finite state space S
- Act finite set of actions
- transition probability fct. $P : S \times Act \times S \to [0, 1]$

$$\forall s \in S \;\; \forall \alpha \in Act. \;\; \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$$

- weight functions wgt_1, wgt_2, $\ldots : S \times Act \to \mathbb{Z}$

energy level of a battery (wgt_1)

win and loss of a share at the stock market (wgt_2)

Weighted MDP

$\mathcal{M} = (S, Act, P, wgt_1, wgt_2, \ldots)$

- finite state space S
- Act finite set of actions
- transition probability fct. $P : S \times Act \times S \to [0,1]$

  $\forall s \in S \ \ \forall \alpha \in Act. \ \sum_{s' \in S} P(s, \alpha, s') \in \{0,1\}$
- weight functions wgt_1, wgt_2, $\ldots : S \times Act \to \mathbb{Z}$

accumulated weight of finite paths:

$$wgt_1(s_0 \xrightarrow{\alpha_1} \ldots \xrightarrow{\alpha_n} s_n) \ = \ \sum_{i=0}^{n-1} wgt_1(s_i, \alpha_{i+1})$$

Weighted MDP

$$\mathcal{M} = (S, Act, P, wgt_1, wgt_2, \ldots)$$

- finite state space S
- Act finite set of actions
- transition probability fct. $P : S \times Act \times S \to [0,1]$

$$\forall s \in S \;\; \forall \alpha \in Act. \;\; \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$$

- weight functions wgt_1, wgt_2, $\ldots : S \times Act \to \mathbb{Z}$

ratios of accumulated weights:

$$ratio = \frac{cost}{util} : FinPaths \to \mathbb{Q} \qquad \begin{aligned} cost &= wgt_1 \\ util &= wgt_2 \end{aligned}$$

Probability measure

$$\mathcal{M} = (S, Act, P, wgt_1, wgt_2, \ldots)$$

- finite state space S
- Act finite set of actions
- transition probability fct. $P : S \times Act \times S \to [0,1]$

$$\forall s \in S \;\; \forall \alpha \in Act. \;\; \sum_{s' \in S} P(s, \alpha, s') \in \{0,1\}$$

- weight functions $wgt_1, \; wgt_2, \; \ldots : S \times Act \to \mathbb{Z}$

---

probabilities measure $\Pr_s^{\sigma}$ for given state $s \in S$ and scheduler $\sigma : FinPaths \to Distr(Act)$

$$\underbrace{FinPaths}_{\text{history}} \;\; \underbrace{Distr(Act)}_{\text{probabilities for next actions}}$$

Classification of schedulers

randomized vs deterministic schedulers:

randomized (R):	select a distribution of actions
deterministic (D):	select a unique action

Classification of schedulers

randomized vs deterministic schedulers:

randomized (R):	select a distribution of actions
deterministic (D):	select a unique action

memory requirements:

consider schedulers as triples (Mem, μ, ν)

- Mem is a set of memory cells
- $\mu : \mathit{Mem} \times S \to \mathit{Distr}(\mathit{Act})$ decision function
- $\nu : \mathit{Mem} \times S \to \mathit{Mem}$ memory-update function

no restriction (H):	possibly infinitely many memory cells
finite-memory (FM):	finitely many memory cells
memoryless (M):	decisions only depend on the current state

Randomized mutual exclusion protocol

Randomized mutual exclusion protocol

- 2 concurrent processes P_1, P_2 with 3 phases:

n_i	noncritical actions of process P_i
w_i	waiting phase of process P_i
c_i	critical section of process P_i

Randomized mutual exclusion protocol

- 2 concurrent processes P_1, P_2 with 3 phases:

n_i	noncritical actions of process P_i
w_i	waiting phase of process P_i
c_i	critical section of process P_i

- competition if both processes are waiting

Randomized mutual exclusion protocol

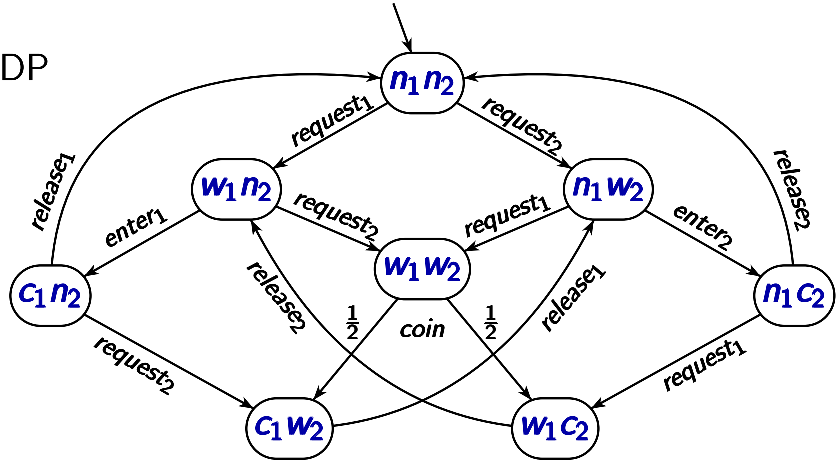
- 2 concurrent processes P_1, P_2 with 3 phases:

n_i	noncritical actions of process P_i
w_i	waiting phase of process P_i
c_i	critical section of process P_i

- competition if both processes are waiting
- resolved by a randomized arbiter who tosses a coin

Randomized mutual exclusion protocol

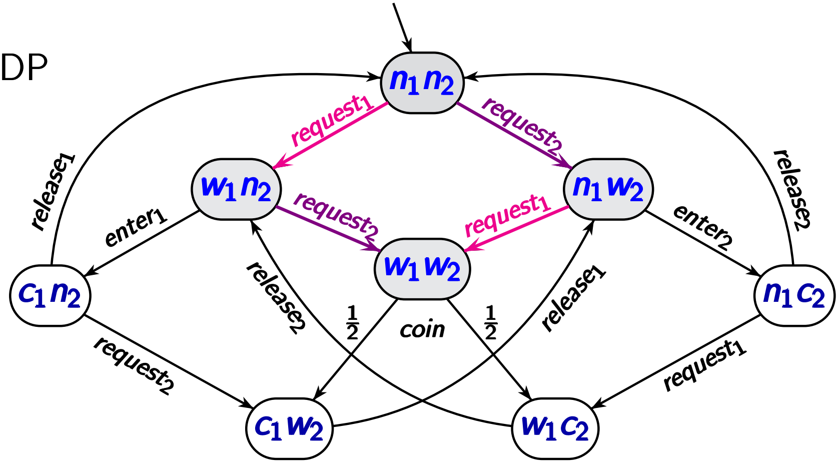
MDP



- interleaving of the request operations
- competition if both processes are waiting
- randomized arbiter tosses a coin if both are waiting

Randomized mutual exclusion protocol

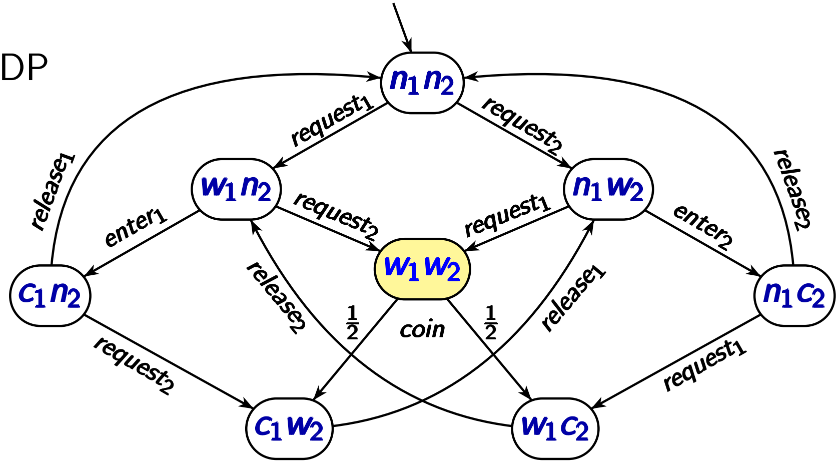
MDP



- interleaving of the request operations
- competition if both processes are waiting
- randomized arbiter tosses a coin if both are waiting

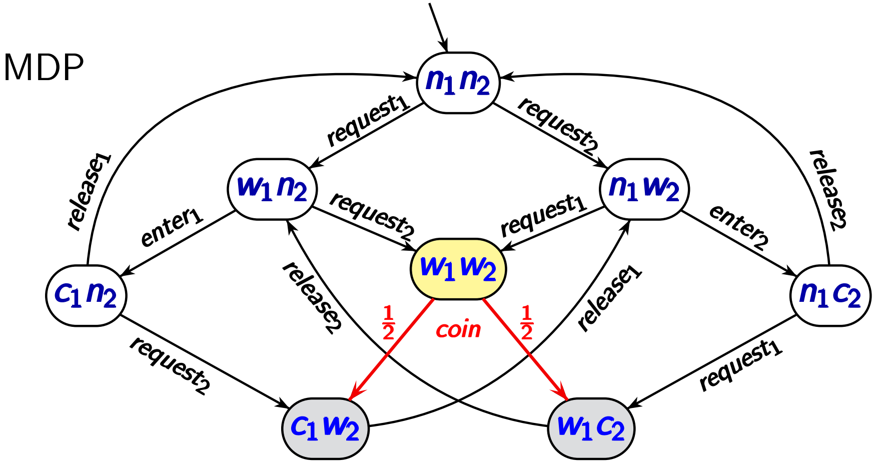
Randomized mutual exclusion protocol

MDP



- interleaving of the request operations
- competition if both processes are waiting
- randomized arbiter tosses a coin if both are waiting

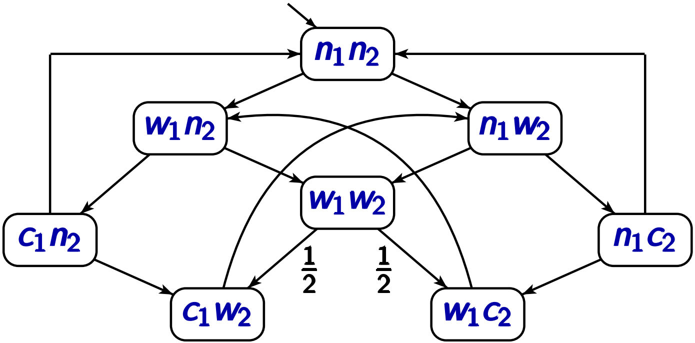
Randomized mutual exclusion protocol



- interleaving of the request operations
- competition if both processes are waiting
- randomized arbiter tosses a coin if both are waiting

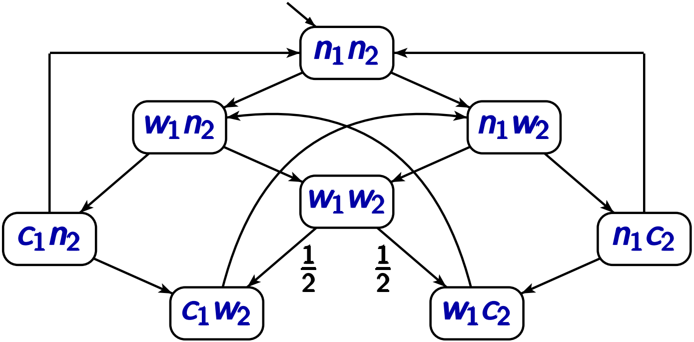
Properties of the randomized MUTEX

Properties of the randomized MUTEX



safety: the processes are never simultaneously in their critical section

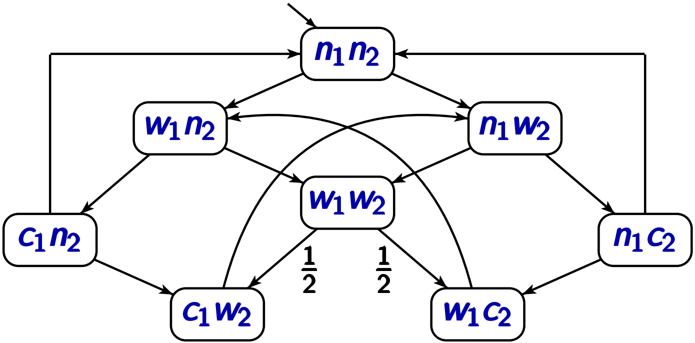
Properties of the randomized MUTEX



safety: the processes are never simultaneously in their critical section

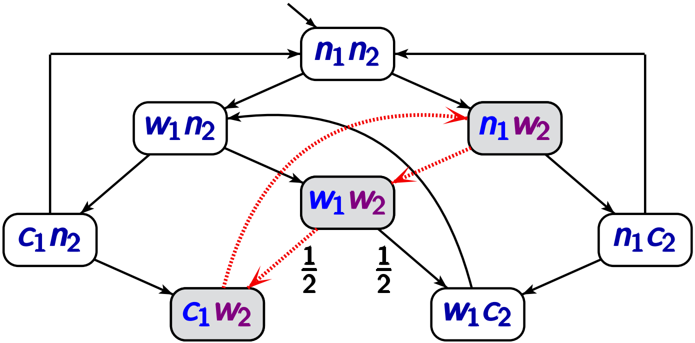
holds on all paths as state $\langle c_1, c_2 \rangle$ is unreachable

Properties of the randomized MUTEX



liveness: each waiting process will eventually enter its critical section

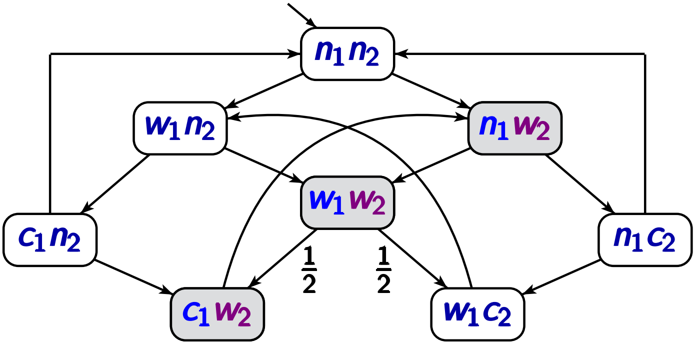
Properties of the randomized MUTEX



liveness: each waiting process will eventually enter its critical section

does not hold on all paths, but almost surely

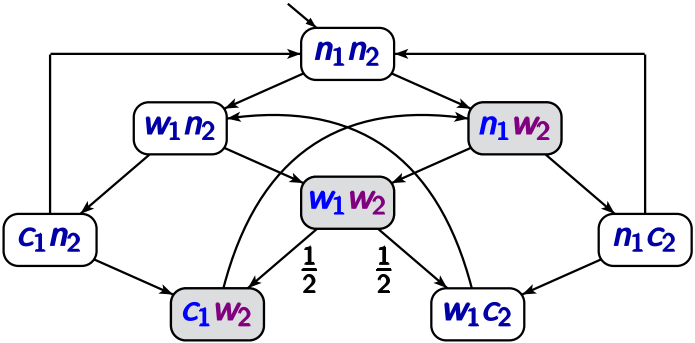
Properties of the randomized MUTEX



Suppose process **2** is waiting.

What is the probability that process **2** enters its critical section within the next **3** steps **?**

Properties of the randomized MUTEX

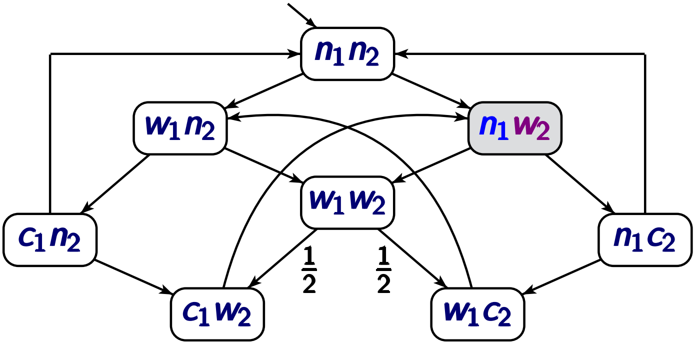


Suppose process **2** is waiting.

What is the probability that process **2** enters its critical section within the next **3** steps **?**

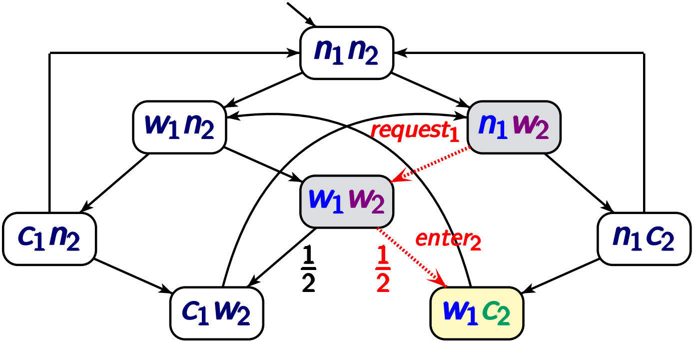
... depends ...

Randomized mutual exclusion protocol



Suppose the current state is $\langle n_1, w_2 \rangle$.

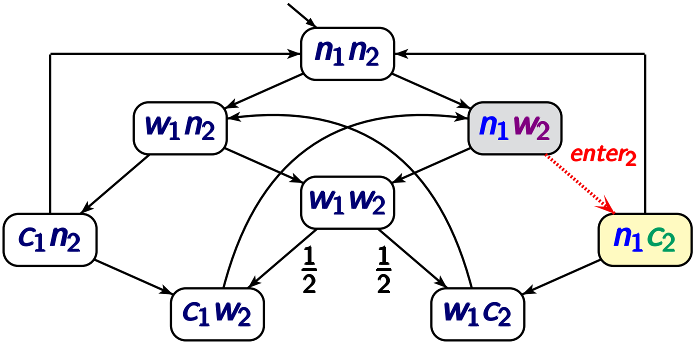
Randomized mutual exclusion protocol



The probability that process **2** enters its critical section within the next **3** steps is:

$\frac{1}{2}$ if process **1** is scheduled in state $\langle n_1, w_2 \rangle$

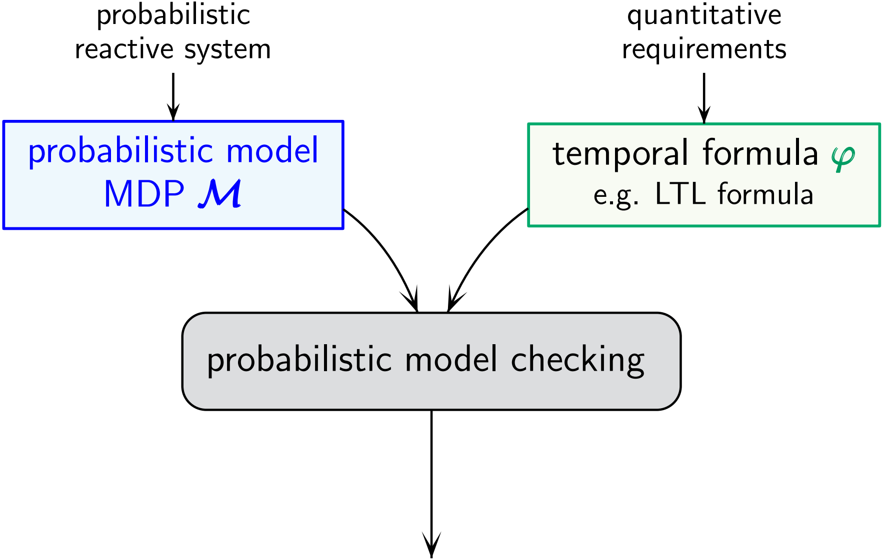
Randomized mutual exclusion protocol



The probability that process 2 enters its critical section within the next 3 steps is:

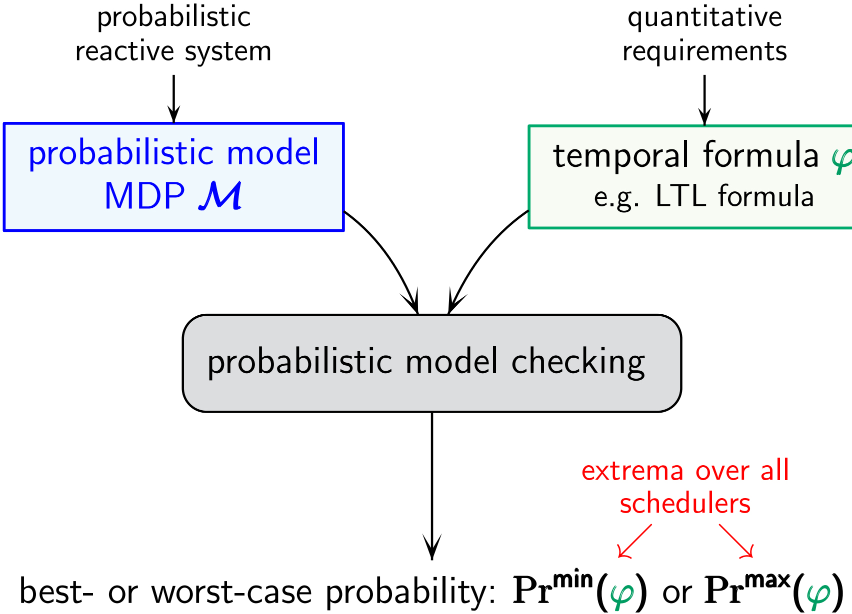
- $\frac{1}{2}$ if process 1 is scheduled in state $\langle n_1, w_2 \rangle$
- 1 if process 2 is scheduled in state $\langle n_1, w_2 \rangle$

Probabilistic model checking

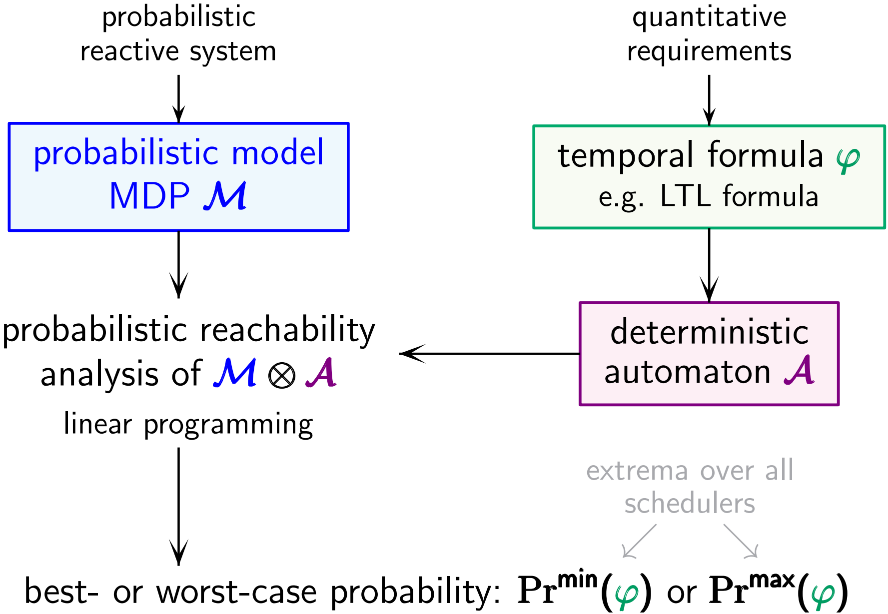


best- or worst-case probability: $\Pr^{\min}(\varphi)$ or $\Pr^{\max}(\varphi)$

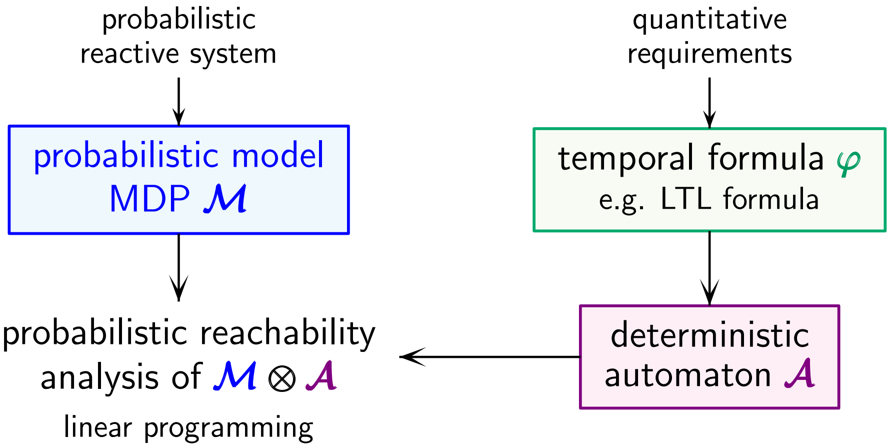
Probabilistic model checking



Probabilistic model checking



Probabilistic model checking



$$\Pr^{\max}_{\mathcal{M},s}(\varphi) \;=\; \Pr^{\max}_{\mathcal{M}\otimes\mathcal{A},s'}\big(\Diamond accEC\big)$$

maximal probability
to reach an accepting
end component

End components (EC) [de Alfaro'96]

End components (EC) [de Alfaro'96]

Let $\mathcal{M} = (S, Act, P, \ldots)$ be an MDP.

An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP

End components (EC) [de Alfaro'96]

Let $\mathcal{M} = (S, Act, P, \ldots)$ be an MDP.

An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP, i.e., a pair $\mathcal{E} = (T, A)$ where $\varnothing \neq T \subseteq S$ and $A : T \to 2^{Act}$ s.t.

(1) $\ldots$

(2) $\ldots$

(3) $\ldots$

End components (EC) [DE ALFARO'96]

Let $\mathcal{M} = (S, Act, P, \ldots)$ be an MDP.

An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP, i.e., a pair $\mathcal{E} = (T, A)$ where $\varnothing \neq T \subseteq S$ and $A : T \to 2^{Act}$ s.t.

(1) enabledness of selected actions:

$$\varnothing \neq A(t) \subseteq Act(t) \quad \text{for all } t \in T$$

(2) ...

(3) ...

End components (EC) [de Alfaro'96]

Let $\mathcal{M} = (S, Act, P, \ldots)$ be an MDP.

An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP, i.e., a pair $\mathcal{E} = (T, A)$ where $\varnothing \neq T \subseteq S$ and $A : T \to 2^{Act}$ s.t.

(1) enabledness of selected actions:

$$\varnothing \neq A(t) \subseteq Act(t) \quad \text{for all } t \in T$$

(2) closed under probabilistic branching:

$$\forall t \in T \, \forall \alpha \in A(t). \ \big(P(t, \alpha, u) > 0 \Longrightarrow u \in T\big)$$

(3) ...

End components (EC) [de Alfaro'96]

Let $\mathcal{M} = (S, Act, P, \ldots)$ be an MDP.

An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP, i.e., a pair $\mathcal{E} = (T, A)$ where $\varnothing \neq T \subseteq S$ and $A : T \to 2^{Act}$ s.t.

(1) enabledness of selected actions:

$$\varnothing \neq A(t) \subseteq Act(t) \quad \text{for all } t \in T$$

(2) closed under probabilistic branching:

$$\forall t \in T\, \forall \alpha \in A(t).\ \big(P(t, \alpha, u) > 0 \Longrightarrow u \in T\big)$$

(3) the underlying graph is strongly connected

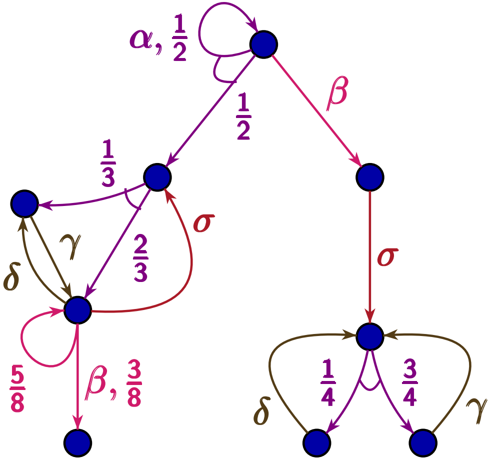
Let $\mathcal{M} = (S, Act, P, \ldots)$ be an MDP.

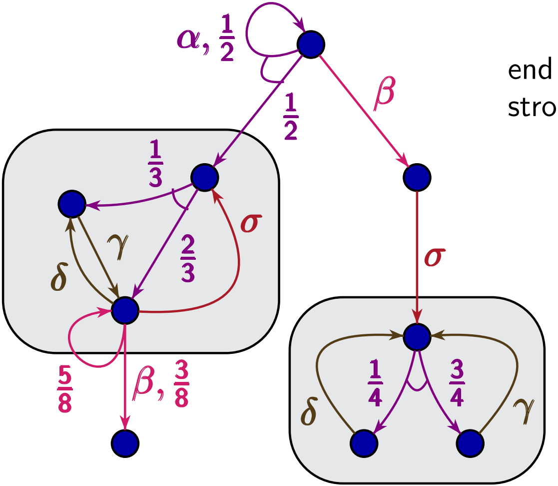
An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP, i.e., a pair $\mathcal{E} = (T, A)$ where $\varnothing \neq T \subseteq S$ and $A : T \to 2^{Act}$ s.t. ...

Often viewed as a set of state-action pairs:

$$\mathcal{E} = \{ (s, \alpha) : s \in T, \ \alpha \in A(s) \}$$

End components (EC) [de Alfaro'96]

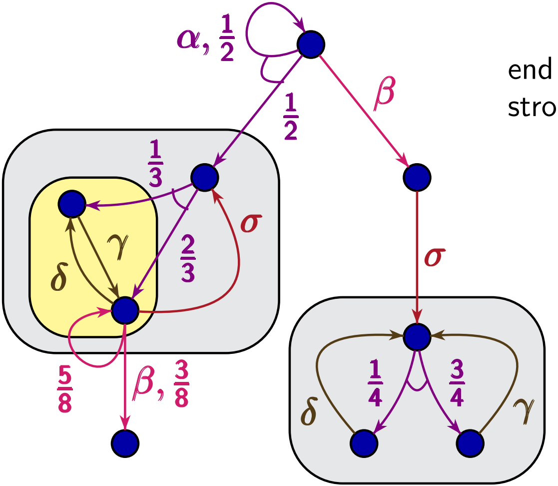




end component (EC):
strongly connected sub-MDP

End components (EC)

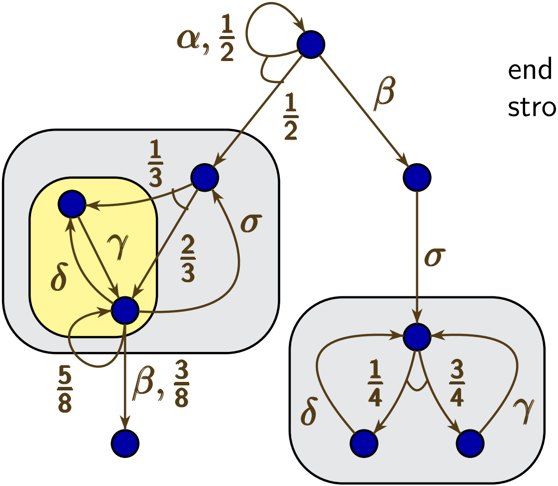
[DE ALFARO'96]



end component (EC):
strongly connected sub-MDP

End components (EC) [de Alfaro'96]

For all schedulers: almost all infinite paths eventually enter an EC and visit all its states infinitely often.



end component (EC):
strongly connected sub-MDP

End components (EC) ... for MDPs without traps

For all schedulers: almost all infinite paths eventually enter an EC and visit all its states infinitely often.

More precisely, for all schedulers σ and states s:

$$\Pr^{\sigma}\left\{ \pi \in \mathit{Paths}(s) : \begin{array}{l} \mathit{limit}(\pi) \text{ is an} \\ \text{end component} \end{array} \right\} = 1$$

limit of an infinite path π:

$$\mathit{limit}(\pi) = \left\{ \begin{array}{l} \text{set of state-action pairs that} \\ \text{appear infinitely often in } \pi \end{array} \right.$$

trap: state without actions

End components (EC) ... for MDPs without traps

For all schedulers: almost all infinite paths eventually enter an EC and visit all its states infinitely often.

More precisely, for all schedulers σ and states s:

$$\Pr^{\sigma}\left\{ \pi \in \mathit{Paths}(s) : \begin{array}{l} \mathit{limit}(\pi) \text{ is an} \\ \text{end component} \end{array} \right\} = 1$$

Let E be a limit property and $T_1, \ldots, T_k \subseteq S$ s.t.

$$\pi \models E \quad \text{iff} \quad \exists i \geqslant 0.\ \inf(\pi) = T_i$$

↑ set of states that appear infinitely often in π

End components (EC) ... for MDPs without traps

For all schedulers: almost all infinite paths eventually enter an EC and visit all its states infinitely often.

More precisely, for all schedulers σ and states s:

$$\Pr^{\sigma}\left\{ \pi \in Paths(s) : \begin{array}{l} limit(\pi) \text{ is an} \\ \text{end component} \end{array} \right\} = 1$$

Let E be a limit property and $T_1, \ldots, T_k \subseteq S$ s.t.

$$\pi \models E \quad \text{iff} \quad \exists i \geqslant 0. \ \inf(\pi) = T_i$$

Then: $\Pr_s^{\max}(E) = \Pr_s^{\max}(\Diamond T)$ where

$$T \ = \ \bigcup \left\{ T_i : T_i \text{ constitutes an end component} \right\}$$

Quantitative analysis of Rabin conditions

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$\Diamond$	eventually	$\Diamond\Box$	almost forever
$\Box$	always	$\Box\Diamond$	infinitely often

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$$\Pr_s^{\max}(E) \;=\; \Pr_s^{\max}(\,\Diamond\, accEC\,)$$

$\Diamond$ eventually $\quad$ $\Diamond\Box$ almost forever

$\Box$ always $\quad$ $\Box\Diamond$ infinitely often

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$$\Pr^{\max}_s(E) \;=\; \Pr^{\max}_s(\,\Diamond\, accEC\,)$$

↑

union of all end components T that "meet E", i.e.,

$\exists i \in \{1,\ldots,k\}.\;\; T \cap L_i = \varnothing$ and $T \cap U_i \neq \varnothing$

$\Diamond$	eventually	$\Diamond\Box$	almost forever
$\Box$	always	$\Box\Diamond$	infinitely often

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$$\Pr_s^{\max}(E) = \Pr_s^{\max}(\Diamond\, accEC)$$

$$= \Pr_s^{\max}(\Diamond\, accMEC)$$

$\bigcup_{1 \leqslant i \leqslant k}$ union of all maximal end components T in $\mathcal{M} \setminus L_i$ s.t. $T \cap U_i \neq \varnothing$

$\Diamond$	eventually	$\Diamond\Box$	almost forever
$\Box$	always	$\Box\Diamond$	infinitely often

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$$\begin{aligned} \Pr_s^{\max}(E) &= \Pr_s^{\max}(\Diamond\, accEC\,) \\ &= \Pr_s^{\max}(\Diamond\, accMEC\,) \end{aligned}$$

$\bigcup_{1 \leqslant i \leqslant k}$ union of all maximal end components T in $\mathcal{M} \setminus L_i$ s.t. $T \cap U_i \neq \varnothing$

analogous approach for generalized Rabin conditions:

$$\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_{i,1} \wedge \ldots \wedge \Box\Diamond U_{i,k_i})$$

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$$\begin{aligned} \Pr_s^{\max}(E) &= \Pr_s^{\max}(\,\Diamond\, accEC\,) \\ &= \Pr_s^{\max}(\,\Diamond\, accMEC\,) \end{aligned}$$

---

model checking algorithm for Rabin condition E:

1. compute the maximal end components
2. check which of them fulfills E
3. compute maximal reachability probabilities

   by linear-programming techniques

Quantitative analysis of Rabin conditions

Let E be a Rabin condition $\bigvee_{1 \leqslant i \leqslant k} (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)$.

$$\begin{aligned} \Pr_s^{\max}(E) &= \Pr_s^{\max}(\Diamond\, accEC) \\ &= \Pr_s^{\max}(\Diamond\, accMEC) \end{aligned}$$

---

model checking algorithm for Rabin condition E:

1. compute the maximal end components
2. check which of them fulfills E
3. compute maximal reachability probabilities
   by linear-programming techniques

Computation of maximal end components

maximal end component (MEC):
end component that is not contained in any other end component

Computation of maximal end components

```
REPEAT
```

compute the SCCs of $\mathcal{M}$;

maximal end component (MEC):

end component that is not contained in any other end component

Computation of maximal end components

REPEAT

compute the SCCs of $\mathcal{M}$;

IF there exist states s, t and an action α such that $P(s, \alpha, t) > 0$ and s, t belong to different SCCs

maximal end component (MEC):

end component that is not contained in any other end component

Computation of maximal end components

```
REPEAT
```

compute the SCCs of $\mathcal{M}$;

IF there exist states s, t and an action α such that $P(s, \alpha, t) > 0$ and s, t belong to different SCCs

THEN choose such a pair $\langle s, \alpha \rangle$;
remove α from $Act(s)$;

maximal end component (MEC):
end component that is not contained in any other end component

Computation of maximal end components

REPEAT

compute the SCCs of $\mathcal{M}$;

IF there exist states s, t and an action α such that $P(s, \alpha, t) > 0$ and s, t belong to different SCCs

THEN choose such a pair $\langle s, \alpha \rangle$;

remove α from $Act(s)$;

IF s is a trap THEN remove s FI

maximal end component (MEC):

end component that is not contained in any other end component

Computation of maximal end components

```
REPEAT
    compute the SCCs of M;
    IF  there exist states s, t and an action α such that
        P(s, α, t) > 0 and s, t belong to different SCCs
    THEN  choose such a pair ⟨s, α⟩;
          remove α from Act(s);
          IF s is a trap  THEN  remove s  FI
    FI
UNTIL no further changes
```

Computation of maximal end components

```
REPEAT
    compute the SCCs of M;
    IF   there exist states s, t and an action α such that
         P(s, α, t) > 0 and s, t belong to different SCCs
    THEN  choose such a pair ⟨s, α⟩;
          remove α from Act(s);
          IF s is a trap  THEN  remove s  FI
    FI
UNTIL no further changes
```

return the non-trivial SCCs as maximal end components

Computation of maximal end components

```
REPEAT
    compute the SCCs of M;
    IF   there exist states s, t and an action α such that
         P(s, α, t) > 0 and s, t belong to different SCCs
    THEN  choose such a pair ⟨s, α⟩;
          remove α from Act(s);
          IF s is a trap  THEN  remove s  FI
    FI
UNTIL no further changes
```

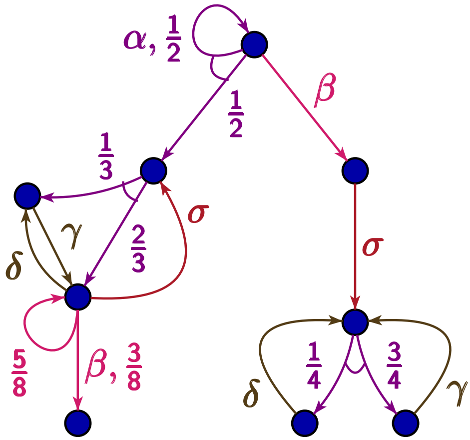
time complexity: $\mathcal{O}(\, size(\mathcal{M})^2 \,)$

return the non-trivial SCCs as maximal end components

MEC-quotient

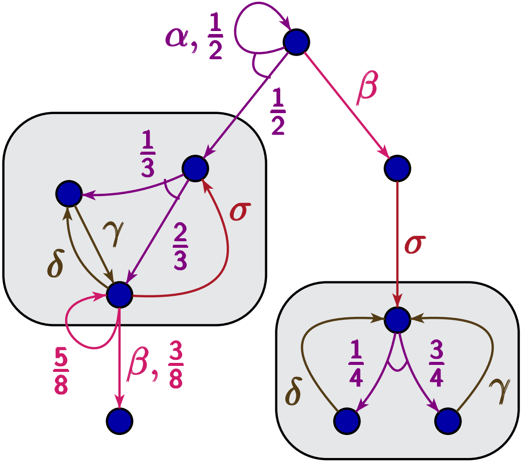
Idea: The MEC-quotient is the MDP $\mathbf{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient



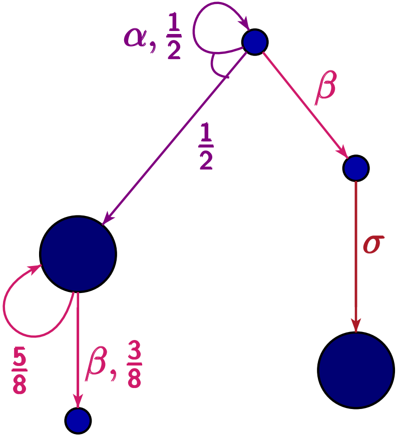
Idea: The MEC-quotient is the MDP $\mathrm{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient



Idea: The MEC-quotient is the MDP $\mathrm{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient



Idea: The MEC-quotient is the MDP $\mathrm{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient

Given MDP $\mathcal{M} = (S, Act, P, \ldots)$ with MECs $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i = (T_i, A_i)$.

Idea: The MEC-quotient is the MDP $\mathrm{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient

Given MDP $\mathcal{M} = (S, Act, P, \ldots)$ with MECs $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i = (T_i, A_i)$. W.l.o.g., if $s, t \in T_i$ then:

$$Act(s) \cap Act(t) \;=\; A_i(s) \cap A_i(t)$$

Idea: The MEC-quotient is the MDP $\mathrm{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient

Given MDP $\mathcal{M} = (S, Act, P, \ldots)$ with MECs $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i = (T_i, A_i)$. W.l.o.g., if $s, t \in T_i$ then:

$$Act(s) \cap Act(t) \;=\; A_i(s) \cap A_i(t)$$

MEC-quotient $\mathrm{MEC}(\mathcal{M}) = (S', Act, P', \ldots)$ where

$$S' = (S \setminus T) \cup \{\mathcal{E}_1, \ldots, \mathcal{E}_k\} \;\text{ where } T = \bigcup_{1 \leqslant i \leqslant k} T_i$$

Idea: The MEC-quotient is the MDP $\mathrm{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.

MEC-quotient

Given MDP $\mathcal{M} = (S, Act, P, \ldots)$ with MECs $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i = (T_i, A_i)$. W.l.o.g., if $s, t \in T_i$ then:

$$Act(s) \cap Act(t) \ = \ A_i(s) \cap A_i(t)$$

MEC-quotient $\mathrm{MEC}(\mathcal{M}) = (S', Act, P', \ldots)$ where

$$S' = (S \setminus T) \cup \{\mathcal{E}_1, \ldots, \mathcal{E}_k\} \ \text{ where } T = \bigcup_{1 \leqslant i \leqslant k} T_i$$

enabled actions:

for $s \in S \setminus T$: as in $\mathcal{M}$

for state $\mathcal{E}_i$: all actions in $\bigcup_{s \in T_i} Act(s) \setminus A_i(s)$

MEC-quotient

Given MDP $\mathcal{M} = (S, Act, P, \ldots)$ with MECs $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i = (T_i, A_i)$. W.l.o.g., if $s, t \in T_i$ then:

$$Act(s) \cap Act(t) \; = \; A_i(s) \cap A_i(t)$$

MEC-quotient $\mathrm{MEC}(\mathcal{M}) = (S', Act, P', \ldots)$ where

$$S' = (S \setminus T) \cup \{\mathcal{E}_1, \ldots, \mathcal{E}_k\} \;\; \text{where } T = \bigcup_{1 \leqslant i \leqslant k} T_i$$

transition probabilities, e.g., if $s \in S \setminus T$, $\alpha \in Act(s)$:

$$P'(s, \alpha, s') \; = \; P(s, \alpha, s') \;\; \text{if } s' \in S \setminus T$$

$$P'(s, \alpha, \mathcal{E}_i) \; = \; \sum_{t \in T_i} P(s, \alpha, t)$$

MEC-quotient

Given MDP $\mathcal{M} = (S, Act, P, \ldots)$ with MECs $\mathcal{E}_1, \ldots, \mathcal{E}_k$ where $\mathcal{E}_i = (T_i, A_i)$. W.l.o.g., if $s, t \in T_i$ then:

$$Act(s) \cap Act(t) \;=\; A_i(s) \cap A_i(t)$$

MEC-quotient $\mathrm{MEC}(\mathcal{M}) = (S', Act, P', \ldots)$ where

$$S' = (S \setminus T) \cup \{\mathcal{E}_1, \ldots, \mathcal{E}_k\} \quad \text{where } T = \bigcup_{1 \leqslant i \leqslant k} T_i$$

if $s \in T_i$ and $\alpha \in Act(s) \setminus A_i(s)$:

$$P'(\mathcal{E}_i, \alpha, s') \;=\; P(s, \alpha, s') \quad \text{if } s' \in S \setminus T$$

$$P'(\mathcal{E}_i, \alpha, \mathcal{E}_j) \;=\; \sum_{t \in T_j} P(s, \alpha, t)$$

Properties of the MECs and the MEC-quotient

Properties of the MECs and the MEC-quotient

For all states s, t that belong to the same MEC:

$$\Pr_s^{\max}(\varphi) = \Pr_t^{\max}(\varphi)$$

for each prefix-independent path property φ.

Examples: $\varphi = \Diamond G$ or $\varphi = \Diamond\Box G$ or ...

The same holds for mininimal probabilities for prefix-independent properties and min/max expectations of long-run objectives.

Properties of the MECs and the MEC-quotient

For all states s, t that belong to the same MEC:

$$\Pr_s^{\max}(\varphi) = \Pr_t^{\max}(\varphi)$$

for each prefix-independent path property φ.

Hence: $\mathcal{M}$ and $\mathrm{MEC}(\mathcal{M})$ have the same maximal probabilities for prefix-independent properties.

Examples: $\varphi = \Diamond G$ or $\varphi = \Diamond\Box G$ or ...

The same holds for mininimal probabilities for prefix-independent properties and min/max expectations of long-run objectives.

Properties of the MECs and the MEC-quotient

For all states s, t that belong to the same MEC:

$$\Pr_s^{\max}(\varphi) = \Pr_t^{\max}(\varphi)$$

for each prefix-independent path property φ.

Hence: $\mathcal{M}$ and $\mathrm{MEC}(\mathcal{M})$ have the same maximal probabilities for prefix-independent properties.

$\mathrm{MEC}(\mathcal{M})$ has no end components.

Properties of the MECs and the MEC-quotient

For all states s, t that belong to the same MEC:

$$\Pr^{\max}_{s}(\varphi) = \Pr^{\max}_{t}(\varphi)$$

for each prefix-independent path property φ.

Hence: $\mathcal{M}$ and $\mathrm{MEC}(\mathcal{M})$ have the same maximal probabilities for prefix-independent properties.

$\mathrm{MEC}(\mathcal{M})$ has no end components. Hence:

$$\Pr^{\min}_{\mathrm{MEC}(\mathcal{M}),s}(\Diamond\, \mathit{Trap}) = 1$$

↑

set of states t with $\mathit{Act}(t) = \varnothing$

Properties of the MECs and the MEC-quotient

For all states s, t that belong to the same MEC:

$$\Pr_s^{\max}(\varphi) = \Pr_t^{\max}(\varphi)$$

for each prefix-independent path property φ.

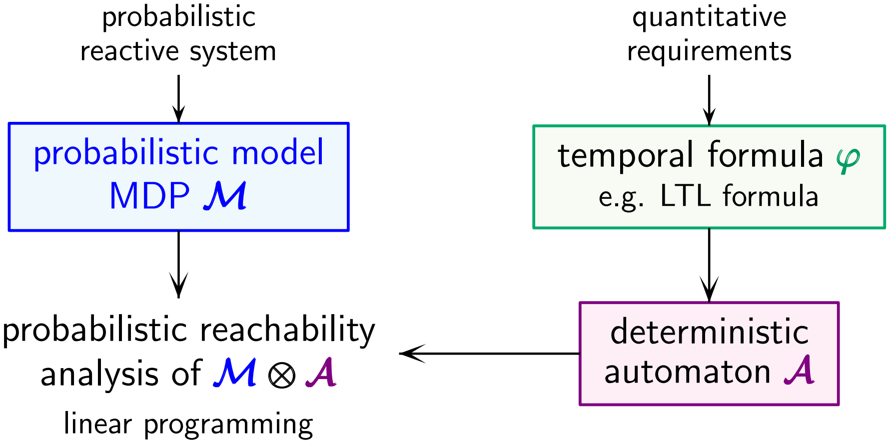
Hence: $\mathcal{M}$ and $\mathrm{MEC}(\mathcal{M})$ have the same maximal probabilities for prefix-independent properties.

$\mathrm{MEC}(\mathcal{M})$ has no end components. Hence:

$$\Pr^{\min}_{\mathrm{MEC}(\mathcal{M}),s}(\Diamond \mathit{Trap}) = 1$$

... transition probability matrix is contracting ...

Probabilistic model checking



$$\Pr^{\max}_{\mathcal{M},s}(\varphi) \;=\; \Pr^{\max}_{\mathcal{M}\otimes\mathcal{A},s'}\big(\Diamond accEC\big)$$

maximal probability to reach an accepting end component

Maximal reachability probabilities

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S
set $G \subseteq S$ of goal states

task: compute $x_s = \mathrm{Pr}_s^{\max}(\Diamond G) = \max_\sigma \mathrm{Pr}_s^\sigma(\Diamond G)$

Maximal reachability probabilities

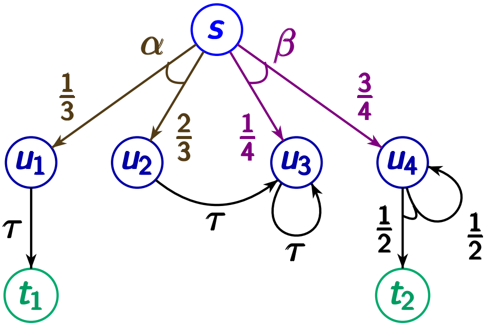
given: MDP $\mathcal{M}$ with state space S
set $G \subseteq S$ of goal states

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the least solution in $[0,1]^S$ of the equation system:

$$x_s = 1 \quad \text{if } s \in G$$
$$x_s = \max_{\alpha} \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} \quad \text{if } s \notin G$$

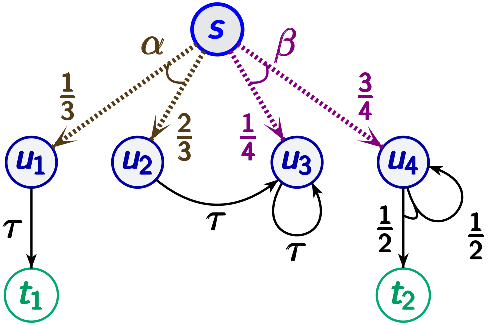
α ranges over all actions in $Act(s)$



$G = \{t_1, t_2\}$

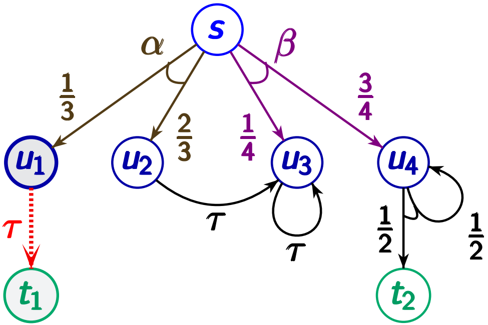
The vector $(x_s)_{s \in S}$ where $x_s = \Pr^{\max}_s(\Diamond G)$ is the least solution of

$$\begin{aligned} x_s &= 1 && \text{if } s \in G \\ x_s &= \max_{\alpha} \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} && \text{if } s \notin G \end{aligned}$$



$G = \{t_1, t_2\}$

$$x_s = \max\Big\{ \underbrace{\tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2}}_{\text{action } \alpha},\ \underbrace{\tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4}}_{\text{action } \beta} \Big\}$$

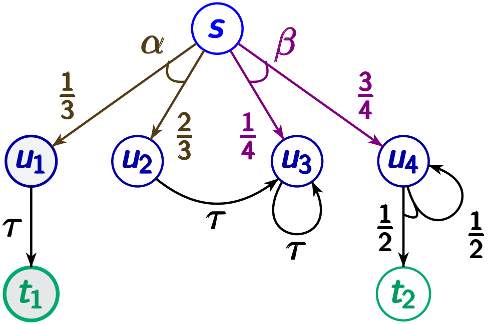


$G = \{t_1, t_2\}$

$$x_s = \max\{\ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2},\quad \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4}\ \}$$

$$x_{u_1} = x_{t_1}$$

↑
unique
successor
of u_1



$G = \{t_1, t_2\}$

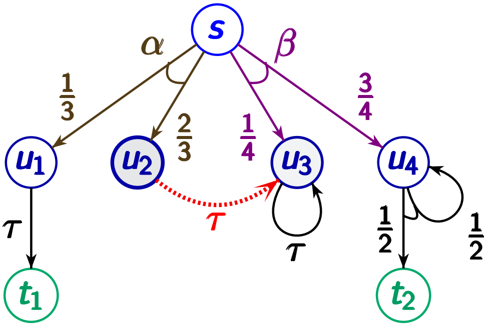
$$x_s = \max\left\{ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2},\quad \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4} \right\}$$

$$x_{u_1} = x_{t_1} = 1$$

↑

goal state

$t_1 \in G$



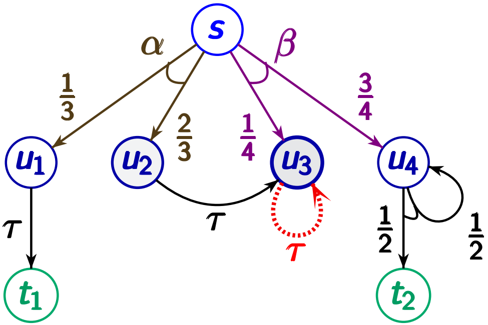
$G = \{t_1, t_2\}$

$$x_s = \max\{\ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2},\ \ \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4}\ \}$$

$$x_{u_1} = x_{t_1} = 1$$

$$x_{u_2} = x_{u_3}$$

↑ unique successor of state u_2



$G = \{t_1, t_2\}$

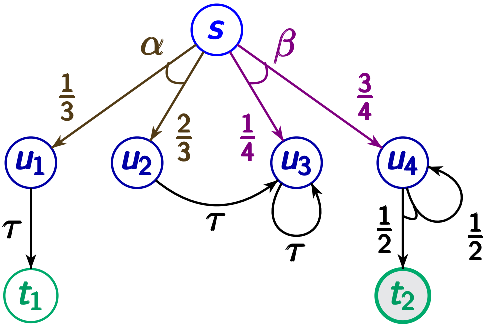
$$x_s = \max\{\ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2},\ \ \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4}\ \}$$

$$x_{u_1} = x_{t_1} = 1$$

$$x_{u_2} = x_{u_3} = 0$$

least solution of $x_{u_3} = x_{u_3}$

↑ unique successor of state u_2



$G = \{t_1, t_2\}$

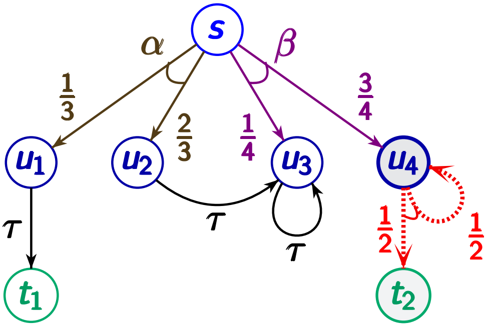
$$x_s = \max\{ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2},\ \ \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4} \}$$

$$x_{u_1} = x_{t_1} = 1$$

$$x_{u_2} = x_{u_3} = 0$$

least solution of $x_{u_3} = x_{u_3}$

$$x_{t_2} = 1$$



$G = \{t_1, t_2\}$

$$x_s = \max\{ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2},\ \ \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4} \}$$

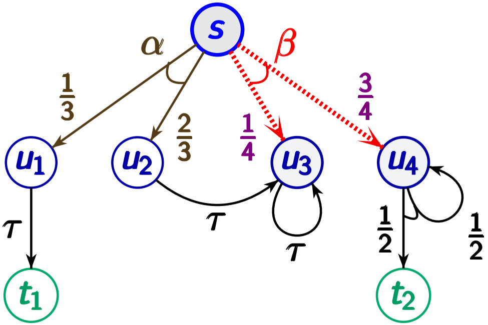
$$x_{u_1} = x_{t_1} = 1$$

$$x_{u_2} = x_{u_3} = 0$$

least solution of $x_{u_3} = x_{u_3}$

$$x_{t_2} = 1$$

$$x_{u_4} = \tfrac{1}{2}x_{u_4} + \tfrac{1}{2}x_{t_2} = \tfrac{1}{2}x_{u_4} + \tfrac{1}{2} = 1$$



$G = \{t_1, t_2\}$

$$x_s = \max\{ \tfrac{1}{3}x_{u_1} + \tfrac{2}{3}x_{u_2}, \quad \tfrac{1}{4}x_{u_3} + \tfrac{3}{4}x_{u_4} \} = \tfrac{3}{4}$$

$$x_{u_1} = x_{t_1} = 1$$

$$x_{u_2} = x_{u_3} = 0$$

least solution of $x_{u_3} = x_{u_3}$

$$x_{t_2} = 1$$

$$x_{u_4} = \tfrac{1}{2}x_{u_4} + \tfrac{1}{2}x_{t_2} = \tfrac{1}{2}x_{u_4} + \tfrac{1}{2} = 1$$

Maximal reachability probabilities

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the least solution in $[0,1]^S$ of the equation system:

$$x_s = 1 \quad \text{if } s \in G$$

$$x_s = \max_{\alpha} \sum_{t \in S} P(s, \alpha, t) \cdot x_t \quad \text{otherwise}$$

α ranges over all actions in $Act(s)$

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the least solution in $[0,1]^S$ of the equation system:

$$
\begin{array}{lcll}
x_s & = & 1 & \text{if } s \in G \\
x_s & = & 0 & \text{if } s \not\models \exists \Diamond G \\
x_s & = & \max\limits_{\alpha} \sum\limits_{t \in S} P(s,\alpha,t) \cdot x_t & \text{otherwise}
\end{array}
$$

α ranges over all actions in $Act(s)$

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max\limits_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the least solution in $[0,1]^S$ of the equation system:

$$
\begin{array}{lcll}
x_s & = & 1 & \text{if } s \in G \\
x_s & = & 0 & \text{if } s \not\models \exists \Diamond G \\
x_s & = & \max\limits_{\alpha} \sum\limits_{t \in S} P(s, \alpha, t) \cdot x_t & \text{otherwise}
\end{array}
$$

"Bellman equations"

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \mathrm{Pr}^{\max}_s(\Diamond G) = \max\limits_{\sigma} \mathrm{Pr}^{\sigma}_s(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the least solution in $[0,1]^S$ of the equation system:

$$
\begin{array}{lcll}
x_s & = & 1 & \text{if } s \in G \\
x_s & = & 0 & \text{if } s \not\models \exists\Diamond G \\
x_s & = & \max\limits_{\alpha} \sum\limits_{t \in S} P(s,\alpha,t) \cdot x_t & \text{otherwise}
\end{array}
$$

... induces an optimal MD-scheduler ...

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the least solution in $[0,1]^S$ of the equation system:

$$
\begin{array}{lcll}
x_s & = & 1 & \text{if } s \in G^* \\
x_s & = & 0 & \text{if } s \not\models \exists \Diamond G \\
x_s & = & \max\limits_{\alpha} \sum\limits_{t \in S} P(s, \alpha, t) \cdot x_t & \text{otherwise}
\end{array}
$$

pre-analysis: $G^* = \{ \, s \in S \; : \; x_s = 1 \, \}$

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr^{\max}_s(\Diamond G) = \max_\sigma \Pr^\sigma_s(\Diamond G)$

The vector $(x_s)_{s\in S}$ is the unique solution in $[0,1]^S$ of the equation system:

$$
\begin{array}{lcll}
x_s & = & 1 & \text{if } s \in G^* \\
x_s & = & 0 & \text{if } s \not\models \exists\Diamond G \\
x_s & = & \max\limits_{\alpha} \sum\limits_{t\in S} P(s,\alpha,t)\cdot x_t & \text{otherwise}
\end{array}
$$

if $\mathcal{M}$ has no end components

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr^{\max}_s(\Diamond G) = \max_\sigma \Pr^\sigma_s(\Diamond G)$

value iteration: $x_s = \lim_{n\to\infty} x_s^{(n)}$

$$
\begin{aligned}
x_s^{(n)} &= 1 && \text{if } s \in G^* \\
x_s^{(n)} &= 0 && \text{if } s \not\models \exists \Diamond G \\
x_s^{(n)} &= \max_\alpha \sum_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} && \text{else}
\end{aligned}
$$

if $\mathcal{M}$ has no end components

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr^{\max}_s(\Diamond G) = \max\limits_{\sigma} \Pr^{\sigma}_s(\Diamond G)$

value iteration: $x_s = \lim\limits_{n \to \infty} x_s^{(n)}$

$$
\begin{array}{lll}
x_s^{(n)} = 1 & & \text{if } s \in G^* \\
x_s^{(n)} = 0 & & \text{if } s \not\models \exists \Diamond G \\
x_s^{(n)} = \max\limits_{\alpha} \sum\limits_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} & & \text{else}
\end{array}
$$

if $\mathcal{M}$ has no end components or if $x_s^{(0)} \leqslant x_s$

Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

value iteration: $x_s = \lim_{n \to \infty} x_s^{(n)}$

$$
\begin{aligned}
x_s^{(n)} &= 1 && \text{if } s \in G^* \\
x_s^{(n)} &= 0 && \text{if } s \not\models \exists \Diamond G \\
x_s^{(n)} &= \max_{\alpha} \sum_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} && \text{else}
\end{aligned}
$$

... termination condition ?

Interval iteration

[Haddad/Monmege'14]

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr^{\max}_s(\Diamond G) = \max_\sigma \Pr^\sigma_s(\Diamond G)$

value iteration: $x_s = \lim_{n \to \infty} x_s^{(n)}$

$$
\begin{aligned}
x_s^{(n)} &= 1 && \text{if } s \in G^* \\
x_s^{(n)} &= 0 && \text{if } s \not\models \exists \Diamond G \\
x_s^{(n)} &= \max_\alpha \sum_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} && \text{else}
\end{aligned}
$$

... use lower and upper iteration in the MEC-quotient ...

Maximal reachability probabilities via LP

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_\sigma \Pr_s^\sigma(\Diamond G)$

The vector $(x_s)_{s\in S}$ is the least solution in $[0,1]^S$ of the linear constraints:

$$
\begin{array}{lcll}
x_s & = & 1 & \text{if } s \in G^* \\
x_s & = & 0 & \text{if } s \not\models \exists\Diamond G \\
x_s & \geqslant & \sum\limits_{t\in S} P(s,\alpha,t)\cdot x_t & \text{for } \alpha \in Act(s)
\end{array}
$$

Maximal reachability probabilities via LP

given: MDP $\mathcal{M}$ with state space S and $G \subseteq S$

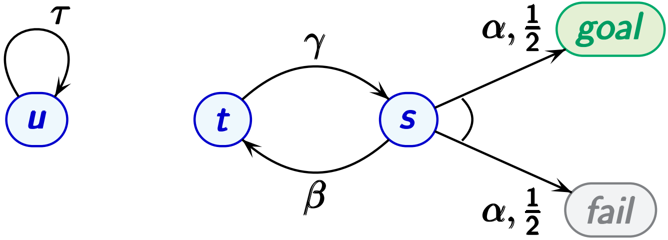
task: compute $x_s = \Pr_s^{\max}(\Diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\Diamond G)$

The vector $(x_s)_{s \in S}$ is the unique solution in $\mathbb{R}^S$ of the linear program:

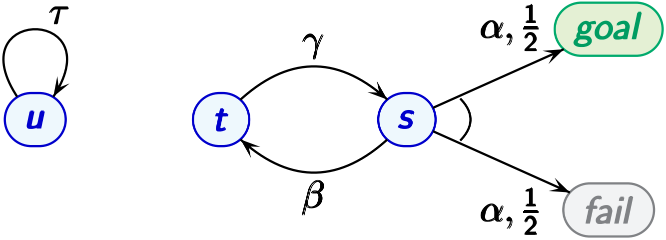
$$
\begin{array}{rcll}
x_s & = & 1 & \text{if } s \in G^* \\
x_s & = & 0 & \text{if } s \not\models \exists \Diamond G \\
x_s & \geqslant & \sum\limits_{t \in S} P(s, \alpha, t) \cdot x_t & \text{for } \alpha \in Act(s)
\end{array}
$$

where $\sum\limits_{s \in S} x_s$ is minimal

Least vs unique solution



Least vs unique solution

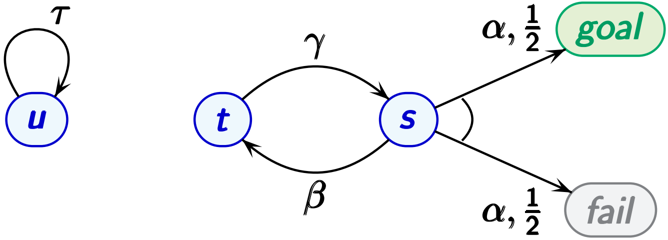


Bellmann equations:

$$x_u = x_u \qquad\qquad x_s = \max\left\{ x_t, \frac{1}{2} \right\}$$

$$x_t = x_s$$

Least vs unique solution

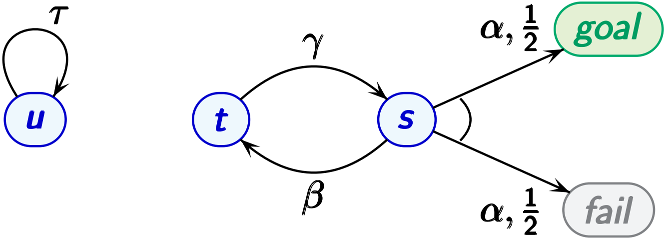


Bellmann equations:

$$x_u = 0 \qquad x_s = \max\left\{ x_t,\ \tfrac{1}{2} \right\}$$

as $u \not\models \exists\Diamond goal$ $\qquad x_t = x_s$

Least vs unique solution



Bellmann equations:

$$x_u = 0$$

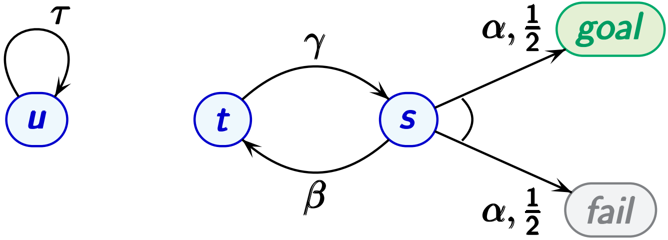
as $u \not\models \exists\Diamond goal$

$$x_s = \max\left\{ x_t, \tfrac{1}{2} \right\}$$
$$x_t = x_s$$

solutions:

$$x_t = x_s \geqslant \tfrac{1}{2}$$

Least vs unique solution



Bellmann equations:

$$x_u = 0$$

as $u \not\models \exists\Diamond goal$

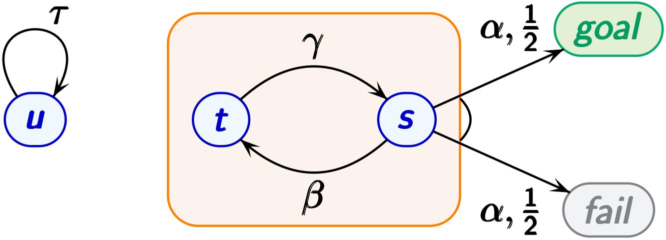
$$x_s = \max\left\{ x_t, \frac{1}{2} \right\}$$

$$x_t = x_s$$

least solution:

$$x_t = x_s = \frac{1}{2}$$

Least vs unique solution



Bellmann equations:

$$x_u = 0$$

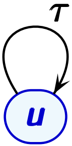
as $u \not\models \exists\Diamond goal$

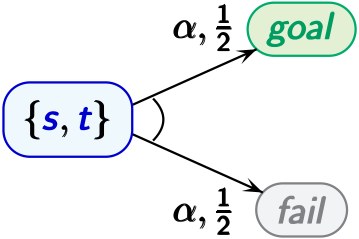
$$x_s = \max\left\{ x_t, \frac{1}{2} \right\}$$
$$x_t = x_s$$

least solution:

$$x_t = x_s = \frac{1}{2}$$

Least vs unique solution





Bellmann equations:

$$x_u = 0$$

as $u \not\models \exists\Diamond \mathit{goal}$

$$x_s = \max\left\{ x_t, \tfrac{1}{2} \right\}$$

$$x_t = x_s$$

unique solution:

$$x_{\{s,t\}} = \tfrac{1}{2}$$

Stochastic shortest/longest path problem

weighted
MDP $\mathcal{M}$

$\diamondplus$ *goal*
accumulated weight until
reaching a goal state

best- or worst-case expectation
$\mathbb{E}^{\mathsf{min}}(\diamondplus\, goal)$ or $\mathbb{E}^{\mathsf{max}}(\diamondplus\, goal)$

extrema over all schedulers

Stochastic shortest/longest path problem

weighted
MDP $\mathcal{M}$

$\oplus$ *goal*
accumulated weight until
reaching a goal state

requirement for $\mathcal{M}$:
$\Pr^{\min}(\Diamond \textit{goal}) = 1$

best- or worst-case expectation
$\mathbb{E}^{\min}(\oplus \textit{goal})$ or $\mathbb{E}^{\max}(\oplus \textit{goal})$

extrema over all schedulers

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\min}_s(\Diamond G) = 1$ for all states s

task: compute $x_s = \underbrace{\mathbb{E}^{\max}_s(\oplus G)}$

"stochastic longest path"

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\min}_s(\Diamond G) = 1$ for all states s

task: compute $x_s = \underbrace{\mathbb{E}^{\max}_s(\oplus G)}$
"stochastic longest path"

random variable $\oplus G : MaxPaths \to \mathbb{Z}$

if $\pi = s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} \ldots$ where $s_n \in G$, $s_0, \ldots, s_{n-1} \notin G$:

$$(\oplus G)(\pi) = wgt(s_0 \xrightarrow{\alpha_0} \ldots \xrightarrow{\alpha_n} s_n)$$

if $\pi \not\models \Diamond G$ then $(\oplus G)(\pi) = \bot$ "undefined"

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr_s^{\min}(\Diamond G) = 1$ for all states s

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$

The vector $(x_s)_{s \in S}$ is the unique solution in $\mathbb{R}^S$ of:

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s = \max_{\alpha \in Act(s)} \Big(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \Big)$$

"Bellman equations"

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\min}_s(\Diamond G) = 1$ for all states s

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$

The vector $(x_s)_{s \in S}$ is the unique solution in $\mathbb{R}^S$ of:

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s = \max_{\alpha \in Act(s)} \Big(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \Big)$$

... fixpoint operator is a contracting map ...

[Bertsekas/Tsitsiklis'91]

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr_s^{\min}(\Diamond G) = 1$ for all states s

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$

The vector $(x_s)_{s \in S}$ is the unique solution in $\mathbb{R}^S$ of:

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s = \max_{\alpha \in Act(s)} \Big(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \Big)$$

... induces an optimal MD-scheduler ...

[Bertsekas/Tsitsiklis'91]

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\min}_s(\lozenge G) = 1$ for all states s

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$

The vector $(x_s)_{s \in S}$ is the unique solution in $\mathbb{R}^S$ of:

If $s \in G$ then $x_s^{(n)} = 0$. Otherwise:

$$x_s^{(n)} = \max_{\alpha \in Act(s)} \Big(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} \Big)$$

value iteration (arbitrary starting vector)

[Bertsekas/Tsitsiklis'91]

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\min}_s(\Diamond G) = 1$ for all states s

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$

The vector $(x_s)_{s \in S}$ is the unique solution in $\mathbb{R}^S$ of:

If $s \in G$ then $x_s = 0$. Otherwise, for $\alpha \in Act(s)$:

$$x_s \geqslant wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t$$

where $\sum_{s \in S} x_s$ is minimal

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles
* mean-payoff
* expected accumulated weights

PCTL* over MDPs

[Bianco/de Alfaro'95]

- syntax of state and path formulas as for PCTL* over Markov chains
- probability operator $\mathbb{P}_{\mathbf{I}}(\ldots)$ ranges over all schedulers

PCTL* over MDPs

[Bianco/de Alfaro'95]

state formulas:

$$\Phi \ ::= \ \mathit{true} \ \big| \ a \ \big| \ \Phi_1 \wedge \Phi_2 \ \big| \ \neg\Phi \ \big| \ \mathbb{P}_{\mathrm{I}}(\varphi)$$

path formulas:

$$\varphi \ ::= \ \Phi \ \big| \ \varphi_1 \wedge \varphi_2 \ \big| \ \neg\varphi \ \big| \ \bigcirc\varphi \ \big| \ \varphi_1 \,\mathsf{U}\, \varphi_2$$

PCTL* over MDPs

[Bianco/de Alfaro'95]

state formulas:

$$\Phi \;::=\; \mathit{true} \;\big|\; a \;\big|\; \Phi_1 \wedge \Phi_2 \;\big|\; \neg\Phi \;\big|\; \mathbb{P}_I(\varphi)$$

path formulas:

$$\varphi \;::=\; \Phi \;\big|\; \varphi_1 \wedge \varphi_2 \;\big|\; \neg\varphi \;\big|\; \bigcirc\varphi \;\big|\; \varphi_1 \,\mathsf{U}\, \varphi_2$$

given an MDP $\mathcal{M}$, define by structural induction:

- a satisfaction relation $\models$ for states s in $\mathcal{M}$ and PCTL* state formulas Φ
- a satisfaction relation $\models$ for infinite paths π in $\mathcal{M}$ and PCTL* path formulas φ

Satisfaction relation for PCTL* state formulas

$s \models \mathit{true}$

$s \models a$ iff $a \in L(s)$

$s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1$ and $s \models \Phi_2$

$s \models \neg\Phi$ iff $s \not\models \Phi$

$s \models \mathbb{P}_{\mathrm{I}}(\varphi)$ iff for all schedulers σ:

$$\Pr^{\sigma}\{\pi \in \mathit{Paths}(s) : \pi \models \varphi\} \in \mathrm{I}$$

Satisfaction relation for PCTL* state formulas

$s \models \mathit{true}$

$s \models a$ iff $a \in L(s)$

$s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1$ and $s \models \Phi_2$

$s \models \neg\Phi$ iff $s \not\models \Phi$

$s \models \mathbb{P}_{\mathrm{I}}(\varphi)$ iff for all schedulers σ:

$$\Pr^{\sigma}\{\pi \in \mathit{Paths}(s) : \pi \models \varphi\} \in \mathrm{I}$$

↑

probability measure in the Markov chain induced by σ

Satisfaction relation for PCTL* state formulas

$s \models \mathit{true}$

$s \models a$ iff $a \in L(s)$

$s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1$ and $s \models \Phi_2$

$s \models \neg\Phi$ iff $s \not\models \Phi$

$s \models \mathbb{P}_{\mathrm{I}}(\varphi)$ iff for all schedulers σ:

$$\Pr^{\sigma}\{\pi \in \mathit{Paths}(s) : \pi \models \varphi\} \in \mathrm{I}$$

probability measure in the Markov chain induced by σ

semantics of path formulas as for Markov chains

PCTL* model checking for MDP

PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, Act, P, AP, L, s_0)$

PCTL* state formula Φ

task: check whether $s_0 \models \Phi$

PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, Act, P, AP, L, s_0)$
PCTL* state formula Φ

task: check whether $s_0 \models \Phi$

main procedure as for PCTL* over Markov chains:

recursively compute the satisfaction sets

$$Sat(\Psi) \;=\; \{\, s \in S \,:\, s \models \Psi \,\}$$

for all state subformulas Ψ of Φ

PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, Act, P, AP, L, s_0)$
PCTL* state formula Φ

task: check whether $s_0 \models \Phi$

main procedure as for PCTL* over Markov chains:

recursively compute the satisfaction sets

$$Sat(\Psi) = \{ s \in S : s \models \Psi \}$$

for all state subformulas Ψ of Φ

treatment of the propositional logic fragment: $\surd$

Treatment of probability operator

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

- compute the maximal probabilities for φ

$$\mathrm{Pr}_s^{\max}(\varphi) \;=\; \sup_{\mathfrak{D}} \mathrm{Pr}^{\mathfrak{D}}\{\, \pi \in \mathit{Paths}(s) \, : \, \pi \models \varphi \,\}$$

for all states s

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

- compute the maximal probabilities for φ

$$\mathrm{Pr}_s^{\max}(\varphi) = \max_{D} \mathrm{Pr}^{D}\{\, \pi \in Paths(s) : \pi \models \varphi \,\}$$

for all states s

↑

there exists optimal
finite-memory schedulers

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

- compute the maximal probabilities for φ

  $$\Pr_s^{\max}(\varphi) = \max_{D} \Pr^{D}\{\, \pi \in Paths(s) \,:\, \pi \models \varphi \,\}$$

  for all states s

- return $\{\, s \in S \,:\, \Pr_s^{\max}(\varphi) \leqslant p \,\}$

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

- compute the maximal probabilities for φ

  $$\Pr_s^{\max}(\varphi) \;=\; \max_{D} \Pr^{D}\{\, \pi \in Paths(s) \;:\; \pi \models \varphi \,\}$$

  for all states s
- return $\{\, s \in S \;:\; \Pr_s^{\max}(\varphi) \leqslant p \,\}$

lower probability bounds $\mathbb{P}_{\geqslant p}(\varphi)$ or $\mathbb{P}_{>p}(\varphi)$

analogous, but minimal probabilities for φ

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

compute the maximal probabilities for φ

$$\Pr_s^{\max}(\varphi) = \max_{D} \Pr^{D}\{\pi \in Paths(s) : \pi \models \varphi\}$$

special case: $\varphi = \Diamond \Psi$

↑

reachability condition

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

compute the maximal probabilities for φ

$$\mathrm{Pr}^{\max}_s(\varphi) \; = \; \max_{D} \mathrm{Pr}^{D}\{\, \pi \in \mathit{Paths}(s) \,:\, \pi \models \varphi \,\}$$

special case: $\varphi = \Diamond \Psi$

compute $\mathrm{Pr}^{\max}_s(\Diamond\Psi)$ by solving a linear program

↑
maximal
reachability
probabilities

Treatment of probability operator

upper probability bounds $\mathbb{P}_{\leqslant p}(\varphi)$ or $\mathbb{P}_{<p}(\varphi)$

compute the maximal probabilities for φ

$$\Pr_s^{\max}(\varphi) = \max_{D} \Pr^{D}\{\pi \in Paths(s) : \pi \models \varphi\}$$

special case: $\varphi = \Diamond \Psi$

compute $\Pr_s^{\max}(\Diamond \Psi)$ by solving a linear program

general case:

via determininistic automaton $\mathcal{A}$ for φ and maximal reachability probabilities in $\mathcal{M} \times \mathcal{A}$

PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, Act, P, \ldots)$

PCTL* state formula $\mathbb{P}_{\leqslant p}(\varphi)$

task: compute $Sat\big(\mathbb{P}_{\leqslant p}(\varphi)\big)$

PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, Act, P, \ldots)$
PCTL* state formula $\mathbb{P}_{\leqslant p}(\varphi)$

task: compute $Sat\big(\mathbb{P}_{\leqslant p}(\varphi)\big)$

method: compute $x_s = \Pr_s^{\max}(\varphi)$ via a reduction to the probabilistic reachability problem

PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, Act, P, \ldots)$
PCTL* state formula $\mathbb{P}_{\leqslant p}(\varphi)$

task: compute $Sat\big(\mathbb{P}_{\leqslant p}(\varphi)\big)$

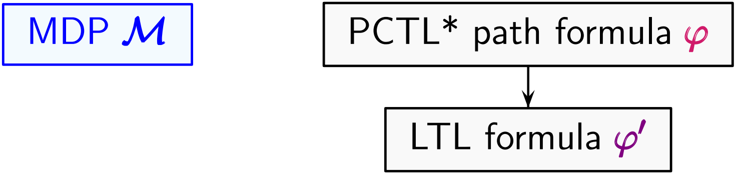
method: compute $x_s = \Pr^{\max}_s(\varphi)$ via a reduction to the probabilistic reachability problem

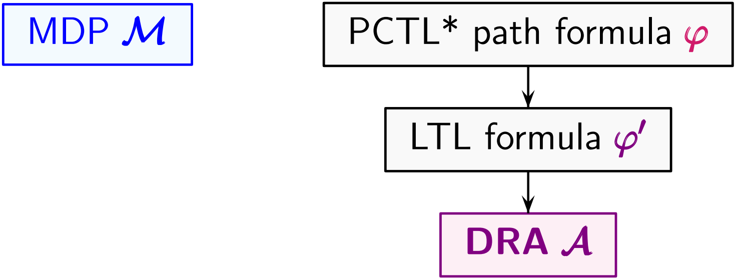
> using DRA $\mathcal{A}$ for φ and
> linear program for $\mathcal{M} \times \mathcal{A}$

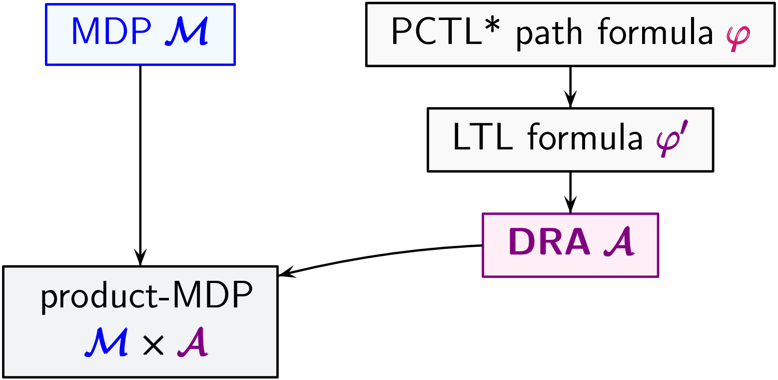
DRA: deterministic Rabin automaton

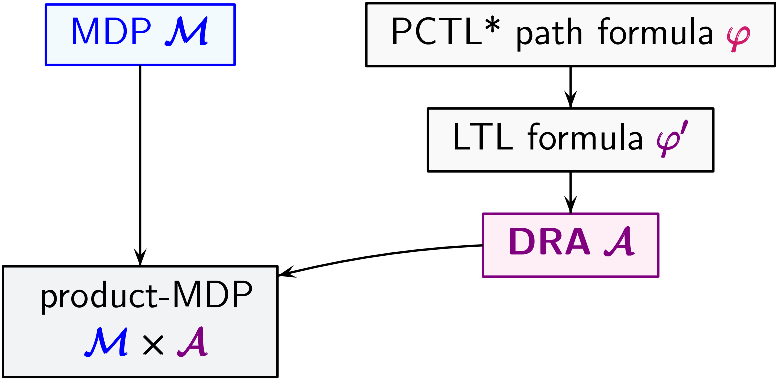
MDP $\mathcal{M}$

PCTL* path formula φ

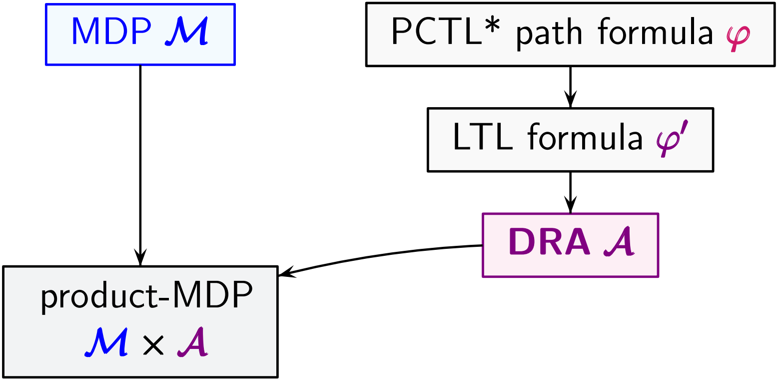

MDP $\mathcal{M}$
PCTL* path formula φ
LTL formula φ'


MDP $\mathcal{M}$
PCTL* path formula φ
LTL formula φ'
DRA $\mathcal{A}$


MDP $\mathcal{M}$
PCTL* path formula φ
LTL formula φ'
DRA $\mathcal{A}$
product-MDP
$\mathcal{M} \times \mathcal{A}$



$$\Pr^{\max}_{\mathcal{M}}(\varphi) \;=\; \Pr^{\max}_{\mathcal{M}\times\mathcal{A}}\Big(\underbrace{\bigvee_i (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i)}_{\text{acceptance condition of } \mathcal{A}} \Big)$$



$$\Pr^{\max}_{\mathcal{M}}(\varphi) = \Pr^{\max}_{\mathcal{M}\times\mathcal{A}}\Big(\bigvee_i (\Diamond\Box\neg L_i \wedge \Box\Diamond U_i) \Big)$$

$$= \Pr^{\max}_{\mathcal{M}\times\mathcal{A}}\big(\Diamond accMEC \big)$$

Lower probability bounds

given: MDP $\mathcal{M} = (S, Act, P, \ldots)$

PCTL* formula $\mathbb{P}_{\geqslant p}(\varphi)$

task: compute $Sat\big(\mathbb{P}_{\geqslant p}(\varphi)\big)$

Lower probability bounds

given: MDP $\mathcal{M} = (S, Act, P, \ldots)$

PCTL* formula $\mathbb{P}_{\geqslant p}(\varphi)$

task: compute $Sat\big(\mathbb{P}_{\geqslant p}(\varphi)\big)$

simple fact: for each scheduler D and state s:

$$\Pr_s^D(\varphi) \;=\; 1 - \Pr_s^D(\neg\varphi)$$

... duality of lower and upper probability bounds

Lower probability bounds

given: MDP $\mathcal{M} = (S, Act, P, \ldots)$

PCTL* formula $\mathbb{P}_{\geqslant p}(\varphi)$

task: compute $Sat\big(\mathbb{P}_{\geqslant p}(\varphi)\big)$

simple fact: for each scheduler D and state s:

$$\mathrm{Pr}_s^D(\varphi) \;=\; 1 - \mathrm{Pr}_s^D(\neg\varphi)$$

... duality of lower and upper probability bounds

For each state s and PCTL* path formula φ:

$$\mathrm{Pr}_s^{\min}(\varphi) \;=\; 1 - \mathrm{Pr}_s^{\max}(\neg\varphi)$$

Complexity of PCTL/PCTL* model checking

Complexity of PCTL/PCTL* model checking

	PCTL	PCTL*
Markov chain	***PTIME*** [Hansson/Jonsson'94]	***PSPACE***-complete [Vardi/Wolper'86]
Markov decision process	***PTIME*** [Bianco/deAlfaro'95]	2***EXP***-complete [Courcoubetis/Yannakakis'88]

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles
* mean-payoff
* expected accumulated weights

Conditional probabilities for MDP

for Markov decision processes:

$$\Pr^{\max}_{\mathcal{M},s}(\varphi \mid \psi) \;=\; \max_{\sigma} \frac{\Pr^{\sigma}_{s}(\varphi \wedge \psi)}{\Pr^{\sigma}_{s}(\psi)}$$

↑

all schedulers σ
with $\Pr^{\sigma}_{s}(\psi) > 0$

Conditional probabilities for MDP

for Markov decision processes:

$$\Pr^{\max}_{\mathcal{M},s}(\varphi \mid \psi) \;=\; \max_{\sigma} \frac{\Pr^{\sigma}_{s}(\varphi \wedge \psi)}{\Pr^{\sigma}_{s}(\psi)}$$

exponential-time procedure for PCTL [Andrés/Rossum'08]
even for reachability $\varphi = \Diamond F$, $\psi = \Diamond G$

PCTL probabilistic computation tree logic

Conditional probabilities for MDP

for Markov decision processes:

$$\Pr^{\max}_{\mathcal{M},s}(\varphi \mid \psi) \;=\; \max_{\sigma} \frac{\Pr^{\sigma}_{s}(\varphi \wedge \psi)}{\Pr^{\sigma}_{s}(\psi)}$$

exponential-time procedure for PCTL [Andrés/Rossum'08]

even for reachability $\varphi = \Diamond F$, $\psi = \Diamond G$

---

transformation-based approach for LTL

MDP $\mathcal{M}$ $\rightsquigarrow$ MDP $\mathcal{M}_{\varphi|\psi}$ of linear size for reachability

$$\Pr^{\max}_{\mathcal{M},s}(\varphi \mid \psi) \;=\; \Pr^{\max}_{\mathcal{M}_{\varphi|\psi},s}(\varphi')$$

[Baier/Klein/Klüppelholz/Märcker'14]

Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$

objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$

objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

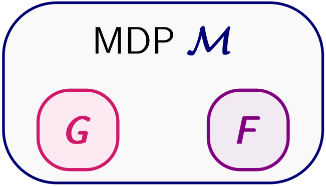
step 1: generate a normal form MDP $\mathcal{M}'$

Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$

objective $\varphi = \lozenge F$, condition $\psi = \lozenge G$

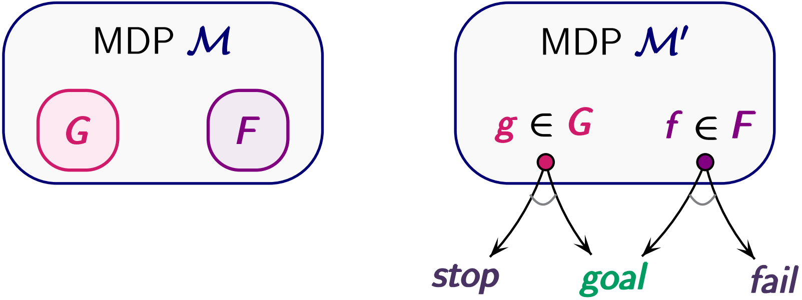
step 1: generate a normal form MDP $\mathcal{M}'$



Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$
objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

step 1: generate a normal form MDP $\mathcal{M}'$



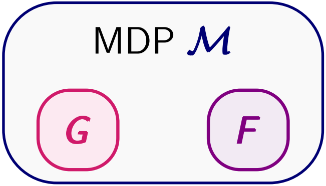
three fresh trap states

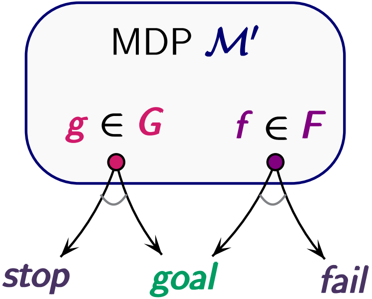
Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and $F, G \subseteq S$

objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

step 1: generate a normal form MDP $\mathcal{M}'$





$$P'(g, goal) = \Pr^{\max}_{\mathcal{M},g}(\Diamond F)$$

$$P'(g, stop) = 1 - \Pr^{\max}_{\mathcal{M},g}(\Diamond F)$$

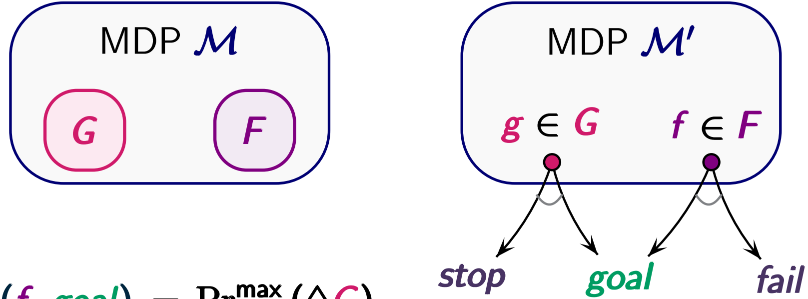
three fresh trap states

Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$

objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

step 1: generate a normal form MDP $\mathcal{M}'$



$$P'(f, goal) = \Pr^{\max}_{\mathcal{M},f}(\Diamond G)$$

$$P'(f, fail) = 1 - \Pr^{\max}_{\mathcal{M},f}(\Diamond G)$$

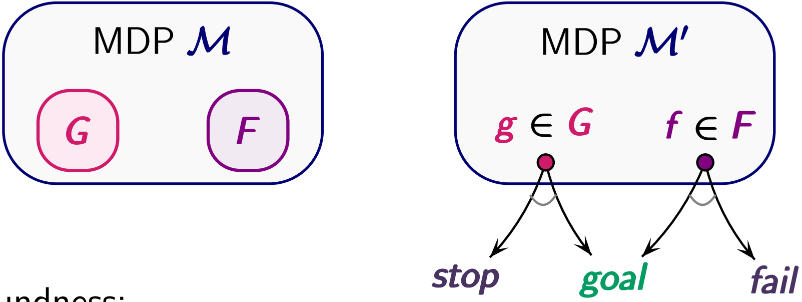
three fresh trap states

Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$

objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

step 1: generate a normal form MDP $\mathcal{M}'$



soundness:

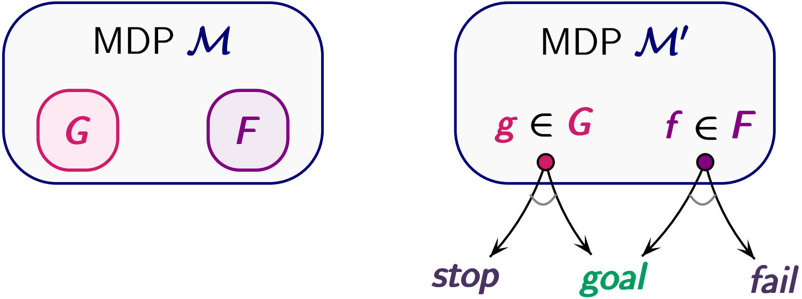
$$\Pr^{\max}_{\mathcal{M},s}(\Diamond F \mid \Diamond G) = \Pr^{\max}_{\mathcal{M}',s}(\Diamond goal \mid \Diamond(goal \vee stop))$$

Transformation-based approach for MDP

given: MDP $\mathcal{M} = (S, P)$ and F, $G \subseteq S$

objective $\varphi = \Diamond F$, condition $\psi = \Diamond G$

step 1: generate a normal form MDP $\mathcal{M}'$



step 2: normal form MDP $\mathcal{M}' \rightsquigarrow$ MDP $\mathcal{M}''$ s.t. ...

Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

$$\mathrm{Pr}^{\max}_{\mathcal{M}',s_{init}}\big(\Diamond goal \mid \Diamond(goal \vee stop) \big) = \mathrm{Pr}^{\max}_{\mathcal{M}'',s_{init}}\big(\Diamond goal \big)$$

Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

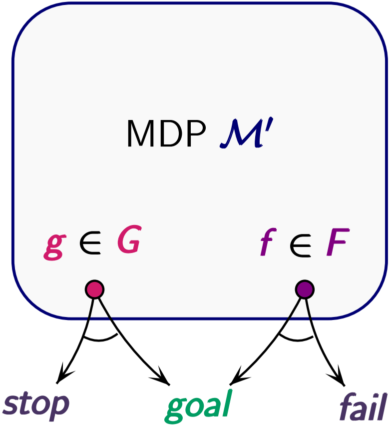
$$\Pr^{\max}_{\mathcal{M}',s_{init}}\big(\ \Diamond goal \mid \Diamond(goal \vee stop)\ \big) = \Pr^{\max}_{\mathcal{M}'',s_{init}}\big(\ \Diamond goal\ \big)$$

idea: $\mathcal{M}''$ redistributes the probabilities of the paths π with $\pi \not\models \Diamond(goal \vee stop)$

Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

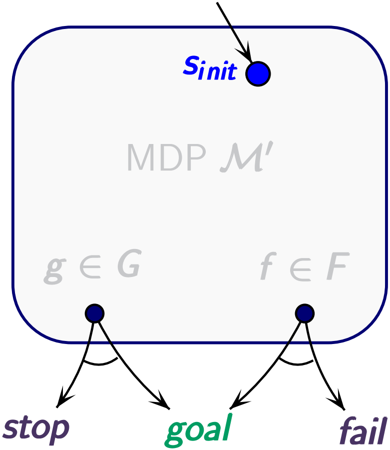
$$\Pr^{\max}_{\mathcal{M}',s_{init}}\big(\ \Diamond goal \mid \Diamond(goal \vee stop)\ \big) = \Pr^{\max}_{\mathcal{M}'',s_{init}}\big(\ \Diamond goal\ \big)$$



Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

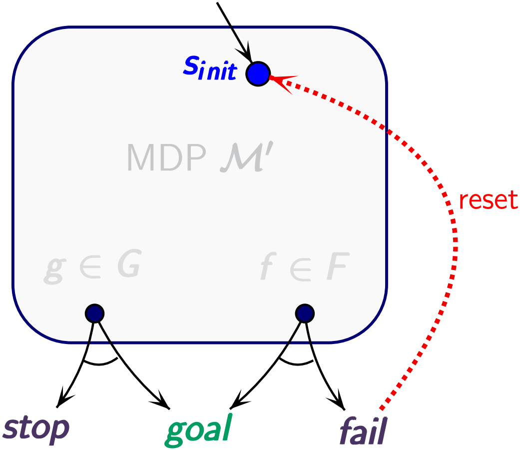
$$\Pr^{\max}_{\mathcal{M}',s_{init}}\big(\,\Diamond goal \mid \Diamond(goal \vee stop)\,\big) = \Pr^{\max}_{\mathcal{M}'',s_{init}}\big(\,\Diamond goal\,\big)$$



Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

$$\Pr^{\max}_{\mathcal{M}',s_{init}}(\,\Diamond goal \mid \Diamond(goal \vee stop)\,) = \Pr^{\max}_{\mathcal{M}'',s_{init}}(\,\Diamond goal\,)$$

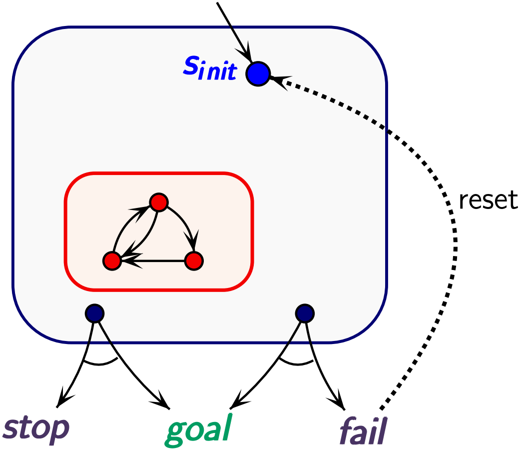


Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

$$\Pr^{\max}_{\mathcal{M}',s_{init}}(\ \Diamond goal \mid \Diamond(goal \vee stop)\ ) = \Pr^{\max}_{\mathcal{M}'',s_{init}}(\ \Diamond goal\ )$$

How to deal with states that might never reach one of the trap states?

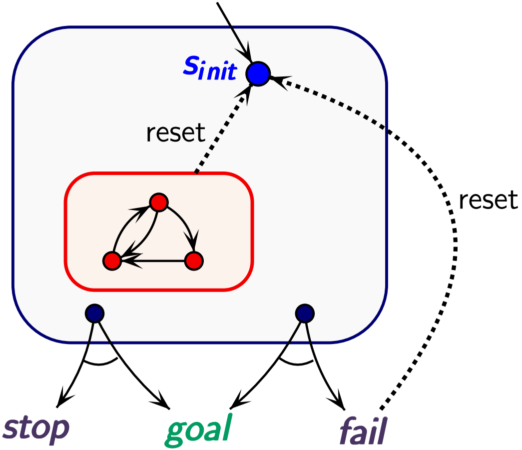


Transformation-based approach for MDP

step 2: normal form MDP $\mathcal{M}'$ $\rightsquigarrow$ MDP $\mathcal{M}''$ s.t.

$$\Pr^{\max}_{\mathcal{M}',s_{init}}\big(\Diamond goal \mid \Diamond(goal \vee stop)\big) = \Pr^{\max}_{\mathcal{M}'',s_{init}}\big(\Diamond goal\big)$$

add reset-transitions from all end components that do not contain a trap state



Summary: conditional probabilities for MDP

for Markov decision processes:

$$\Pr^{\max}_{\mathcal{M},s}(\varphi \mid \psi) \;=\; \max_{\sigma} \frac{\Pr^{\sigma}_{s}(\varphi \wedge \psi)}{\Pr^{\sigma}_{s}(\psi)}$$

computation by reduction to unconditional probabilities

* reset-mechanism for reachability objective and condition
* generalization for LTL objectives/conditions via ω-automata

Summary: conditional probabilities for MDP

for Markov decision processes:

$$\mathrm{Pr}^{\max}_{\mathcal{M},s}(\varphi \,|\, \psi) \;=\; \max_{\sigma} \frac{\mathrm{Pr}^{\sigma}_{s}(\varphi \wedge \psi)}{\mathrm{Pr}^{\sigma}_{s}(\psi)}$$

computation by reduction to unconditional probabilities

* reset-mechanism for reachability objective and condition
* generalization for LTL objectives/conditions via ω-automata

complexity-theoretic results ... as for unconditional probabilities

- model-checking problem for conditional PCTL in P
- threshold problem for LTL objectives/conditions is 2EXPTIME-complete

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles
* mean-payoff
* expected accumulated weights

Quantiles

well-known in statistics:

If f is a real-valued random variable and $q \in [0,1[$ a probability threshold then

$$\inf \left\{ r \in \mathbb{R} \,:\, \Pr\{f \leqslant r\} > q \right\}$$

is the q-quantile of f.

note: the fct. $\mathbb{R} \to [0,1]$, $r \mapsto \Pr\{f \leqslant r\}$ is increasing

Quantiles

well-known in statistics:

If f is a real-valued random variable and $q \in [0,1[$ a probability threshold then

$$\inf \left\{ r \in \mathbb{R} : \Pr\{f \leqslant r\} > q \right\}$$

is the q-quantile of f.

... can be very useful for the analysis of systems ...

Examples for quantiles in Markov chains

energy-aware job scheduling:

$$\underbrace{\Pr_s\big(\lozenge^{\leqslant e}_{\geqslant u}\, goal\,\big)}$$

probability to reach the goal,
when the energy consumption is at most e
and the gained utility is at least u

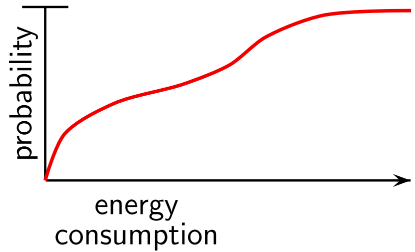
Examples for quantiles in Markov chains

energy-aware job scheduling:

$$\underbrace{\Pr_s\left(\Diamond^{\leqslant e}_{\geqslant u}\ goal\right)}$$

probability to reach the goal,
when the energy consumption is at most e
and the gained utility is at least u

for fixed utility value u



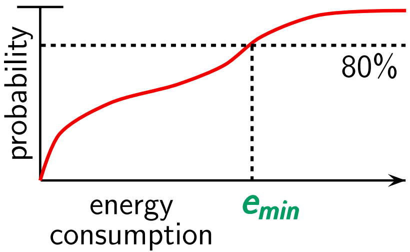
Examples for quantiles in Markov chains

energy-aware job scheduling:

$$\mathbf{min}\ \left\{\ e \in \mathbb{N} :\ \underbrace{\Pr_s\left(\Diamond^{\leqslant e}_{\geqslant u}\ goal\right)}_{} > 0.8\ \right\}$$

probability to reach the goal,
when the energy consumption is at most e
and the gained utility is at least u

for fixed utility value u

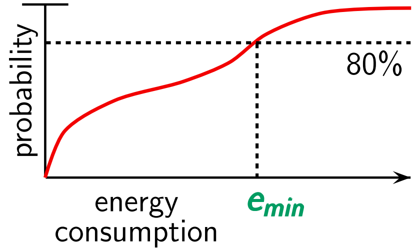


Examples for quantiles in Markov chains

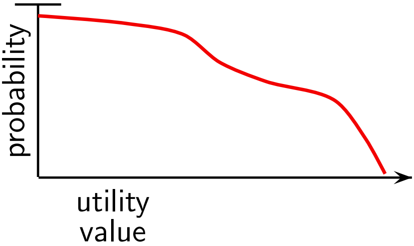
energy-aware job scheduling:

$$\min \left\{ e \in \mathbb{N} : \underbrace{\Pr_s\left(\Diamond^{\leqslant e}_{\geqslant u}\, goal \right)}_{\substack{\text{probability to reach the goal,} \\ \text{when the energy consumption is at most e} \\ \text{and the gained utility is at least u}}} > 0.8 \right\}$$

for fixed utility value u



for fixed energy budget e



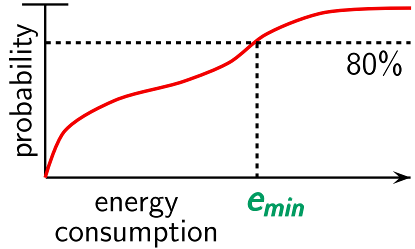
Examples for quantiles in Markov chains

energy-aware job scheduling:

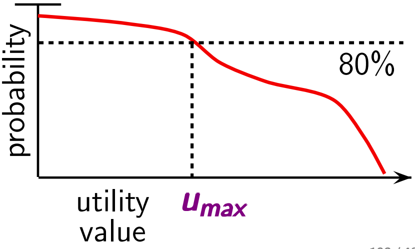
$$\min \left\{ e \in \mathbb{N} : \Pr_s\left(\Diamond^{\leqslant e}_{\geqslant u}\, goal \right) > 0.8 \right\}$$

$$\max \left\{ u \in \mathbb{N} : \Pr_s\left(\Diamond^{\leqslant e}_{\geqslant u}\, goal \right) > 0.8 \right\}$$

for fixed utility value u



for fixed energy budget e



Quantiles in Markovian models

Markov chains:

$$\min \{ r \in \mathbb{N} : \Pr_s(\Diamond^{\leqslant r} goal) > 0.8 \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s(\Diamond_{\geqslant r} goal) > 0.8 \}$$

Markov decision processes:

$$\min \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond^{\leqslant r} goal) > 0.8 \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond_{\geqslant r} goal) > 0.8 \}$$

Quantiles in Markovian models

Markov chains:

$$\min \{ r \in \mathbb{N} : \Pr_s(\Diamond^{\leqslant r} \mathit{goal}) > 0.8 \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s(\Diamond_{\geqslant r} \mathit{goal}) > 0.8 \}$$

Markov decision processes:

$$\min \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond^{\leqslant r} \mathit{goal}) > 0.8 \}$$

$$\min \{ r \in \mathbb{N} : \Pr_s^{\min}(\Diamond^{\leqslant r} \mathit{goal}) > 0.8 \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond_{\geqslant r} \mathit{goal}) > 0.8 \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s^{\min}(\Diamond_{\geqslant r} \mathit{goal}) > 0.8 \}$$

Computing quantiles in MDP

Computing quantiles in MDP

e.g., existential quantiles

$$\min \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond^{\leqslant r} G) > q \right\}$$

$$\max \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond_{\geqslant r} G) > q \right\}$$

---

results on the computation of quantiles:

- qualitative quantiles in poly-time
- EXP-compl. for quantitative quantiles
- iterative LP-approach for quantitative quantiles

Computing quantiles in MDP

e.g., existential quantiles

$$\min \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond^{\leqslant r} G) = 1 \right\}$$
$$\max \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond_{\geqslant r} G) > 0 \right\}$$

---

results on the computation of quantiles:

- qualitative quantiles in poly-time [Ummels/Baier'13]
- EXP-compl. for quantitative quantiles
- iterative LP-approach for quantitative quantiles

Computing quantiles in MDP

e.g., existential quantiles

$$\min \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond^{\leqslant r} G) > q \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond_{\geqslant r} G) > q \}$$

---

results on the computation of quantiles:

- qualitative quantiles in poly-time [Ummels/Baier'13]
- EXP-compl. for quantitative quantiles [Haase/Kiefer'15]
- iterative LP-approach for quantitative quantiles

Computing quantiles in MDP

e.g., existential quantiles

$$\min \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond^{\leqslant r} G) > q \}$$

$$\max \{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond_{\geqslant r} G) > q \}$$

---

results on the computation of quantiles:

- qualitative quantiles in poly-time [Ummels/Baier'13]
- EXP-compl. for quantitative quantiles [Haase/Kiefer'15]
- iterative LP-approach for quantitative quantiles
  [Ummels/Baier'13] [Baier/Daum/Dubslaff/Klein/Klüppelholz'14]

Computing quantitative quantiles

$$\mathrm{qu}(s_0) \;=\; \min\left\{\, r \in \mathbb{N} \,:\, \Pr^{\max}_{s_0}(\Diamond^{\leqslant r} G) > q \,\right\}$$

existential quantile for

- upper reward-bounded reachability
- lower probability bound

$$\Pr^{\max}_{s}(\varphi) > p \quad \text{iff} \quad \begin{cases} \text{there exists a scheduler } \sigma \\ \text{with } \Pr^{\sigma}_{s}(\varphi) > p \end{cases}$$

Computing quantitative quantiles

$$\mathrm{qu}(s_0) \;=\; \min\left\{\, r \in \mathbb{N} \,:\, \Pr^{\max}_{s_0}(\Diamond^{\leqslant r} G) > q \,\right\}$$

1. compute $p \;=\; \Pr^{\max}_{s_0}(\,\Diamond G\,)$
2. return $\mathrm{qu}(s_0) = \infty$ if $p \leqslant q$
3. ...

Computing quantitative quantiles

$$\mathrm{qu}(s_0) \;=\; \min\left\{\, r \in \mathbb{N} \,:\, \underbrace{\Pr^{\max}_{s_0}(\Diamond^{\leqslant r} G)}_{p_{s_0,r}} > q \,\right\}$$

1. compute $p \;=\; \Pr^{\max}_{s_0}(\Diamond G)$
2. return $\mathrm{qu}(s_0) = \infty$ if $p \leqslant q$
3. for $r = 0, 1, 2, \ldots$ compute the values $p_{s,r}$ for all states $s \in S$ and return the smallest value r such that $p_{s_0,r} > q$

Computing quantitative quantiles

$$\mathrm{qu}(s_0) \;=\; \min\left\{\, r \in \mathbb{N} \,:\, \underbrace{\Pr^{\max}_{s_0}(\Diamond^{\leqslant r} G)}_{p_{s_0,r}} > q \,\right\}$$

1. compute $p \;=\; \Pr^{\max}_{s_0}(\,\Diamond G\,)$
2. return $\mathrm{qu}(s_0) = \infty$ if $p \leqslant q$
3. for $r = 0, 1, 2, \ldots$ compute the values $p_{s,r}$ for all states $s \in S$ and return the smallest value r such that $p_{s_0,r} > q$

exponential bound on the number of required iterations

(in practice much faster)

Computing quantitative quantiles

$$\mathrm{qu}(s_0) \;=\; \min\left\{\, r \in \mathbb{N} \;:\; \underbrace{\Pr^{\max}_{s_0}(\Diamond^{\leqslant r} G)}_{p_{s_0,r}} > q \,\right\}$$

1. compute $p \;=\; \Pr^{\max}_{s_0}(\Diamond G)$
2. return $\mathrm{qu}(s_0) = \infty$ if $p \leqslant q$
3. for $r = 0, 1, 2, \ldots$ compute the values $p_{s,r}$ for all states $s \in S$ and return the smallest value r such that $p_{s_0,r} > q$

computation of $p_{s,r}$ by an iterative linear-programming approach with back propagation

linear program for the values $p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

$$x_{s,r} = 0 \quad \text{if } s \not\models \exists \Diamond G$$

$$x_{s,r} = 1 \quad \text{if } s \in G$$

If $s \notin G$, $s \models \exists \Diamond G$ and $\alpha \in Act(s)$ then:

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r} \quad \text{if } rew(s,\alpha) = 0$$

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r-\ell} \quad \text{if } \ell = rew(s,\alpha) > 0$$

solution: $x_{s,r} = p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

linear program for the values $p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

$$x_{s,r} = 0 \quad \text{if } s \not\models \exists\Diamond G$$

$$x_{s,r} = 1 \quad \text{if } s \in G$$

minimize $\sum_s x_{s,r}$

If $s \notin G$, $s \models \exists\Diamond G$ and $\alpha \in Act(s)$ then:

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r} \quad \text{if } rew(s,\alpha) = 0$$

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r-\ell} \quad \text{if } \ell = rew(s,\alpha) > 0$$

unique solution: $x_{s,r} = p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

linear program for the values $p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

$$x_{s,r} = 0 \quad \text{if } s \not\models \exists\Diamond G$$

$$x_{s,r} = 1 \quad \text{if } s \in G$$

minimize $\sum_s x_{s,r}$

If $s \notin G$, $s \models \exists\Diamond G$ and $\alpha \in Act(s)$ then:

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r} \quad \text{if } rew(s,\alpha) = 0$$

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r-\ell} \quad \text{if } \ell = rew(s,\alpha) > 0$$

use the solutions $p_{t,i} = \Pr^{\max}_s(\Diamond^{\leqslant i} G)$ for $i < r$

computed in previous iterations

linear program for the values $p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

$$x_{s,r} = 0 \quad \text{if } s \not\models \exists\Diamond G$$

$$x_{s,r} = 1 \quad \text{if } s \in G$$

minimize $\sum_s x_{s,r}$

If $s \notin G$, $s \models \exists\Diamond G$ and $\alpha \in Act(s)$ then:

$$x_{s,r} \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,r} \qquad \text{if } rew(s,\alpha) = 0$$

$$x_{s,r} \geqslant const$$

use the solutions $p_{t,i} = \Pr^{\max}_s(\Diamond^{\leqslant i} G)$ for $i < r$

computed in previous iterations

linear program for the values $p_{s,r} = \Pr^{\max}_s(\Diamond^{\leqslant r} G)$

$$x_{s,r} = 0 \quad \text{if } s \not\models \exists\Diamond G$$

$$x_{s,r} = 1 \quad \text{if } s \in G$$

$$\text{minimize} \sum_s x_{s,r}$$

If $s \notin G$, $s \models \exists\Diamond G$ and $\alpha \in Act(s)$ then:

$$x_{s,r} \geqslant \sum_{t \in S} P(s, \alpha, t) \cdot x_{t,r} \quad \text{if } rew(s, \alpha) = 0$$

$$x_{s,r} \geqslant const$$

linear in the
size of the MDP

linear program to be solved in the r-th iteration

Expectation quantiles

Expectation quantiles: example

$\mathcal{M} = (S, Act, P, energy, utility, s_0)$ MDP with two reward functions

expectation quantile for utility threshold $u \in \mathbb{Q}$:

$$\min \{ e \in \mathbb{N} : \mathrm{ExpUtil}_{s_0}^{\max}[energy \leqslant e] > u \}$$

minimal energy budget e required to ensure that the expected utility is larger than u

(under some scheduler)

Expectation quantiles: example

$\mathcal{M} = (S, Act, P, energy, utility, s_0)$ MDP with two reward functions

expectation quantile for utility threshold $u \in \mathbb{Q}$:

$$\min \left\{ e \in \mathbb{N} : \mathrm{ExpUtil}^{\max}_{s_0}[energy \leqslant e] > u \right\}$$

minimal energy budget e required to ensure that the expected utility is larger than u

computation of expectation quantiles:

iterative linear programming approach
(with back propagation as for probabilistic quantiles)

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles
* mean-payoff
* expected accumulated weights

Mean-payoff

Mean-payoff

given: a weighted graph without trap states

mean-payoff functions $\overline{\mathrm{MP}}$, $\underline{\mathrm{MP}}$: *InfPaths* $\to \mathbb{R}$:

$$\overline{\mathrm{MP}}(s_0\, s_1\, s_2 \ldots) = \limsup_{n\to\infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} wgt(s_i)$$

$$\underline{\mathrm{MP}}(s_0\, s_1\, s_2 \ldots) = \liminf_{n\to\infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} wgt(s_i)$$

Mean-payoff

given: a weighted graph without trap states

mean-payoff functions $\overline{\mathrm{MP}}, \underline{\mathrm{MP}} : \mathit{InfPaths} \to \mathbb{R}$:

$$\overline{\mathrm{MP}}(s_0\, s_1\, s_2 \ldots) = \limsup_{n \to \infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} \mathit{wgt}(s_i)$$

$$\underline{\mathrm{MP}}(s_0\, s_1\, s_2 \ldots) = \liminf_{n \to \infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} \mathit{wgt}(s_i)$$

if $\mathit{wgt}(s) = +1$, $\mathit{wgt}(t) = -1$ then there exists $n_1, n_2, \ldots$ and $k_1, k_2, \ldots \in \mathbb{N}$ s.t. for $\pi = s^{n_1}\, t^{k_1}\, s^{n_2}\, t^{k_2} \ldots$:

$$\underline{\mathrm{MP}}(\pi) < 0 < \overline{\mathrm{MP}}(\pi)$$

Expected mean-payoff in finite MC or MDP

fundamental results:

in finite MC: $\mathbb{E}_s(\underline{\mathrm{MP}}) = \mathbb{E}_s(\overline{\mathrm{MP}})$

in finite MDP:

$$\mathbb{E}_s^{\max}(\underline{\mathrm{MP}}) = \mathbb{E}_s^{\max}(\overline{\mathrm{MP}})$$

$$\mathbb{E}_s^{\min}(\underline{\mathrm{MP}}) = \mathbb{E}_s^{\min}(\overline{\mathrm{MP}})$$

and optimal MD-scheduler exist

notation: $\mathbb{E}_s^*(\mathrm{MP})$ rather than $\mathbb{E}_s^*(\underline{\mathrm{MP}})$ resp. $\mathbb{E}_s^*(\overline{\mathrm{MP}})$

Expected mean-payoff in finite MC

fundamental results:

in finite MC: $\mathbb{E}_s(\underline{\mathrm{MP}}) = \mathbb{E}_s(\overline{\mathrm{MP}})$

for finite MC without traps:

> Almost all paths eventually enter a BSCC and visit all its states infinitely often.

BSCC: bottom strongly connected component

Expected mean-payoff in finite MC

fundamental results:

in finite MC: $\mathbb{E}_s(\underline{\mathrm{MP}}) = \mathbb{E}_s(\overline{\mathrm{MP}})$

for finite MC without traps:

> Almost all paths eventually enter a BSCC and visit all its states infinitely often ...
>
> ... with the same long-run frequencies ...

BSCC: bottom strongly connected component

Long-run frequencies in finite MC

steady-state probabilities in BSCC B of a finite MC:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t\big(\bigcirc^i s\big) \quad \text{for each } t \in B$$

for almost all paths $\pi = s_0\, s_1\, s_2 \ldots$ with $\pi \models \Diamond B$:

$$\theta^B(s) \;=\; \underbrace{\lim_{n\to\infty} \frac{1}{n+1} \cdot \mathit{freq}(s, s_0\, s_1 \ldots s_n)}_{\text{long-run frequency of state } s \text{ in path } \pi}$$

... limit exists for almost all paths ...

Long-run frequencies in finite MC

steady-state probabilities in BSCC B of a finite MC:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t(\bigcirc^i s) \quad \text{for each } t \in B$$

for almost all paths $\pi = s_0\, s_1\, s_2 \ldots$ with $\pi \models \lozenge B$:

$$\theta^B(s) \;=\; \underbrace{\lim_{n\to\infty} \frac{1}{n+1} \cdot freq(s, s_0\, s_1 \ldots s_n)}_{\text{long-run frequency of state } s \text{ in path } \pi}$$

$$freq(s, s_0\, s_1 \ldots s_n) \;=\; \left\{ \begin{array}{l} \text{number of occurrences of } s \\ \text{in the sequence } s_0\, s_1 \ldots s_n \end{array} \right.$$

Mean-payoff in finite weighted MC

steady-state probabilities in BSCC B of a finite MC:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t\big(\bigcirc^i s\big) \quad \text{for each } t \in B$$

for almost all paths $\pi = s_0\, s_1\, s_2 \ldots$ with $\pi \models \Diamond B$:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n+1} \cdot \mathit{freq}(s, s_0\, s_1 \ldots s_n)$$

if $\pi \models \Diamond B$ where B is a BSCC then almost surely

$$\mathrm{MP}(\pi) \;=\; \sum_{s\in B} \theta^B(s) \cdot \mathit{wgt}(s)$$

Mean-payoff in finite weighted MC

steady-state probabilities in BSCC B of a finite MC:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t\big(\bigcirc^i s\big) \quad \text{for each } t \in B$$

for almost all paths $\pi = s_0\, s_1\, s_2 \ldots$ with $\pi \models \Diamond B$:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n+1} \cdot \mathit{freq}\big(s, s_0\, s_1 \ldots s_n\big)$$

if $\pi \models \Diamond B$ where B is a BSCC then almost surely

$$\mathrm{MP}(\pi) \;=\; \underbrace{\sum_{s\in B} \theta^B(s) \cdot \mathit{wgt}(s)}_{\text{only depends on } B}$$

Mean-payoff in finite weighted MC

steady-state probabilities in BSCC B of a finite MC:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t(\bigcirc^i s) \quad \text{for each } t \in B$$

for almost all paths $\pi = s_0\, s_1\, s_2 \ldots$ with $\pi \models \Diamond B$:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n+1} \cdot \mathit{freq}(s, s_0\, s_1 \ldots s_n)$$

if $\pi \models \Diamond B$ where B is a BSCC then almost surely

$$\mathrm{MP}(\pi) \;=\; \underbrace{\sum_{s\in B} \theta^B(s) \cdot \mathit{wgt}(s)}_{\text{only depends on } B} \;\stackrel{\text{def}}{=}\; \mathrm{MP}(B)$$

Mean-payoff in finite weighted MC

steady-state probabilities in BSCC B of a finite MC:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t(\bigcirc^i s) \quad \text{for each } t \in B$$

for almost all paths $\pi = s_0\, s_1\, s_2 \ldots$ with $\pi \models \Diamond B$:

$$\theta^B(s) \;=\; \lim_{n\to\infty} \frac{1}{n+1} \cdot \mathit{freq}(s, s_0\, s_1 \ldots s_n)$$

if $\pi \models \Diamond B$ where B is a BSCC then almost surely

$$\mathrm{MP}(\pi) \;=\; \sum_{s\in B} \theta^B(s) \cdot \mathit{wgt}(s) \;\stackrel{\text{def}}{=}\; \mathrm{MP}(B)$$

expected mean-payoff: $\sum_{B} \Pr_{s_0}(\Diamond B) \cdot \mathrm{MP}(B)$

Mean-payoff in MDPs

random variable $\overline{\mathrm{MP}} : \mathit{InfPaths} \to \mathbb{R}$ defined by

$$\overline{\mathrm{MP}}(s_0\, s_1\, s_2 \ldots) = \limsup_{n \to \infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} \mathit{wgt}(s_i)$$

Mean-payoff in MDPs

Given MDP with weight function $\mathit{wgt} : S \to \mathbb{Q}$, find a scheduler maximizing the expected mean-payoff.

random variable $\overline{\mathrm{MP}} : \mathit{InfPaths} \to \mathbb{R}$ defined by

$$\overline{\mathrm{MP}}(s_0\, s_1\, s_2 \ldots) = \limsup_{n\to\infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} \mathit{wgt}(s_i)$$

Mean-payoff in MDPs

Given MDP with weight function $\mathit{wgt} : S \to \mathbb{Q}$, find a scheduler maximizing the expected mean-payoff.

Results: [Howard'60], [Dernan'66], [Kallenberg'80] ...

- optimal MD-scheduler exist
- computable in polynomial-time via linear program
  to encode the long-run frequencies of MR-scheduler
- value and policy iteration algorithms
- extensions for multiple mean-payoff constraints
  [Brazdil/Brozek/Chatterjee/Foreijt/Kucera'14]

Mean-payoff in MDPs

Given MDP with weight function $wgt : S \to \mathbb{Q}$, find a scheduler maximizing the expected mean-payoff.

Results: [Howard'60], [Dernan'66], [Kallenberg'80] ...

- optimal MD-scheduler exist
- computable in polynomial-time via linear program
  to encode the long-run frequencies of MR-scheduler
- value and policy iteration algorithms
- extensions for multiple mean-payoff constraints
  [Brazdil/Brozek/Chatterjee/Foreijt/Kucera'14]

Mean-payoff in strongly connected MDPs

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

... uses variables $x_{s,\alpha}$ for $s \in S$, $\alpha \in Act(s)$
to encode the long-run frequencies of the
state-action pairs (s, α) in MR-schedulers

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

... uses variables $x_{s,\alpha}$ for $s \in S$, $\alpha \in Act(s)$
to encode the long-run frequencies of the
state-action pairs (s, α) in MR-schedulers

Given the values $x_{s,\alpha}$, a corresponding MR-scheduler σ can be defined by:

- if $x_s \overset{\text{def}}{=} \sum_{\alpha \in Act(s)} x_{s,\alpha} > 0$ then: $\sigma(s)(\alpha) = x_{s,\alpha} / x_s$

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

... uses variables $x_{s,\alpha}$ for $s \in S$, $\alpha \in Act(s)$ to encode the long-run frequencies of the state-action pairs (s, α) in MR-schedulers

Given the values $x_{s,\alpha}$, a corresponding MR-scheduler σ can be defined by:

- if $x_s \stackrel{\text{def}}{=} \sum_{\alpha \in Act(s)} x_{s,\alpha} > 0$ then: $\sigma(s)(\alpha) = x_{s,\alpha}/x_s$
- if $x_s = 0$ then σ behaves an MD-scheduler that reaches a state t with $x_t = 1$ with probability 1

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

maximize $\sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha)$ subject to:

↑

variables for the long-run frequencies of state-action pairs

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

maximize $\sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha)$ subject to:

mean-payoff of
MR-scheduler σ given by

$$\sigma(s)(\alpha) = x_{s,\alpha}/x_s$$

for each state s with $x_s \stackrel{\text{def}}{=} \sum_{\alpha \in Act(s)} x_{s,\alpha} > 0$

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

maximize $\sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha)$ subject to:

$$x_t \;=\; \sum_{s,\alpha} x_{s,\alpha} \cdot P(s,\alpha,t) \quad \text{for } t \in S$$

balance equation
for state t

$$x_t \;=\; \sum_{\beta \in Act(t)} x_{t,\beta}$$

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

maximize $\sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha)$ subject to:

$$x_t = \sum_{s,\alpha} x_{s,\alpha} \cdot P(s,\alpha,t) \quad \text{for } t \in S$$

$$x_{s,\alpha} \geqslant 0 \quad \text{for } s \in S \text{ and } \alpha \in Act(s)$$

long-run frequencies are non-negative

$$x_t = \sum_{\beta \in Act(t)} x_{t,\beta}$$

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

$$\text{maximize} \sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha) \text{ subject to:}$$

$$x_t = \sum_{s,\alpha} x_{s,\alpha} \cdot P(s,\alpha,t) \quad \text{for } t \in S$$

$$x_{s,\alpha} \geqslant 0 \quad \text{for } s \in S \text{ and } \alpha \in Act(s)$$

$$\sum_{s,\alpha} x_{s,\alpha} = 1 \qquad x_t = \sum_{\beta \in Act(t)} x_{t,\beta}$$

long-run frequencies yield a distribution

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

$$\text{maximize} \sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha) \text{ subject to:}$$

$$x_t = \sum_{s,\alpha} x_{s,\alpha} \cdot P(s,\alpha,t) \quad \text{for } t \in S$$

$$x_{s,\alpha} \geqslant 0 \quad \text{for } s \in S \text{ and } \alpha \in Act(s)$$

$$\sum_{s,\alpha} x_{s,\alpha} = 1 \qquad\qquad x_t = \sum_{\beta \in Act(t)} x_{t,\beta}$$

Each solution induces an optimal MR-scheduler.

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

$$\text{maximize} \sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha) \text{ subject to:}$$

$$x_t = \sum_{s,\alpha} x_{s,\alpha} \cdot P(s,\alpha,t) \quad \text{for } t \in S$$

$$x_{s,\alpha} \geqslant 0 \quad \text{for } s \in S \text{ and } \alpha \in Act(s)$$

$$\sum_{s,\alpha} x_{s,\alpha} = 1 \qquad x_t = \sum_{\beta \in Act(t)} x_{t,\beta}$$

Each solution induces an optimal MR-scheduler.

But how to obtain an optimal MD-scheduler **?**

Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

$$\text{maximize} \sum_{s,\alpha} x_{s,\alpha} \cdot wgt(s,\alpha) \text{ subject to:}$$

$$x_t = \sum_{s,\alpha} x_{s,\alpha} \cdot P(s,\alpha,t) \quad \text{for } t \in S$$

$$x_{s,\alpha} \geqslant 0 \quad \text{for } s \in S \text{ and } \alpha \in Act(s)$$

$$\sum_{s,\alpha} x_{s,\alpha} = 1$$

$$x_s = \sum_{\alpha \in Act(s)} x_{s,\alpha}$$

optimal MD-scheduler: for each state s with $x_s > 0$ pick an action α with $x_{s,\alpha} > 0$

Mean-payoff in MDPs: general case

given: weighted MDP $\mathcal{M}$ without trap states

task: find a scheduler that maximizes the expected mean-payoff

State s is called a trap if $Act(s) = \varnothing$.

Mean-payoff in MDPs: general case

given: weighted MDP $\mathcal{M}$ without trap states

task: find a scheduler that maximizes the expected mean-payoff

method 1: [KALLENBERG'80]

use an LP with two variables for each state-action pair

$x_{s,\alpha}$ long-run frequency

$y_{s,\alpha}$ frequency in the transient part

State s is called a trap if $Act(s) = \varnothing$.

Mean-payoff in MDPs: general case

given: weighted MDP $\mathcal{M}$ without trap states

task: find a scheduler that maximizes the expected mean-payoff

method 1: [KALLENBERG'80]

use an LP with two variables for each state-action pair

$x_{s,\alpha}$ long-run frequency

$y_{s,\alpha}$ frequency in the transient part

method 2:

compute the maximal expected mean-payoff of the MECs and "compose" the result for the full MDP

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, \ldots, \mathcal{E}_k$

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$

$\mathcal{M}'$ arises from $\mathrm{MEC}(\mathcal{M})$ by adding

- a fresh trap state *goal*
- a new action symbol τ
- transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ for $i, ..., k$

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$ with weight mp_i for the transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ and weight 0 for all other states

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, \ldots, \mathcal{E}_k$

step 2: for $i = 1, \ldots, k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$ with weight mp_i for the transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ and weight 0 for all other states

step 4: compute the maximal expected total weight in $\mathcal{M}'$

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, \ldots, \mathcal{E}_k$

step 2: for $i = 1, \ldots, k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$ with weight mp_i for the transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ and weight 0 for all other states

step 4: compute the maximal expected total weight

$\Pr^{\min}_{\mathcal{M}'}(\Diamond goal) = 1$ maximal expected total weight and optimal MD-scheduler exist

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$ with weight mp_i for the transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ and weight 0 for all other states

step 4: compute the maximal expected total weight in $\mathcal{M}'$

$$\mathbb{E}^{\max}_{\mathcal{M}'}(\text{“total weight”}) = \mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$$

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$

step 3: construct the modified MEC-quotient $\mathcal{M}'$ with weight mp_i for the transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ and weight 0 for all other states

step 4: compute the maximal expected total weight in $\mathcal{M}'$

question: how to compute an optimal scheduler **?**

Mean-payoff in MDPs: general case

step 1: compute the maximal end components $\mathcal{E}_1, ..., \mathcal{E}_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff mp_i of $\mathcal{E}_i$
... and an optimal MD-scheduler σ_i

step 3: construct the modified MEC-quotient $\mathcal{M}'$ with weight mp_i for the transitions $\mathcal{E}_i \xrightarrow{\tau} goal$ and weight 0 for all other states

step 4: compute the maximal expected total weight in $\mathcal{M}'$ ... and an optimal MD-scheduler ν

optimal MD-scheduler arises by combining $\nu, \sigma_1, \ldots, \sigma_k$

Expected long-run ratios

$\textit{ratio} = \frac{\textit{cost}}{\textit{util}}$ where $\textit{cost}$, $\textit{util}$ are reward functions

Expected long-run ratios

for Markov chains:

trivially computable in each BSCC as the quotient of the mean-payoff of both reward functions

$$\sum_{B} \Pr_s(\Diamond B) \cdot \frac{\mathrm{MP}[\mathit{cost}](B)}{\mathrm{MP}[\mathit{util}](B)}$$

B ranges over all BSCCs of the MC

$\mathit{ratio} = \frac{\mathit{cost}}{\mathit{util}}$ where cost, util are reward functions

Expected long-run ratios

for Markov chains:

trivially computable in each BSCC as the quotient of the mean-payoff of both reward functions

for MDPs:

- optimal MD-schedulers exist [GIMBERT'07]
- LP-based approach [DE ALFARO'98]

$\textit{ratio} = \frac{\textit{cost}}{\textit{util}}$ where ***cost***, ***util*** are reward functions

Expected long-run ratios

for Markov chains:

trivially computable in each BSCC as the quotient of the mean-payoff of both reward functions

for MDPs:

- optimal MD-schedulers exist [GIMBERT'07]
- LP-based approach [DE ALFARO'98]

minimize y subject to

$$x_s \geqslant cost(s,\alpha) - y \cdot util(s,\alpha) + \sum_{t \in S} P(s,\alpha,t) \cdot x_t$$

for all states s and $\alpha \in Act(s)$

Expected long-run ratios

for Markov chains:

trivially computable in each BSCC as the quotient of the mean-payoff of both reward functions

for MDPs:

- optimal MD-schedulers exist [Gimbert'07]
- LP-based approach [de Alfaro'98]
- fractional LP for uni-chain MDPs [Essen/Jobstmann'11]

  using an encoding of MR-scheduler as for mean-payoff;
  synthesis of an MD-scheduler maximizing the long-run ratio

Tutorial: Probabilistic Model Checking

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles
* mean-payoff
* expected accumulated weights

Stochastic shortest/longest path problem

weighted
MDP $\mathcal{M}$

$\diamondplus$ *goal*
accumulated weight until
reaching a goal state

requirement for $\mathcal{M}$:
$\Pr^{\min}(\Diamond goal) = 1$

best- or worst-case expectation
$\mathbb{E}^{\min}(\diamondplus goal)$ or $\mathbb{E}^{\max}(\diamondplus goal)$

extrema over all schedulers

Stochastic shortest/longest path problem

weighted MDP $\mathcal{M}$

$\oplus$ *goal*
accumulated weight until reaching a goal state

relaxed requirement:
$\Pr^{\max}(\Diamond goal) = 1$

Bertsekas/Tsitsiklis'91
de Alfaro'99

best- or worst-case expectation
$\mathbb{E}^{\min}(\oplus goal)$ or $\mathbb{E}^{\max}(\oplus goal)$

extrema over all proper schedulers

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.

$$T = \{s \in S : \Pr_s^{\max}(\Diamond G) = 1\} \neq \varnothing$$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in T$

maximum over all proper schedulers

σ is proper iff $\Pr_s^{\sigma}(\Diamond G) = 1$ for all $s \in T$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$T = \{s \in S : \Pr^{\max}_s(\Diamond G) = 1\} \neq \varnothing$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in T$

W.l.o.g. $T = S$.

σ is proper iff $\Pr^{\sigma}_s(\Diamond G) = 1$ for all $s \in T$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.

$$T = \{s \in S : \Pr_s^{\max}(\Diamond G) = 1\} \neq \varnothing$$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in T$

W.l.o.g. $T = S$.

replace $\mathcal{M}$ with the sub-MDP consisting of

- the states in T and
- the state-action pairs (s, α) where $s \in T \setminus G$, $\alpha \in Act(s)$ and

$$\Pr_s^{\max}(\Diamond G) = \sum_{t \in S} P(s, \alpha, t) \cdot \Pr_t^{\max}(\Diamond G)$$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$T = \{ s \in S : \Pr^{\max}_s(\Diamond G) = 1 \} \neq \varnothing$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in T$

W.l.o.g. $T = S$. In particular, $s \models \exists \Diamond G$ for all $s \in S$.

replace $\mathcal{M}$ with the sub-MDP consisting of

- the states in T and
- the state-action pairs (s, α) where $s \in T \setminus G$, $\alpha \in Act(s)$ and

$$\Pr^{\max}_s(\Diamond G) = \sum_{t \in S} P(s, \alpha, t) \cdot \Pr^{\max}_t(\Diamond G)$$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.

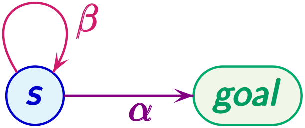
$$T = \{s \in S : \Pr_s^{\max}(\Diamond G) = 1\} \neq \varnothing$$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in T$

W.l.o.g. $T = S$. In particular, $s \models \exists \Diamond G$ for all $s \in S$.

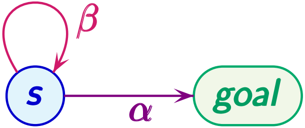
$\mathbb{E}_s^{\max}(\oplus goal)$ can be infinite !

Maximal expected accumulated weight



$$wgt(s, \alpha) = 0$$
$$wgt(s, \beta) = 1$$

Maximal expected accumulated weight



$$wgt(s, \alpha) = 0$$
$$wgt(s, \beta) = 1$$

maximal expected acumulated weight:

$$\mathbb{E}_s^{\max}(\diamondplus\, goal) = +\infty$$

note that $\mathbb{E}_s^{\sigma_n}(\diamondplus\, goal) = n$ where σ_n schedules

- β for the first n visits of s
- α for the $(n{+}1)$-st visit of s

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

$x_s < +\infty$ for all $s \in S$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\Diamond\!\!\!\!+ G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- the vector $(x_s)_{s \in S}$ is computable via the Bellman equations

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($ "total weight" $) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s = \max_{\alpha \in Act(s)} \Big(wgt(s,\alpha) + \sum_{t \in S} P(s,\alpha,t) \cdot x_t \Big)$$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s = \max_{\alpha \in Act(s)} \Big(wgt(s,\alpha) + \sum_{t \in S} P(s,\alpha,t) \cdot x_t \Big)$$

... unique fixpoint

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr_s^{\max}(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in S$

If $\mathbb{E}_s^{\sigma}(\text{“total weight”}) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s = \max_{\alpha \in Act(s)} \Big(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \Big)$$

... unique fixpoint, optimal MD-scheduler exist ...

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s(\text{"total weight"}) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

If $s \in G$ then $x_s = 0$. Otherwise:

$$x_s \geqslant \max_{\alpha \in Act(s)} \Big(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \Big)$$

unique solution where $\sum_{s \in S} x_s$ is minimal

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr_s^{\max}(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in S$

If $\mathbb{E}_s^{\sigma}($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

If $s \in G$ then $x_s^{(n)} = 0$. Otherwise:

$$x_s^{(n)} = \max_{\alpha \in Act(s)} \left(wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} \right)$$

value iteration (arbitrary starting vector)

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [BERTSEKAS/TSITSIKLIS'91]

- $x_s < +\infty$ for all $s \in S$
- the vector $(x_s)_{s \in S}$ is computable via the Bellman equations

How to compute x_s if $\mathbb{E}^{\sigma}_s($"total weight"$) > -\infty$ for some improper scheduler σ ?

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr_s^{\max}(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in S$

If $\mathbb{E}_s^{\sigma}(\text{“total weight”}) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- the vector $(x_s)_{s \in S}$ is computable via the Bellman equations

How to compute x_s if $\mathbb{E}_s^{\sigma}(\text{“total weight”}) > -\infty$ for some improper scheduler σ ? How to check finiteness ?

Non-negative weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

consider the case of non-negative weights,
i.e., $wgt(s, \alpha) \geqslant 0$ for all state-action pairs

Non-negative weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G) = \infty$ iff s can reach a positive EC

end component that contains a state-action pair with positive weight

Non-negative weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr_s^{\max}(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}_s^{\max}(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}_s^{\max}(\oplus G) = \infty$ iff s can reach a positive EC
- if $\mathcal{M}$ has no positive ECs and $\mathcal{N} = \mathrm{MEC}(\mathcal{M})$ then:

$$\mathbb{E}_{\mathcal{M},s}^{\max}(\oplus G) \;=\; \mathbb{E}_{\mathcal{N},s}^{\max}(\oplus G)$$

The MEC-quotient has no end components and maximal expected accumulated weights are computable using the Bellman equations.

Non-negative weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G) = \infty$ iff s can reach a positive EC
- if $\mathcal{M}$ has no positive ECs and $\mathcal{N} = \mathrm{MEC}(\mathcal{M})$ then:

$$\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$$

Hence: $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G)$ is computable in polynomial time.

Non-positive weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G)$ is finite ... and non-positive

Non-positive weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G)$ is finite ... and non-positive
- if $\mathcal{N}$ is the MDP arising from $\mathcal{M}$ by collapsing all zero-ECs then ...

↑

end component $\mathcal{E}$ with $wgt(s, \alpha) = 0$
for all state-action pairs (s, α) in $\mathcal{E}$

Non-positive weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G)$ is finite ... and non-positive
- if $\mathcal{N}$ is the MDP arising from $\mathcal{M}$ by collapsing all zero-ECs then $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$

↑

end component $\mathcal{E}$ with $wgt(s, \alpha) = 0$
for all state-action pairs (s, α) in $\mathcal{E}$

Non-positive weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G)$ is finite ... and non-positive
- if $\mathcal{N}$ is the MDP arising from $\mathcal{M}$ by collapsing all zero-ECs then $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$

↑

computable as the MECs of the MDP $\mathcal{M}_0$ consisting of the state-action pairs in $\mathcal{M}$ with weight 0

Non-positive weights

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

results: [De Alfaro'99]

- $\mathbb{E}^{\max}_s(\oplus G)$ is finite ... and non-positive
- if $\mathcal{N}$ is the MDP arising from $\mathcal{M}$ by collapsing all zero-ECs then $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$
- $\mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$ computable via Bellman equations

  ... expected total weight of each improper scheduler is $-\infty$

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t.
$\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- $(x_s)_{s \in S}$ is computable via the Bellman equations

Treatment of non-negative or non-positive weights: $\surd$

General case: **???**

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$ s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- $(x_s)_{s \in S}$ is computable via the Bellman equations

Treatment of non-negative or non-positive weights: $\surd$

General case: ... consider the MECs separately ...

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\mathsf{max}}_{\mathcal{E}}(\diamondsuit G) = \infty$$

iff $\quad \ldots$

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\diamondsuit G) = \infty$$

iff $\mathcal{E}$ is weight-divergent

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr^{\max}_{\mathcal{E},s}(\Diamond(wgt \geqslant r)) = 1 \right\} = \infty$$

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\diamondplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr^{\max}_{\mathcal{E},s}(\lozenge(wgt \geqslant r)) = 1 \} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\{ \pi : \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \} = 1$

$pref(\pi, n)$ = prefix of π of length n

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr^{\max}_{\mathcal{E},s}(\Diamond(wgt \geqslant r)) = 1 \} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\{ \pi : \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or ...

$pref(\pi, n)$ = prefix of π of length n

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} \,:\, \Pr^{\max}_{\mathcal{E},s}(\Diamond(wgt \geqslant r)) = 1 \right\} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\left\{ \pi \,:\, \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \right\} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$ & $\mathcal{E}$ is gambling

$pref(\pi, n)$ = prefix of π of length n

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\diamondsuit G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr^{\max}_{\mathcal{E},s}(\Diamond(wgt \geqslant r)) = 1 \} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\{ \pi : \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$ & $\mathcal{E}$ is gambling

there exists scheduler s.t. almost surely:

$$\limsup_{n\to\infty} wgt(pref(\pi, n)) = +\infty$$
$$\liminf_{n\to\infty} wgt(pref(\pi, n)) = -\infty$$

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr^{\max}_s(\Diamond(wgt \geqslant r)) = 1 \} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\{ \pi : \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$ & $\mathcal{E}$ is gambling

can be checked in polynomial time

exists scheduler s.t. almost surely:

$$\limsup_{n\to\infty} wgt(pref(\pi, n)) = +\infty$$

$$\liminf_{n\to\infty} wgt(pref(\pi, n)) = -\infty$$

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr^{\max}_{s}(\Diamond(wgt \geqslant r)) = 1 \} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\{ \pi : \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$ & $\mathcal{E}$ is gambling

can be checked in polynomial time

how to check whether an EC is gambling ?

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr^{\max}_{s}(\Diamond(wgt \geqslant r)) = 1 \right\} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\left\{ \pi : \limsup_{n\to\infty} wgt(pref(\pi, n)) = \infty \right\} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$ & $\mathcal{E}$ is gambling

The problem to check whether a given EC is gambling is NP-hard

Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^{\max}_{\mathcal{E}}(\oplus G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states s in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr^{\max}_{s}(\Diamond(wgt \geqslant r)) = 1 \right\} = \infty$$

iff $\Pr^{\max}_{\mathcal{E}}\left\{ \pi : \limsup_{n \to \infty} wgt(pref(\pi, n)) = \infty \right\} = 1$

iff $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) > 0$ or $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$ & $\mathcal{E}$ is gambling

The problem to check whether a given EC is gambling

- is NP-hard
- solvable in polynomial-time if $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ.

$\mathcal{E}'$ is a finite strongly connected Markov chain

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

$\mathcal{E}$ is a finite strongly connected Markov chain

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC

$\mathcal{E}$ is a finite strongly connected Markov chain

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

> If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC, i.e., the total weight of all cycles in $\mathcal{E}$ is 0.

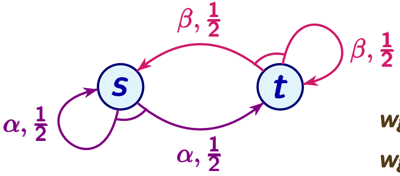
$\mathcal{E}$ is a finite strongly connected Markov chain

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC, i.e., the total weight of all cycles in $\mathcal{E}$ is 0.



$wgt(s,\alpha) = +1$
$wgt(t,\beta) = -1$

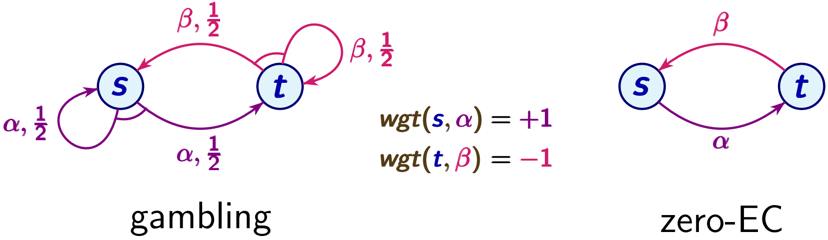
gambling

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

> If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC, i.e., the total weight of all cycles in $\mathcal{E}$ is 0.



Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

> If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC, i.e., the total weight of all cycles in $\mathcal{E}$ is 0.

Let $\mathcal{E}$ be a zero-EC and s, t states in $\mathcal{E}$. There exists $w(s,t) \in \mathbb{Z}$ such that:

$$w(s,t) = wgt(\pi) \quad \text{for all paths } \pi \text{ from } s \text{ to } t$$

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = 0$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

> If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC, i.e., the total weight of all cycles in $\mathcal{E}$ is 0.

Let $\mathcal{E}$ be a zero-EC and s, t states in $\mathcal{E}$. There exists $w(s,t) \in \mathbb{Z}$ such that:

$$w(s,t) = wgt(\pi) \quad \text{for all paths } \pi \text{ from } s \text{ to } t$$

Then: $w(t,s) = -w(s,t)$

Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{E}}(\mathrm{MP}) = \mathbf{0}$.

Pick an MD-scheduler σ s.t. $\mathbb{E}^{\sigma}_{\mathcal{E},s}(\mathrm{MP}) = \mathbf{0}$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of σ. W.l.o.g. $\mathcal{E} = \mathcal{E}'$.

> If $\mathcal{E}$ is not gambling then $\mathcal{E}$ is a zero-EC, i.e., the total weight of all cycles in $\mathcal{E}$ is 0.

Let $\mathcal{E}$ be a zero-EC and s, t states in $\mathcal{E}$. There exists $w(s,t) \in \mathbb{Z}$ such that:

$$w(s,t) = wgt(\pi) \quad \text{for all paths } \pi \text{ from } s \text{ to } t$$

Then: $w(t,s) = -w(s,t)$

> ... remove $\mathcal{E}$ from $\mathcal{M}$ ...

Spider construction ... for removing zero-ECs

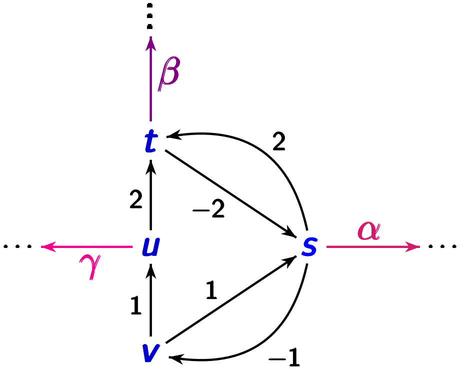
given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

task: construct an MDP $\mathcal{N}$ with the same non-zero ECs and where $\mathcal{E}$ is no longer a zero-EC

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

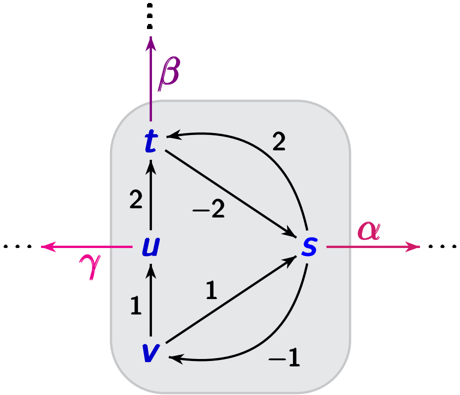
task: construct an MDP $\mathcal{N}$ with the same non-zero ECs and where $\mathcal{E}$ is no longer a zero-EC



Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

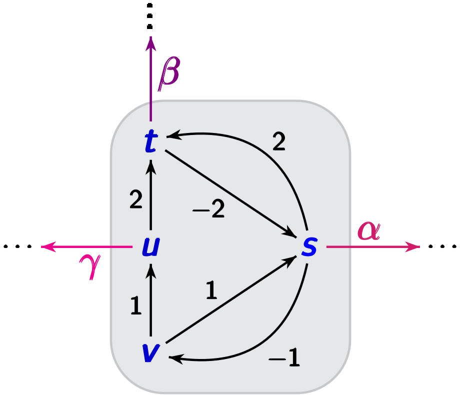
task: construct an MDP $\mathcal{N}$ with the same non-zero ECs and where $\mathcal{E}$ is no longer a zero-EC

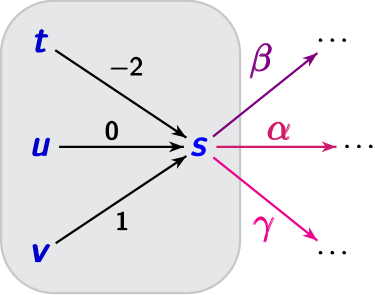


Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

task: construct an MDP $\mathcal{N}$ with the same non-zero ECs and where $\mathcal{E}$ is no longer a zero-EC

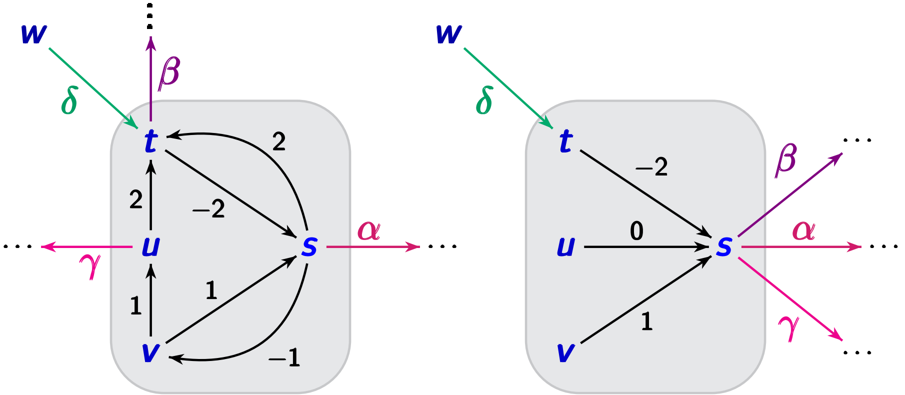




Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

task: construct an MDP $\mathcal{N}$ with the same non-zero ECs and where $\mathcal{E}$ is no longer a zero-EC



Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

task: construct an MDP $\mathcal{N}$ with the same non-zero ECs and where $\mathcal{E}$ is no longer a zero-EC

W.l.o.g: $Act(s) \cap Act(t) = \varnothing$ if $s \neq t$.

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

1. pick a state s in $\mathcal{E}$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

1. pick a state s in $\mathcal{E}$
2. remove all state-action pairs in $\mathcal{E}$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

1. pick a state s in $\mathcal{E}$
2. remove all state-action pairs in $\mathcal{E}$
3. for each state t in $\mathcal{E}$ with $t \neq s$:
   add transition $t \xrightarrow{\tau} s$ with $wgt(t,\tau) = \underbrace{-w(s,t)}_{w(t,s)}$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

1. pick a state s in $\mathcal{E}$
2. remove all state-action pairs in $\mathcal{E}$
3. for each state t in $\mathcal{E}$ with $t \neq s$:
   add transition $t \xrightarrow{\tau} s$ with $wgt(t, \tau) = -w(s, t)$
4. replace each state-action pair (t, β) in $\mathcal{M} \setminus \mathcal{E}$
   where $t \neq s$ with the pair (s, β)

$$s \xrightarrow{\text{in } \mathcal{E}} t \xrightarrow{\beta} \ldots$$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

1. pick a state s in $\mathcal{E}$
2. remove all state-action pairs in $\mathcal{E}$
3. for each state t in $\mathcal{E}$ with $t \neq s$:
   add transition $t \xrightarrow{\tau} s$ with $wgt(t,\tau) = -w(s,t)$
4. replace each state-action pair (t,β) in $\mathcal{M} \setminus \mathcal{E}$
   where $t \neq s$ with the pair (s,β):

$$wgt(s,\beta) = w(s,t) + wgt(t,\beta)$$

$$s \overset{\text{in } \mathcal{E}}{- - - - \rightarrow} t \xrightarrow{\beta} \ldots$$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

1. pick a state s in $\mathcal{E}$
2. remove all state-action pairs in $\mathcal{E}$
3. for each state t in $\mathcal{E}$ with $t \neq s$:
   add transition $t \xrightarrow{\tau} s$ with $wgt(t, \tau) = -w(s, t)$
4. replace each state-action pair (t, β) in $\mathcal{M} \setminus \mathcal{E}$
   where $t \neq s$ with the pair (s, β):

$$wgt(s, \beta) \; = \; w(s, t) + wgt(t, \beta)$$

$$P(s, \beta, u) \; = \; P(t, \beta, u) \quad \text{for all states } u \text{ in } \mathcal{M}$$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

spider construction yields a new MDP $\mathcal{N} = \mathcal{M}_{\setminus \mathcal{E}}$

$\mathcal{M}$ is weight-divergent iff $\mathcal{N}$ is weight-divergent

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

spider construction yields a new MDP $\mathcal{N} = \mathcal{M}_{\setminus\mathcal{E}}$

- $\mathcal{M}$ is weight-divergent iff $\mathcal{N}$ is weight-divergent
- $\mathbb{E}^{\max}_{\mathcal{M},s}(\diamondplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\diamondplus G)$ for all states s in $\mathcal{M}$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

spider construction yields a new MDP $\mathcal{N} = \mathcal{M}_{\backslash \mathcal{E}}$

- $\mathcal{M}$ is weight-divergent iff $\mathcal{N}$ is weight-divergent
- $\mathbb{E}^{\max}_{\mathcal{M},s}(\diamondplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\diamondplus G)$ for all states s in $\mathcal{M}$
- $\|\mathcal{N}\| \leqslant \|\mathcal{M}\| - 1$

where $\|\mathcal{M}\|$ = number of state-action pairs in $\mathcal{M}$

Spider construction ... for removing zero-ECs

given: MDP $\mathcal{M}$ and a zero-EC $\mathcal{E}$ of $\mathcal{M}$

spider construction yields a new MDP $\mathcal{N} = \mathcal{M}_{\setminus \mathcal{E}}$

- $\mathcal{M}$ is weight-divergent iff $\mathcal{N}$ is weight-divergent
- $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$ for all states s in $\mathcal{M}$
- $\|\mathcal{N}\| \leqslant \|\mathcal{M}\| - 1$

where $\|\mathcal{M}\|$ = number of state-action pairs in $\mathcal{M}$

idea: apply the spider construction recursively to check weight-divergence of strongly connected MDPs

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"

↑

$\mathcal{M}$ is not weight-divergent

as the total weight of almost all paths tends to $-\infty$

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"
3. pick a BSCC $\mathcal{E}$ of the MC induced by σ

strongly connected MC with
expected mean-payoff 0

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"
3. pick a BSCC $\mathcal{E}$ of the MC induced by σ
4. if $\mathcal{E}$ is a zero-EC then apply the procedure recursively to the MDP $\mathcal{M}_{\setminus\mathcal{E}}$.

↑ spider construction

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"
3. pick a BSCC $\mathcal{E}$ of the MC induced by σ
4. if $\mathcal{E}$ is a zero-EC then apply the procedure recursively to the MDP $\mathcal{M}_{\backslash\mathcal{E}}$.

   Otherwise ... $\mathcal{E}$ is gambling

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant \mathbf{0}$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < \mathbf{0}$ then return "no"
3. pick a BSCC $\mathcal{E}$ of the MC induced by σ
4. if $\mathcal{E}$ is a zero-EC then apply the procedure recursively to the MDP $\mathcal{M}_{\setminus \mathcal{E}}$.

   Otherwise return "yes, $\mathcal{M}$ is weight-divergent".

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"

If $\mathcal{M}$ is not weight-divergent then the algorithm has generated an MDP $\mathcal{N}$ with $\mathbb{E}^{\max}_{\mathcal{M},s}(\diamondplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\diamondplus G)$

recursively to the MDP $\mathcal{M}_{\setminus \mathcal{E}}$.

Otherwise return "yes, $\mathcal{M}$ is weight-divergent".

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"

If $\mathcal{M}$ is not weight-divergent then the algorithm has generated an MDP $\mathcal{N}$ with $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$ and $\mathbb{E}^{\sigma}_{\mathcal{N},s}(\text{"total weight"}) = -\infty$ for each improper scheduler σ. ... as $\mathcal{N}$ has no zero-ECs ...

Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) \leqslant 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP})$ and an optimal MD-scheduler σ
2. if $\mathbb{E}^{\max}_{\mathcal{M}}(\mathrm{MP}) < 0$ then return "no"

If $\mathcal{M}$ is not weight-divergent then the algorithm has generated an MDP $\mathcal{N}$ with $\mathbb{E}^{\max}_{\mathcal{M},s}(\oplus G) = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$ and $\mathbb{E}^{\sigma}_{\mathcal{N},s}(\text{"total weight"}) = -\infty$ for each improper scheduler σ.

... $\mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$ computable via Bellman equations ...

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$
s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- $(x_s)_{s \in S}$ is computable via the Bellman equations

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$
s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- $(x_s)_{s \in S}$ is computable via the Bellman equations

Recursive application of the spider construction ...

- to check that there is no weight-divergent MEC

[Baier/Bertrand/Dubslaff/Gburek/Sankur'17]

Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, Act, P, wgt)$ and $G \subseteq S$
s.t. $\Pr^{\max}_s(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = \mathbb{E}^{\max}_s(\oplus G)$ for $s \in S$

If $\mathbb{E}^{\sigma}_s($"total weight"$) = -\infty$ for each improper scheduler σ then: [Bertsekas/Tsitsiklis'91]

- $x_s < +\infty$ for all $s \in S$
- $(x_s)_{s \in S}$ is computable via the Bellman equations

Recursive application of the spider construction ...

- to check that there is no weight-divergent MEC
- to generate a new MDP $\mathcal{N}$ where $x_s = \mathbb{E}^{\max}_{\mathcal{N},s}(\oplus G)$ and the above criterion applies

Tutorial: Probabilistic Model Checking

Discrete-time Markov chains (DTMC)

* basic definitions
* probabilistic computation tree logic PCTL/PCTL*
* rewards, cost-utility ratios, weights
* conditional probabilities

Markov decision processes (MDP)

* basic definitions
* PCTL/PCTL* model checking
* fairness
* conditional probabilities
* rewards, quantiles, mean-payoff
* expected accumulated weights
* conditional expected accumulated rewards

Stochastic longest path problem

weighted MDP $\mathcal{M}$

$\diamondplus$ *goal*

accumulated weight until reaching a goal state

relaxed requirement:

$\Pr^{\max}(\Diamond \textit{goal}) = 1$

Bertsekas/Tsitsiklis'91
de Alfaro'99

maximal expectation

$\mathbb{E}^{\max}(\diamondplus \textit{goal})$

maximum over all proper schedulers

Maximal conditional expectations

weighted MDP $\mathcal{M}$

$\Diamond\!\!\!\!+\ \mathit{goal}$
accumulated weight until reaching a goal state

relaxed requirement:
$\Pr^{\max}(\Diamond \mathit{goal}) > 0$

BAIER/KLEIN/KLÜPPELHOLZ/WUNDERLICH'17

maximal conditional expectation
$\mathbb{E}^{\max}(\ \Diamond\!\!\!\!+\ \mathit{green} \mid \Diamond \mathit{goal}\ )$

maximum over all positive schedulers

Maximal conditional expectations

weighted
MDP $\mathcal{M}$

$\diamondplus\, \mathit{goal}$
accumulated weight until
reaching a goal state

relaxed requirement:
$\Pr^{\max}(\Diamond \mathit{goal}) > 0$

assumption:
non-negative
weights

maximal conditional expectation
$\mathbb{E}^{\max}(\, \diamondplus\, \mathit{green} \mid \Diamond \mathit{goal} \,)$

maximum over all positive schedulers

Why should we be interested in ..?

Why should we be interested in ..?

- termination time of probabilistic programs

  conditional expected number of steps until termination, under the condition that the program terminates

- failure diagnosis and resilience analysis

  e.g. cost of repair protocols for a certain failure scenario

- various forms of multi-objective reasoning

  e.g., expected utility level, assuming a fixed energy budget

- conditional value-at-risk

  expected loss in worst case scenarios, under the assumption that these scenarios indeed occur

Why is it more difficult ...?

Why is it more difficult ...?

unconditional expected accumulated rewards

- optimal memoryless schedulers exists that maximize the expected reward from every state
- computable via linear programs with one variable per state

Why is it more difficult ...?

unconditional expected accumulated rewards

- optimal memoryless schedulers exists that maximize the expected reward from every state
- computable via linear programs with one variable per state

conditional expected accumulated rewards

- optimal schedulers require memory
- local reasoning not sufficient

Why is it more difficult ...?

unconditional expected accumulated rewards

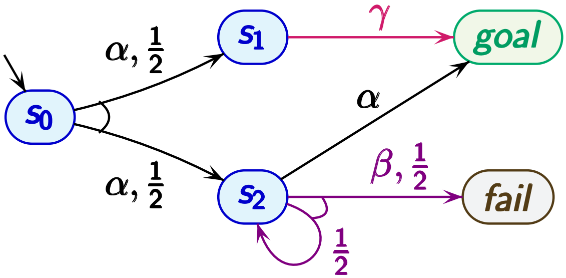
- optimal memoryless schedulers exists that maximize the expected reward from every state
- computable via linear programs with one variable per state

conditional expected accumulated rewards

- optimal schedulers require memory
- local reasoning not sufficient

... let's have a look at an example ...

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

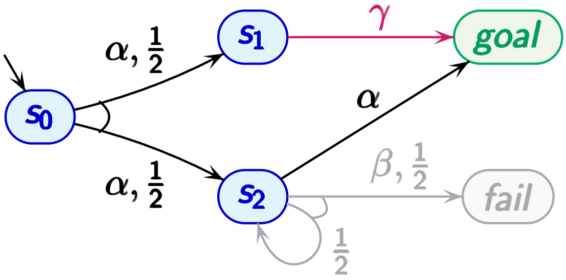
$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

maximal conditional expected reward:

$$\mathbb{E}^{\max}(\, \diamondsuit\!\!\!+ \mathit{goal} \mid \Diamond \mathit{goal}\,) = ???$$

Maximal conditional expected reward



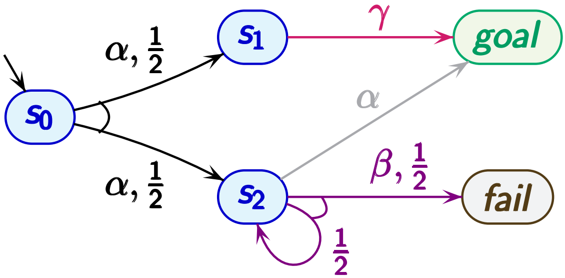
$rew(s_1, \gamma) = r$

$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

"choose always α in state s_2": $\dfrac{\frac{1}{2} \cdot r + \frac{1}{2} \cdot 0}{\frac{1}{2} + \frac{1}{2}} = \dfrac{r}{2}$

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

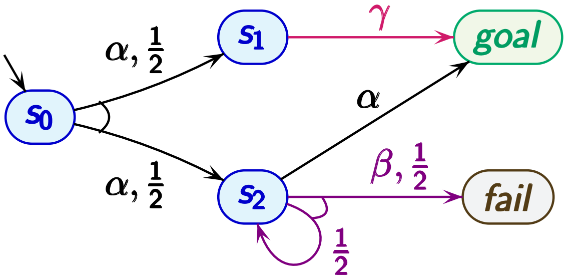
$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

"choose always α in state s_2": $\dfrac{\frac{1}{2} \cdot r + \frac{1}{2} \cdot 0}{\frac{1}{2} + \frac{1}{2}} = \dfrac{r}{2}$

"choose always β in state s_2": $\dfrac{\frac{1}{2} \cdot r + 0}{\frac{1}{2} + 0} = r$

Maximal conditional expected reward

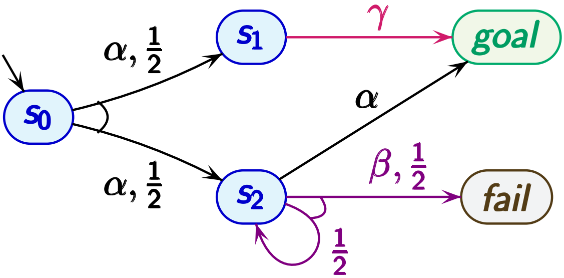


$$rew(s_1, \gamma) = r$$
$$rew(s_2, \beta) = 1$$
$$rew(s_i, \alpha) = 0$$

"choose β exacly for the first n visits of s_2"

$$\frac{\frac{1}{2} \cdot r + \frac{1}{2} \cdot \frac{1}{2^n} \cdot n}{\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2^n}}$$

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

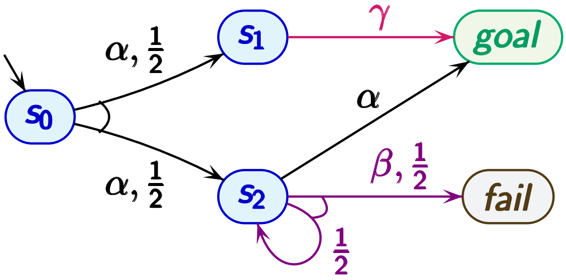
$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

"choose β exacly for the first n visits of s_2"

$$\frac{\frac{1}{2} \cdot r + \frac{1}{2} \cdot \frac{1}{2^n} \cdot n}{\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2^n}} = r + \frac{n - r}{2^n + 1}$$

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

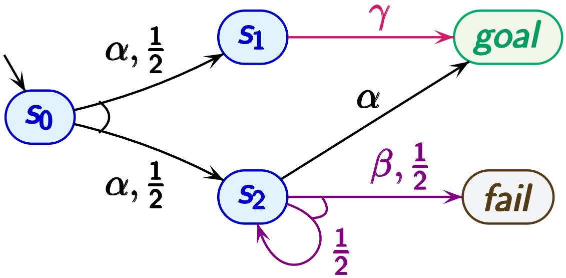
$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

"choose β exacly for the first n visits of s_2"

$$\frac{\frac{1}{2}\cdot r + \frac{1}{2}\cdot\frac{1}{2^n}\cdot n}{\frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2^n}} = r + \frac{n-r}{2^n+1} > r \quad \text{iff} \quad n > r$$

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

$rew(s_2, \beta) = 1$

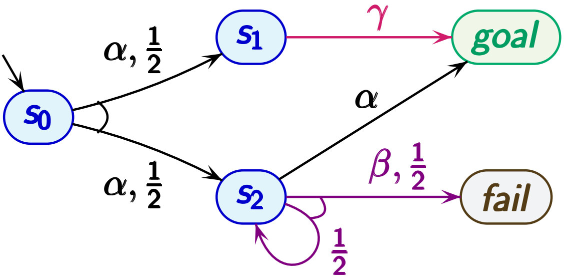
$rew(s_i, \alpha) = 0$

"choose β exacly for the first n visits of s_2"

$$\frac{\frac{1}{2}\cdot r + \frac{1}{2}\cdot\frac{1}{2^n}\cdot n}{\frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2^n}} = r + \frac{n-r}{2^n+1} > r \quad \text{iff} \quad n > r$$

optimal value is achieved for $n = r+2$

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

maximal conditional reward until *goal*:

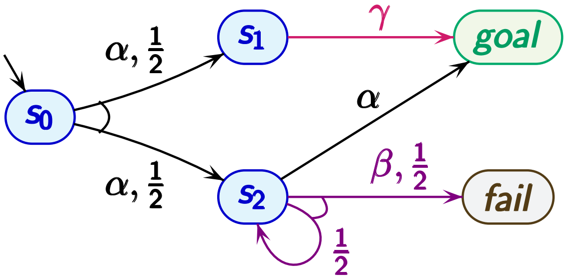
* memory required for optimal schedulers

  optimal scheduler needs counter for the number of visits in s_2

* local reasoning not sufficient

  ... as optimal decisions in s_2 depend on r

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

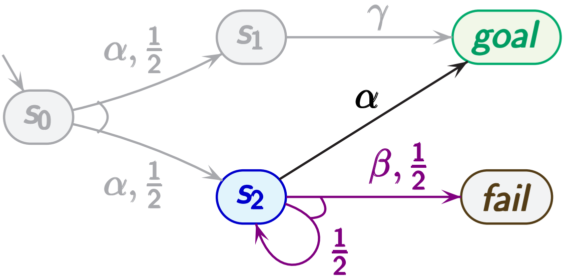
$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

maximal conditional reward until *goal*

... is finite for state s_0, namely $r + \frac{2}{2^{r+2}+1}$

Maximal conditional expected reward



$rew(s_1, \gamma) = r$

$rew(s_2, \beta) = 1$

$rew(s_i, \alpha) = 0$

maximal conditional reward until *goal*

... is finite for state s_0, namely $r + \frac{2}{2^{r+2}+1}$

... but infinite for s_2

$$\sup_{n \in \mathbb{N}} \frac{\frac{n}{2^n}}{\frac{1}{2^n}} = \infty$$

Problem statement

given: MDP $\mathcal{M} = (S, Act, P, rew, s_0)$ and F, $G \subseteq S$
such that $\Pr^{\max}_{s_0}(\, \Diamond F \mid \Diamond G \,) = 1$

task: ...

$\Pr^{\max}_{s_0}(\Diamond F \mid \Diamond G) = 1$ iff there is scheduler σ s.t.

1. $\Pr^{\sigma}_{s_0}(\Diamond G) > 0$ and
2. $\Pr^{\sigma}_{s_0}(\, \Diamond F \mid \Diamond G \,) = 1$

Problem statement

given: MDP $\mathcal{M} = (S, Act, P, rew, s_0)$ and F, $G \subseteq S$
such that $\Pr^{\max}_{s_0}(\Diamond F \mid \Diamond G) = 1$

task: compute $\mathbb{E}^{\max}_{s_0}(\text{⧫} F \mid \Diamond G)$

↑

maximal conditional accumulated reward to reach F

under all schedulers σ where $\Pr^{\sigma}_{s_0}(\Diamond G) > 0$
and $\Pr^{\sigma}_{s_0}(\Diamond F \mid \Diamond G) = 1$

Problem statement

given: MDP $\mathcal{M} = (S, Act, P, rew, s_0)$ and F, $G \subseteq S$
such that $\Pr^{\max}_{s_0}(\, \Diamond F \mid \Diamond G \,) = 1$

task: compute $\mathbb{E}^{\max}_{s_0}(\, \oplus F \mid \Diamond G \,)$

---

after some preprocessing and cleaning-up:

1. all states are reachable from s_0
2. $F = G = \{goal\}$ for a trap state $goal$
3. there is another trap state $fail$ with
$\Pr^{\min}_{s}(\, \Diamond(goal \vee fail)\,) \;=\; 1$ for all states s

Shortform notation used in the sequel

Given a scheduler σ with $\Pr^{\sigma}_{s_0}(\Diamond goal) > 0$, let:

$$\mathbb{CE}^{\sigma} = \mathbb{E}^{\sigma}_{s_0}(\oplus goal \mid \Diamond goal)$$

Maximal conditional expectation:

$$\mathbb{CE}^{\max} = \sup_{\sigma} \mathbb{CE}^{\sigma}$$

↑

ranging over all schedulers σ with $\Pr^{\sigma}_{s_0}(\Diamond goal) > 0$

Shortform notation used in the sequel

Given a scheduler σ with $\mathrm{Pr}^{\sigma}_{s_0}(\Diamond goal) > 0$, let:

$$\mathbb{CE}^{\sigma} = \mathbb{E}^{\sigma}_{s_0}(\,\text{⨁}goal \mid \Diamond goal\,)$$

Maximal conditional expectation:

$$\mathbb{CE}^{\max} = \sup_{\sigma} \mathbb{CE}^{\sigma}$$

↑

supremum over all
deterministic reward-based schedulers
$\sigma : S \times \mathbb{N} \to Act$

Checking finiteness

Given a scheduler σ with $\Pr^{\sigma}_{s_0}(\Diamond goal) > 0$, let:

$$\mathbb{CE}^{\sigma} = \mathbb{E}^{\sigma}_{s_0}(⨁ goal \mid \Diamond goal)$$

Maximal conditional expectation:

$$\mathbb{CE}^{\max} = \sup_{\sigma} \mathbb{CE}^{\sigma}$$

Checking finiteness in polynomial time:

$\mathbb{CE}^{\max} < \infty$ iff $\left\{\right.$ there is no scheduler σ s.t. $\Pr^{\sigma}_{s_0}(\Diamond goal) = 0$ and there is a reachable positive σ-cycle

If $\mathbb{CE}^{\max} < \infty$ then ...

If $\mathbb{CE}^{\max} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\max}$

pseudo-polynomial: time complexity is polynomial in the

* size of the graph structure and
* length of an unary encoding of the probability/reward values

If $\mathbb{CE}^{\mathsf{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\mathsf{max}}$
- threshold problem "is $\mathbb{CE}^{\mathsf{max}} \trianglerighteq \vartheta$?" is PSPACE-hard, and PSPACE-complete for acyclic MDPs

  ... same for upper bounds by duality ...

threshold problem:

given: MDP $\mathcal{M}$, $\vartheta \in \mathbb{Q}$ and $\trianglerighteq \in \{>, \geqslant\}$

task: check whether $\mathbb{CE}^{\mathsf{max}} \trianglerighteq \vartheta$

If $\mathbb{CE}^{\max} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\max}$
- threshold problem "is $\mathbb{CE}^{\max} \trianglerighteq \vartheta$?" is PSPACE-hard, and PSPACE-complete for acyclic MDPs
- there exists a saturation point $\wp$ such that optimal schedulers behave memoryless from reward $\wp$ on

  ... and maximize the probability to reach the goal state

If $\mathbb{CE}^{\mathsf{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\mathsf{max}}$
- threshold problem "is $\mathbb{CE}^{\mathsf{max}} \trianglerighteq \vartheta$?" is PSPACE-hard, and PSPACE-complete for acyclic MDPs
- there exists a saturation point $\wp$ such that optimal schedulers behave memoryless from reward $\wp$ on
- pseudo-polynomial threshold algorithm

If $\mathbb{CE}^{\mathsf{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\mathsf{max}}$
- threshold problem "is $\mathbb{CE}^{\mathsf{max}} \trianglerighteq \vartheta$?" is PSPACE-hard, and PSPACE-complete for acyclic MDPs
- there exists a saturation point $\wp$ such that optimal schedulers behave memoryless from reward $\wp$ on
- pseudo-polynomial threshold algorithm: generates a scheduler σ s.t. $\mathbb{CE}^{\sigma} > \vartheta$ or $\mathbb{CE}^{\mathsf{max}} = \mathbb{CE}^{\sigma} = \vartheta$ (if existent)

If $\mathbb{CE}^{\max} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\max}$
- threshold problem "is $\mathbb{CE}^{\max} \trianglerighteq \vartheta$?" is PSPACE-hard, and PSPACE-complete for acyclic MDPs
- there exists a saturation point $\wp$ such that optimal schedulers behave memoryless from reward $\wp$ on
- pseudo-polynomial threshold algorithm: generates a scheduler σ s.t. $\mathbb{CE}^{\sigma} > \vartheta$ or $\mathbb{CE}^{\max} = \mathbb{CE}^{\sigma} = \vartheta$
- exponential-time algorithm to compute $\mathbb{CE}^{\max}$
  interleaves scheduler-improvement steps with threshold algorithm

Computing an upper bound

Computing an upper bound

unconditional total expected reward in a new MDP

Computing an upper bound

unconditional total expected reward in a new MDP $\mathcal{N}$ that simulates $\mathcal{M}$ under the condition $\Diamond \mathit{goal}$

Computing an upper bound

unconditional total expected reward in a new MDP $\mathcal{N}$ that simulates $\mathcal{M}$ under the condition $\Diamond goal$

first mode:

- augments states with the reward accumulated so far up to $R^{max} = \sum_{s} \max_{\alpha} rew(s, \alpha)$
- reward 0 for all state-actions in the first mode
- mode switch from (s, r) via action α with reward r' if $r' \stackrel{\text{def}}{=} r + rew(s, \alpha) > R^{\max}$

second mode: simulation of $\mathcal{M}$ (without reward-annotations)

Computing an upper bound

unconditional total expected reward in a new MDP $\mathcal{N}$ that simulates $\mathcal{M}$ under the condition $\Diamond goal$

first mode:

- augments states with the reward accumulated so far up to $R^{max} = \sum_s \max_\alpha rew(s, \alpha)$
- reward 0 for all state-actions in the first mode
- mode switch from (s, r) via action α with reward r' if $r' \stackrel{\text{def}}{=} r + rew(s, \alpha) > R^{max}$

second mode: simulation of $\mathcal{M}$ (without reward-annotations)

reset transitions:

from all fail states to $\mathcal{N}$'s initial state $(s_0, 0)$

Sketch of the threshold algorithm

compute the saturation point $\wp$ and optimal decisions for state-reward pairs (s, r) with $r \geqslant \wp$

`FOR` $r = \wp{-}1, \wp{-}2, \ldots, 0$ `DO`

compute most feasible actions for the state-reward pairs (s, r) using

- decisions for (s', r') with $r' > r$
- a linear program to treat zero-reward actions

`OD`

check if $\mathbb{CE}^{\sigma} \trianglerighteq \vartheta$ for the generated scheduler σ

Sketch of the threshold algorithm

compute the saturation point $\wp$ and optimal decisions for state-reward pairs (s, r) with $r \geqslant \wp$

$\texttt{FOR}\ r = \wp{-}1, \wp{-}2, \ldots, 0\ \texttt{DO}$

compute most feasible actions for the state-reward pairs (s, r) using

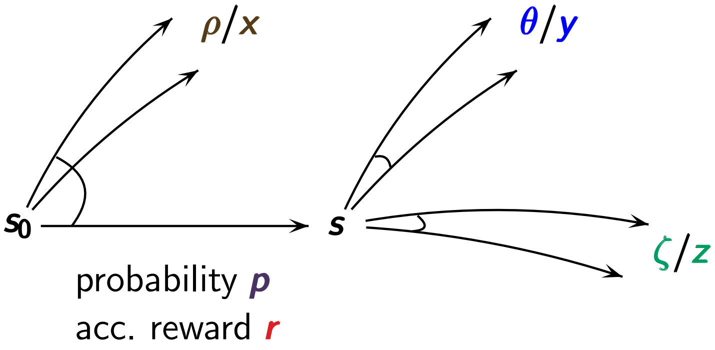
- decisions for (s', r') with $r' > r$
- a linear program to treat zero-reward actions

$\texttt{OD}$

check if $\mathbb{CE}^{\sigma} \trianglerighteq \vartheta$ for the generated scheduler σ

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$



Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

↑ does not depend on ρ, x, p

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

threshold algorithm:

$$r + \frac{\theta - \zeta}{y - z} \geqslant \vartheta \quad \text{iff} \quad \theta - (\vartheta - r)y \geqslant \zeta - (\vartheta - r)z$$

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

threshold algorithm:

$$r + \frac{\theta - \zeta}{y - z} \geqslant \vartheta \quad \text{iff} \quad \theta - (\vartheta - r)y \geqslant \zeta - (\vartheta - r)z$$

... use LP-techniques to maximize $\theta - (\vartheta - r)y$

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

saturation point: smallest value r such that

$$r + \frac{\theta - \zeta}{y - z} \geqslant \mathbb{CE}^{\max} \qquad \text{for all } \tau$$

where σ maximizes the probabilities for reaching the goal

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

saturation point: smallest value r such that

$$r + \frac{\theta - \zeta}{y - z} \geqslant \mathbb{CE}^{\text{ub}} \qquad \text{for all } \tau$$

where σ maximizes the probabilities for reaching the goal

Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0,1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

saturation point: smallest value r such that

$$r + \frac{\theta - \zeta}{y - z} \geqslant \mathbb{CE}^{\mathrm{ub}} \qquad \text{for all } \tau$$

... it suffices to consider "one-step variants" τ of σ

Computing the maximal conditional expectation

using a scheduler-improvement approach with iterative calls of the threshold algorithm

If $\mathbb{CE}^{\max} \geqslant \vartheta$ then the threshold algorithm generates a scheduler σ s.t. $\mathbb{CE}^{\sigma} > \vartheta$ or $\mathbb{CE}^{\max} = \mathbb{CE}^{\sigma} = \vartheta$.

Computing the maximal conditional expectation

using a scheduler-improvement approach with iterative calls of the threshold algorithm

let σ be an arbitrary scheduler;

REPEAT

$\vartheta := \mathbb{CE}^{\sigma}$;

$\sigma :=$ outcome of the algorithm for threshold ϑ

UNTIL $\mathbb{CE}^{\sigma} = \vartheta$

computation of an optimal scheduler

If $\mathbb{CE}^{\max} \geqslant \vartheta$ then the threshold algorithm generates a scheduler σ s.t. $\mathbb{CE}^{\sigma} > \vartheta$ or $\mathbb{CE}^{\max} = \mathbb{CE}^{\sigma} = \vartheta$.

Computing the maximal conditional expectation

using a scheduler-improvement approach with iterative calls of the threshold algorithm

let σ be ...

REPEAT

$\vartheta := \mathbb{CE}^{\sigma}$;

$\sigma :=$ outcome of the algorithm for threshold ϑ

UNTIL $\mathbb{CE}^{\sigma} = \vartheta$

time complexity: double exponential

If $\mathbb{CE}^{\max} \geqslant \vartheta$ then the threshold algorithm generates a scheduler σ s.t. $\mathbb{CE}^{\sigma} > \vartheta$ or $\mathbb{CE}^{\max} = \mathbb{CE}^{\sigma} = \vartheta$.

Computing the maximal conditional expectation

using a scheduler-improvement approach with iterative calls of the threshold algorithm

let σ be . . .

REPEAT

$\vartheta := \mathbb{CE}^{\sigma}$;

$\sigma :=$ outcome of the algorithm for threshold ϑ

UNTIL $\mathbb{CE}^{\sigma} = \vartheta$

time complexity: double exponential

in the worst-case: $|\mathrm{MD}|^{\wp}$ iterations where the saturation point $\wp$ can be exponential in $size(\mathcal{M})$

Computing the maximal conditional expectation

exponential-time algorithm for computing $\mathbb{CE}^{\mathsf{max}}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

Computing the maximal conditional expectation

exponential-time algorithm for computing $\mathbb{CE}^{\max}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

If this holds for all τ then σ is optimal for level r.

Computing the maximal conditional expectation

exponential-time algorithm for computing $\mathbb{CE}^{\max}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad \boxed{r + \frac{\theta - \zeta}{y - z}} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

↑ use these values as threshold values

Computing the maximal conditional expectation

exponential-time algorithm for computing $\mathbb{CE}^{\max}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

$$\mathbb{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \qquad \mathbb{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\mathbb{CE}^{\sigma} > \mathbb{CE}^{\tau} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max\left\{\mathbb{CE}^{\sigma}, \mathbb{CE}^{\tau}\right\}$$

in total: $\mathcal{O}(\wp \cdot |\mathrm{MD}|)$ scheduler-improvement steps

Summary

model checking for systems with discrete probabilities

- Markov chains:
  - linear equation systems (reachability probabilities)
  - analysis of BSCCs (long-run properties)

- Markov decision processes:
  - linear programs (max. reachability prob.)
  - analysis of end components (long-run properties)

Active research area ...

- logics and algorithms for weighted Markovian models
- multi-objective reasoning for MDPs
- parametric model checking for Markovian models
- continuous-time and -space
- probabilistic real-time/hybrid systems
- stochastic games
- various techniques for state-explosion problem
- applications in system biology, security, ...

...

Tool support

PRISM	various models and logics	(Oxford, Birmingham)
	symbolic, explicit and hybrid engines	
STORM	PCTL, bisimulation	(Aachen)
	parametric models	
Modest	MDPs (with clocks)	(Saarbrücken, Twente)
PARAM	parametric models	(Saarbrücken)
ProbDiVinE	parallel LTL model checker	(Brno)
iscasMC	lazy determinization	(Beijing, Liverpool)
⋮	⋮	

Thank You