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

process 1 tosses a coin

process 2 tosses a coin
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)

nondeterministic choice

probabilistic choice
From TS and MC to MDP

transition system
purely nondeterministic

Markov chain
purely probabilistic

Markov decision process (MDP)

integer weights $\text{wgt}(s, \alpha) \in \mathbb{Z}$
Markov decision process (MDP)

\[ M = (S, Act, P, \ldots) \]

- finite state space \( S \)
- \textit{Act} finite set of actions
Markov decision process (MDP)

\[ \mathcal{M} = (S, \text{Act}, P, \ldots) \]

- finite state space \( S \)
- \( \text{Act} \) finite set of actions
- transition probability fct. \( P : S \times \text{Act} \times S \to [0, 1] \)

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

nondeterministic choice between enabled actions

\[ \text{Act}(s) = \{\alpha, \beta\} \]
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. \( P : S \times Act \times S \rightarrow [0, 1] \)
  \[ \forall s \in S \forall \alpha \in Act. \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\} \]
- reward functions \( rew_1, rew_2, \ldots : S \times Act \rightarrow \mathbb{N} \)
  - energy
  - utility
Weighted MDP

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

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

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

- weight functions \( wgt_1, wgt_2, \ldots : S \times Act \rightarrow \mathbb{Z} \)

energy level
of a battery

win and loss
of a share at the
stock market
Weighted MDP

\[ \mathcal{M} = (S, \text{Act}, P, wgt_1, wgt_2, \ldots) \]

- finite state space \( S \)
- \( \text{Act} \) finite set of actions
- transition probability fct. \( P : S \times \text{Act} \times S \rightarrow [0, 1] \)
  \[ \forall s \in S \quad \forall \alpha \in \text{Act}. \quad \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\} \]
- weight functions \( wgt_1, wgt_2, \ldots : S \times \text{Act} \rightarrow \mathbb{Z} \)

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

\[ \mathcal{M} = (S, \text{Act}, P, wgt_1, wgt_2, \ldots) \]

- finite state space \( S \)
- \( \text{Act} \) finite set of actions
- transition probability fct. \( P : S \times \text{Act} \times S \to [0, 1] \)
  \[ \forall s \in S \quad \forall \alpha \in \text{Act}. \quad \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\} \]
- weight functions \( wgt_1, wgt_2, \ldots : S \times \text{Act} \to \mathbb{Z} \)

ratios of accumulated weights:

\[ \text{ratio} = \frac{\text{cost}}{\text{util}} : \text{FinPaths} \to \mathbb{Q} \quad \text{cost} = wgt_1 \quad \text{util} = wgt_2 \]
Probability measure

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

- finite state space \( S \)
- \( \text{Act} \) finite set of actions
- transition probability fct. \( P : S \times \text{Act} \times S \to [0, 1] \)
  \( \forall s \in S \ \forall \alpha \in \text{Act}. \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\} \)
- weight functions \( wgt_1, wgt_2, \ldots : S \times \text{Act} \to \mathbb{Z} \)

probabilities measure \( \Pr_s^\sigma \) for given state \( s \in S \) and scheduler \( \sigma : \text{FinPaths} \to \text{Distr}(\text{Act}) \)
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 \((\text{Mem}, \mu, \nu)\)

- \(\text{Mem}\) is a set of memory cells
- \(\mu: \text{Mem} \times S \rightarrow \text{Distr}(\text{Act})\) decision function
- \(\nu: \text{Mem} \times S \rightarrow \text{Mem}\) 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:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_i$</td>
<td>noncritical actions of process $P_i$</td>
</tr>
<tr>
<td>$w_i$</td>
<td>waiting phase of process $P_i$</td>
</tr>
<tr>
<td>$c_i$</td>
<td>critical section of process $P_i$</td>
</tr>
</tbody>
</table>

- competition if both processes are waiting
Randomized mutual exclusion protocol

- 2 concurrent processes $P_1, P_2$ with 3 phases:

  \[ n_i \text{ noncritical actions of process } P_i \]
  \[ w_i \text{ waiting phase of process } P_i \]
  \[ c_i \text{ critical section of process } P_i \]

- competition if both processes are waiting

- resolved by a randomized arbiter who tosses a coin
Randomized mutual exclusion protocol

- 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
Randomized mutual exclusion protocol

- interleaving of the request operations
- \textbf{competition} if both processes are \textit{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

\[ n_1 n_2 \]

\[ w_1 n_2 \]

\[ w_1 w_2 \]

\[ n_1 w_2 \]

\[ c_1 n_2 \]

\[ c_1 w_2 \]

\[ w_1 c_2 \]

\[ n_1 c_2 \]

**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?
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 \( \langle n_1, w_2 \rangle \)
The probability that process 2 enters its critical section within the next 3 steps is:

\[ \frac{1}{2} \quad \text{if process 1 is scheduled in state } \langle n_1, w_2 \rangle \]
\[ 1 \quad \text{if process 2 is scheduled in state } \langle n_1, w_2 \rangle \]
Probabilistic model checking

- Probabilistic reactive system
- Probabilistic model: MDP $\mathcal{M}$
- Quantitative requirements
- Temporal formula $\varphi$, e.g., LTL formula

Best- or worst-case probability: $\Pr^{\min}(\varphi)$ or $\Pr^{\max}(\varphi)$
Probabilistic model checking

probabilistic reactive system

probabilistic model
MDP $\mathcal{M}$

temporal formula $\varphi$
e.g. LTL formula

probabilistic model checking

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

extrema over all schedulers
Probabilistic model checking

probabilistic reactive system

probabilistic reachability analysis of \( M \otimes A \)

linear programming

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

deterministic automaton \( A \)

temporal formula \( \varphi \)
e.g. LTL formula

quantitative requirements

probabilistic model \( M \)

MDP
Probabilistic model checking

probabilistic reactive system

probabilistic model

MDP $\mathcal{M}$

probabilistic reachability analysis of $\mathcal{M} \otimes \mathcal{A}$

linear programming

deterministic automaton $\mathcal{A}$

temporal formula $\varphi$
e.g. LTL formula

quantitative requirements

maximal probability to reach an accepting end component

$$\Pr_{\mathcal{M},s}(\varphi) = \Pr_{\mathcal{M} \otimes \mathcal{A},s'}(\diamond accEC)$$
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.
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 $\emptyset \neq T \subseteq S$ and $A : T \rightarrow 2^{Act}$ s.t.

(1) \ldots

(2) \ldots

(3) \ldots
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 $\emptyset \neq T \subseteq S$ and $A : T \rightarrow 2^{Act}$ s.t.

1. enabledness of selected actions:
   \[ \emptyset \neq A(t) \subseteq Act(t) \quad \text{for all } t \in T \]

2. \ldots

3. \ldots
End components (EC)  

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 $\emptyset \neq T \subseteq S$ and $A : T \to 2^{Act}$ s.t.

(1) enabledness of selected actions:
\[ \emptyset \neq A(t) \subseteq Act(t) \] for all $t \in T$

(2) closed under probabilistic branching:
\[ \forall t \in T \forall \alpha \in A(t). \ (P(t, \alpha, u) > 0 \implies u \in T) \]

(3) ...
End components (EC)  

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

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

1. enabledness of selected actions:
   \[
   \emptyset \neq A(t) \subseteq \text{Act}(t) \quad \text{for all } t \in T
   \]

2. closed under probabilistic branching:
   \[
   \forall t \in T \forall \alpha \in A(t). \ (P(t, \alpha, u) > 0 \implies u \in T)
   \]

3. the underlying graph is strongly connected
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 $\emptyset \neq T \subseteq S$ and $A : T \rightarrow 2^{\text{Act}}$ 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)
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 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 $\sigma$ and states $s$:

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

limit of an infinite path $\pi$:

$$\text{limit}(\pi) = \left\{ \text{set of state-action pairs that appear infinitely often in } \pi \right\}$$

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 $\sigma$ and states $s$:

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

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

$$\pi \models E \iff \exists i \geq 0. \inf(\pi) = T_i$$

set of states that appear infinitely often in $\pi$
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 $\sigma$ and states $s$:

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

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

$$\pi \models E \iff \exists i \geq 0. \inf(\pi) = T_i$$

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

$$T = \bigcup \{ T_i : T_i \text{ constitutes an end component} \}$$
Quantitative analysis of Rabin conditions
Quantitative analysis of Rabin conditions

Let $E$ be a Rabin condition $\forall (◊□¬L_i \land □◊U_i)$.  

◊ eventually  ◊□ almost forever  
□ always  □◊ infinitely often
Quantitative analysis of Rabin conditions

Let $E$ be a Rabin condition $\bigvee_{1 \leq i \leq k} (\lozenge \square \neg L_i \land \square \lozenge U_i)$.

$$\Pr_s^\text{max}(E) = \Pr_s^\text{max}(\lozenge \text{accEC})$$

◊ eventually
☐ always
☐☐ almost forever
☐☐☐ infinitely often
Quantitative analysis of Rabin conditions

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

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

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

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

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

$\Box$ always  \quad $\Box \Diamond$ infinitely often
Quantitative analysis of Rabin conditions

Let $E$ be a Rabin condition $\bigvee_{1 \leq i \leq k} (\lozenge \square \neg L_i \land \square \lozenge U_i)$.

\[
\Pr_s^\max(E) = \Pr_s^\max(\lozenge \text{accEC}) = \Pr_s^\max(\lozenge \text{accMEC})
\]

union of all maximal end components $T$

in $M \setminus L_i$ s.t. $T \cap U_i \neq \emptyset$

\begin{itemize}
  \item \lozenge eventually
  \item \square almost forever
  \item \square\lozenge infinitely often
  \item \square\square always
\end{itemize}
Quantitative analysis of Rabin conditions

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

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

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

union of all maximal end components $T$ in $M \setminus L_i$ s.t. $T \cap U_i \neq \emptyset$

analogous approach for generalized Rabin conditions:

$$\bigvee_{1 \leq i \leq k} (\Diamond \Box \neg L_i \land \Box \Diamond U_{i,1} \land \ldots \land \Box \Diamond U_{i,k_i})$$
Quantitative analysis of Rabin conditions

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

\[
\Pr_s^{\max}(E) = \Pr_s^{\max}(\Diamond \text{accEC}) \\
= \Pr_s^{\max}(\Diamond \text{accMEC})
\]

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 $\bigvee_{1 \leq i \leq k} (\Diamond \square \neg L_i \land \square \Diamond U_i)$.

$$Pr_{s}^{\text{max}}(E) = Pr_{s}^{\text{max}}(\Diamond accEC)$$

$$= Pr_{s}^{\text{max}}(\Diamond accMEC)$$

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 $\alpha$ 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 $\alpha$ such that

$P(s, \alpha, t) > 0$ and $s, t$ belong to different SCCs

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

remove $\alpha$ from $\text{Act}(s)$;

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

compute the SCCs of $M$;

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

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

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 $\mathcal{M}$;

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

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

IF $s$ is a trap THEN remove $s$ FI

FI

UNTIL no further changes
Computation of maximal end components

REPEAT

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

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

THEN choose such a pair $\langle s, \alpha \rangle$;
    remove $\alpha$ from $\text{Act}(s)$;
    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 \( \alpha \) such that
\[
P(s, \alpha, t) > 0
\]
and \( s, t \) belong to different SCCs

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

remove \( \alpha \) from \( \text{Act}(s) \);

IF \( s \) is a trap THEN remove \( s \) FI

FI

UNTIL no further changes

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

return the non-trivial SCCs as maximal end components
MEC-quotient

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

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

Idea: The MEC-quotient is the MDP $\text{MEC}(\mathcal{M})$ resulting from $\mathcal{M}$ by collapsing all MECs into a single state.
Given MDP $\mathcal{M} = (S, \text{Act}, P, \ldots)$ 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:

$$\text{Act}(s) \cap \text{Act}(t) = A_i(s) \cap A_i(t)$$

Idea: The MEC-quotient is the MDP $\text{MEC}(\mathcal{M})$ 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:

$$\text{Act}(s) \cap \text{Act}(t) = A_i(s) \cap A_i(t)$$

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

$$S' = (S \setminus T) \cup \{\mathcal{E}_1, \ldots, \mathcal{E}_k\}$$

where $T = \bigcup_{1 \leq i \leq k} T_i$

Idea: The MEC-quotient is the MDP $\text{MEC}(\mathcal{M})$ 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:

$$\text{Act}(s) \cap \text{Act}(t) = A_i(s) \cap A_i(t)$$

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

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

enabled actions:

- for $s \in S \setminus T$: as in $\mathcal{M}$
- for state $\mathcal{E}_i$: all actions in $\bigcup_{s \in T_i} \text{Act}(s) \setminus A_i(s)$
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:

$$\text{Act}(s) \cap \text{Act}(t) = A_i(s) \cap A_i(t)$$

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

$$S' = (S \setminus T) \cup \{\mathcal{E}_1, \ldots, \mathcal{E}_k\}$$

where $T = \bigcup_{1 \leq i \leq k} T_i$

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

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

$$P'(s, \alpha, \mathcal{E}_i) = \sum_{t \in T_i} P(s, \alpha, t)$$
Given MDP $\mathcal{M} = (S, \text{Act}, P, \ldots)$ 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:

$$\text{Act}(s) \cap \text{Act}(t) = A_i(s) \cap A_i(t)$$

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

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

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

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

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

$$\Pr^\text{max}_s(\varphi) = \Pr^\text{max}_t(\varphi)$$

for each prefix-independent path property $\varphi$.

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

The same holds for minimal probabilities for prefix-independent properties and min/max expectations of long-run objectives.
For all states \(s, t\) that belong to the same MEC:

\[
Pr_{s}^{\text{max}}(\varphi) = Pr_{t}^{\text{max}}(\varphi)
\]

for each prefix-independent path property \(\varphi\).

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

Examples: \(\varphi = \Diamond G\) or \(\varphi = \Diamond \Box G\) or ...

The same holds for minimal 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:

\[
Pr_{s}^{\text{max}}(\varphi) = Pr_{t}^{\text{max}}(\varphi)
\]

for each prefix-independent path property \( \varphi \).

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

\text{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:

\[
\Pr_{s}^{\text{max}}(\varphi) = \Pr_{t}^{\text{max}}(\varphi)
\]

for each prefix-independent path property \( \varphi \).

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

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

\[
\Pr_{\text{MEC}(\mathcal{M}),s}^{\min}(\Diamond \text{Trap}) = 1
\]

set of states \( t \) with \( \text{Act}(t) = \emptyset \)
Properties of the MECs and the MEC-quotient

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

\[
\Pr_s^{\text{max}}(\varphi) = \Pr_t^{\text{max}}(\varphi)
\]

for each prefix-independent path property \( \varphi \).

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

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

\[
\Pr_{\text{MEC}(\mathcal{M}),s}^{\text{min}}(\Diamond \text{Trap}) = 1
\]

... transition probability matrix is contracting ...
Probabilistic model checking

probabilistic reactive system

probabilistic model

MDP \( \mathcal{M} \)

deterministic automaton \( \mathcal{A} \)

temporal formula \( \varphi \)
e.g. LTL formula

probabilistic reachability analysis of \( \mathcal{M} \otimes \mathcal{A} \)

linear programming

maximal probability to reach an accepting end component

\[
\Pr_{\mathcal{M},s}^{\text{max}}(\varphi) = \Pr_{\mathcal{M} \otimes \mathcal{A},s'}^{\text{max}}(\Diamond \text{accEC})
\]
Maximal reachability probabilities
Maximal reachability probabilities

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

task: compute $x_s = \Pr^\max_s(\Diamond G) = \max_\sigma \Pr^\sigma_s(\Diamond G)$
Maximal reachability probabilities

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

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

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

\[
\begin{align*}
x_s &= 1 \quad \text{if } s \in G \\
x_s &= \max_{\alpha} \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} \quad \text{if } s \notin G
\end{align*}
\]

$\alpha$ ranges over all actions in $\text{Act}(s)$
The vector \((x_s)_{s \in S}\) where \(x_s = \Pr_s^\text{max}(\diamond G)\) is the least solution of

\[
x_s = \begin{cases} 1 & \text{if } s \in G \\ \max_{\alpha} \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} & \text{if } s \notin G \end{cases}
\]
\[ x_s = \max \left\{ \frac{1}{3} x_{u_1} + \frac{2}{3} x_{u_2}, \quad \frac{1}{4} x_{u_3} + \frac{3}{4} x_{u_4} \right\} \]
$G = \{ t_1, t_2 \}$

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

$$x_{u_1} = x_{t_1}$$
\[ x_s = \max\left\{ \frac{1}{3}x_{u_1} + \frac{2}{3}x_{u_2}, \frac{1}{4}x_{u_3} + \frac{3}{4}x_{u_4} \right\} \]

\[ x_{u_1} = x_{t_1} = 1 \]

\[ t_1 \in G \]
\[ G = \{ t_1, t_2 \} \]

\[ x_s = \max \left\{ \frac{1}{3} x_{u_1} + \frac{2}{3} x_{u_2}, \frac{1}{4} x_{u_3} + \frac{3}{4} x_{u_4} \right\} \]

\[ x_{u_1} = x_{t_1} = 1 \]

\[ x_{u_2} = x_{u_3} \]

unique successor of state \( u_2 \)
$x_s = \max\{ \frac{1}{3}x_{u_1} + \frac{2}{3}x_{u_2}, \frac{1}{4}x_{u_3} + \frac{3}{4}x_{u_4} \}$

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

$x_{u_2} = x_{u_3} = 0$  least solution of $x_{u_3} = x_{u_3}$

unique successor of state $u_2$

$G = \{ t_1, t_2 \}$
\[ G = \{ t_1, t_2 \} \]

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

\[ x_{u_1} = x_{t_1} = 1 \]

\[ x_{u_2} = x_{u_3} = 0 \]  

\[ x_{t_2} = 1 \]  

least solution of \( x_{u_3} = x_{u_3} \)
\[ G = \{ t_1, t_2 \} \]

\[
x_s = \max \left\{ \frac{1}{3}x_{u_1} + \frac{2}{3}x_{u_2}, \frac{1}{4}x_{u_3} + \frac{3}{4}x_{u_4} \right\}
\]

\[
x_{u_1} = x_{t_1} = 1
\]

\[
x_{u_2} = x_{u_3} = 0 \quad \text{least solution of } x_{u_3} = x_{u_3}
\]

\[
x_{t_2} = 1
\]

\[
x_{u_4} = \frac{1}{2}x_{u_4} + \frac{1}{2}x_{t_2} = \frac{1}{2}x_{u_4} + \frac{1}{2} = 1
\]
\[ G = \{ t_1, t_2 \} \]

\[
\begin{align*}
x_s &= \max \left\{ \frac{1}{3} x_{u_1} + \frac{2}{3} x_{u_2}, \quad \frac{1}{4} x_{u_3} + \frac{3}{4} x_{u_4} \right\} = \frac{3}{4} \\
x_{u_1} &= x_{t_1} = 1 \\
x_{u_2} &= x_{u_3} = 0 \\
x_{t_2} &= 1 \\
x_{u_4} &= \frac{1}{2} x_{u_4} + \frac{1}{2} x_{t_2} = \frac{1}{2} x_{u_4} + \frac{1}{2} = 1
\end{align*}
\]
Maximal reachability probabilities
Maximal reachability probabilities

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

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

\[
x_s = \begin{cases} 
1 & \text{if } s \in G \\
\max_{\alpha} \sum_{t \in S} P(s, \alpha, t) \cdot x_t & \text{otherwise}
\end{cases}
\]

\(\alpha\) ranges over all actions in $\text{Act}(s)$
Maximal reachability probabilities

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

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

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

\[
\begin{align*}
    x_s &= 1 \quad \text{if } s \in G \\
    x_s &= 0 \quad \text{if } s \not\in \exists \Diamond G \\
    x_s &= \max_{\alpha} \sum_{t \in S} P(s, \alpha, t) \cdot x_t \quad \text{otherwise}
\end{align*}
\]

$\alpha$ ranges over all actions in $\text{Act}(s)$
Maximal reachability probabilities

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

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

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

\[
\begin{align*}
    x_s &= 1 \quad \text{if } s \in G \\
    x_s &= 0 \quad \text{if } s \not\in \exists \diamond G \\
    x_s &= \max_\alpha \sum_{t \in S} P(s, \alpha, t) \cdot x_t \quad \text{otherwise}
\end{align*}
\]

“Bellman equations”
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:

$$
\begin{align*}
    x_s &= 1 & \text{if } s \in G \\
    x_s &= 0 & \text{if } s \not\in \exists \diamond G \\
    x_s &= \max_\alpha \sum_{t \in S} P(s, \alpha, t) \cdot x_t & \text{otherwise}
\end{align*}
$$

... induces an optimal MD-scheduler ...
Maximal reachability probabilities

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

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

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

\[
\begin{align*}
x_s &= 1 & \text{if } s \in G^* \\
x_s &= 0 & \text{if } s \not\in \exists \Diamond G \\
x_s &= \max_{\alpha} \sum_{t \in S} P(s, \alpha, t) \cdot x_t & \text{otherwise}
\end{align*}
\]

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 $x_s = \Pr^\text{max}_s(\diamond G) = \max_\sigma \Pr^\sigma_s(\diamond G)$

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

$$
\begin{align*}
  x_s &= 1 \quad \text{if } s \in G^* \\
  x_s &= 0 \quad \text{if } s \not\in \exists \diamond G \\
  x_s &= \max_\alpha \sum_{t \in S} P(s, \alpha, t) \cdot x_t \quad \text{otherwise}
\end{align*}
$$

if $\mathcal{M}$ has no end components
Maximal reachability probabilities

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

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

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

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

if $\mathcal{M}$ has no end components
Maximal reachability probabilities

given: MDP $\mathcal{M}$ with state space $S$ and $G \subseteq S$
task: compute $x_s = \Pr_s^{\max}(\diamond G) = \max_{\sigma} \Pr_s^{\sigma}(\diamond G)$

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

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

if $\mathcal{M}$ has no end components or if $x_s^{(0)} \leq x_s$
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: $x_s = \lim_{n \to \infty} x_s^{(n)}$

\[
x_s^{(n)} = 1 \quad \text{if } s \in G^*
\]

\[
x_s^{(n)} = 0 \quad \text{if } s \not\in \exists \Diamond G
\]

\[
x_s^{(n)} = \max_{\alpha} \sum_{t \in S} P(s, \alpha, t) \cdot x_t^{(n-1)} \quad \text{else}
\]

... 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: $x_s = \lim_{n \to \infty} x^{(n)}_s$

| $x^{(n)}_s$ | if $s \in G^*$ |
| $x^{(n)}_s$ | if $s \not\models \exists \diamond G$ |
| $x^{(n)}_s$ | else |

$\sum_{t \in S} P(s, \alpha, t) \cdot x^{(n-1)}_t$

... 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:

$$
\begin{align*}
    x_s &= 1 & \text{if } s \in G^* \\
    x_s &= 0 & \text{if } s \not\models \exists \diamond G \\
    x_s &\geq \sum_{t \in S} P(s, \alpha, t) \cdot x_t & \text{for } \alpha \in \text{Act}(s)
\end{align*}
$$
Maximal reachability probabilities via LP

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

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

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

$$
x_s = 1 \quad \text{if } s \in G^* \\
x_s = 0 \quad \text{if } s \not\in \exists \diamond G \\
x_s \geq \sum_{t \in S} P(s, \alpha, t) \cdot x_t \quad \text{for } \alpha \in \text{Act}(s)
$$

where $\sum_{s \in S} x_s$ is minimal
Least vs unique solution

\[
\begin{align*}
\tau & \quad u \\
\gamma & \quad t \\
\beta & \quad s \\
\alpha, \frac{1}{2} & \quad \text{goal} \\
\alpha, \frac{1}{2} & \quad \text{fail}
\end{align*}
\]
Bellmann equations:

\[ x_u = x_u \]

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

\[ x_t = x_s \]
Least vs unique solution

Bellman equations:

\[ x_u = 0 \]
\[ x_s = \max \{ x_t, \frac{1}{2} \} \]

as \( u \not\models \exists \diamond \text{goal} \)

\[ x_t = x_s \]
Least vs unique solution

Bellmann equations:

\[ x_u = 0 \]
\[ \text{as } u \not\models \exists \Diamond \text{goal} \]

\[ x_s = \max \{ x_t, \frac{1}{2} \} \]
\[ x_t = x_s \]

solutions:
\[ x_t = x_s \geq \frac{1}{2} \]
Least vs unique solution

Bellmann equations:

\[ x_u = 0 \]

as \( u \not\models \Diamond \text{goal} \)

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

\[ x_t = x_s \]

least solution:

\[ x_t = x_s = \frac{1}{2} \]
Least vs unique solution

Bellmann equations:

\[ x_u = 0 \]

as \( u \not\models \exists \Diamond \text{goal} \)

\[ x_s = \max \{ x_t, \frac{1}{2} \} \]

\[ x_t = x_s \]

least solution:

\[ x_t = x_s = \frac{1}{2} \]
Least vs unique solution

Bellmann equations:

\[ x_u = 0 \]
as \( u \not\models \exists \diamond \text{goal} \)

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

\[ x_t = x_s \]

unique solution:

\[ x_{\{s,t\}} = \frac{1}{2} \]
Stochastic shortest/longest path problem

weighted MDP $\mathcal{M}$

$\forall \text{goal}$ accumulated weight until reaching a goal state

best- or worst-case expectation

$\mathbb{E}^\min(\forall \text{goal})$ or $\mathbb{E}^\max(\forall \text{goal})$

extrema over all schedulers
Stochastic shortest/longest path problem

weighted MDP $\mathcal{M}$

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

best- or worst-case expectation
\[ \mathbb{E}^{\text{min}}(\Diamond \text{goal}) \text{ or } \mathbb{E}^{\text{max}}(\Diamond \text{goal}) \]

$\Diamond \text{goal}$ accumulated weight until reaching a goal state

extrema over all schedulers
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 $x_s = E_s^{\max}(\diamond G)$

“stochastic longest path”
Maximal expected accumulated weight

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

task: compute $x_s = \mathbb{E}_{s}^{\max}(\Box G)$

“stochastic longest path”

random variable $\Box G : \text{MaxPaths} \rightarrow \mathbb{Z}$

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

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

if $\pi \not\models \Box G$ then $(\Box G)(\pi) = \bot$ “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 $x_s = E_s^{\max}(\Diamond G)$

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:

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

“Bellman equations”
Maximal expected accumulated weight

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

task: compute $x_{s} = E_{s}^{\max}(\Diamond G)$

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:

$$x_{s} = \max_{\alpha \in \text{Act}(s)} \left( \text{wgt}(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_{t} \right)$$

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

[Bertsekas/Tsitsiklis’91]
Maximal expected accumulated weight

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

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

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:

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

... 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. $\Pr_s^{\min}(\Diamond G) = 1$ for all states $s$

task: compute $x_s = E_s^\max(\Diamond G)$

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:

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

value iteration (arbitrary starting vector)  

[Bertsekas/Tsitsiklis’91]
Maximal expected accumulated weight

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

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

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 $\alpha \in \text{Act}(s)$:

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

where $\sum_{s \in S} x_s$ is minimal
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
PCTL* over MDPs

- syntax of state and path formulas as for PCTL* over Markov chains
- probability operator $P_I(\ldots)$ ranges over all schedulers
state formulas:

\[ \phi ::= \text{true} \mid a \mid \phi_1 \land \phi_2 \mid \neg \phi \mid \mathcal{P}_I(\varphi) \]

path formulas:

\[ \varphi ::= \phi \mid \varphi_1 \land \varphi_2 \mid \neg \varphi \mid \bigcirc \varphi \mid \varphi_1 \mathcal{U} \varphi_2 \]
PCTL* over MDPs

[Bianco/de Alfaro’95]

state formulas:

\[ \Phi \ ::= \ true \mid a \mid \Phi_1 \land \Phi_2 \mid \neg \Phi \mid P_1(\varphi) \]

path formulas:

\[ \varphi \ ::= \ \Phi \mid \varphi_1 \land \varphi_2 \mid \neg \varphi \mid \bigcirc \varphi \mid \varphi_1 \cup \varphi_2 \]

given an MDP \( M \), define by structural induction:

- a satisfaction relation \( \models \) for states \( s \) in \( M \) and PCTL* state formulas \( \Phi \)
- a satisfaction relation \( \models \) for infinite paths \( \pi \) in \( M \) and PCTL* path formulas \( \varphi \)
Satisfaction relation for PCTL* state formulas

\( s \models true \)

\( s \models a \quad \text{iff} \quad a \in L(s) \)

\( s \models \Phi_1 \land \Phi_2 \quad \text{iff} \quad s \models \Phi_1 \quad \text{and} \quad s \models \Phi_2 \)

\( s \models \neg \Phi \quad \text{iff} \quad s \not\models \Phi \)

\( s \models P_1(\varphi) \quad \text{iff} \quad \text{for all schedulers } \sigma : \)

\[ \Pr^\sigma \left\{ \pi \in \text{Paths}(s) : \pi \models \varphi \right\} \in I \]
Satisfaction relation for PCTL* state formulas

\[ s \models true \]
\[ s \models a \quad \text{iff} \quad a \in L(s) \]
\[ s \models \Phi_1 \land \Phi_2 \quad \text{iff} \quad s \models \Phi_1 \quad \text{and} \quad s \models \Phi_2 \]
\[ s \models \neg \Phi \quad \text{iff} \quad s \nvDash \Phi \]
\[ s \models P_1(\varphi) \quad \text{iff} \quad \text{for all schedulers } \sigma: \]
\[ \Pr^\sigma \{ \pi \in Paths(s) : \pi \models \varphi \} \in I \]

probability measure in the Markov chain induced by \( \sigma \)
Satisfaction relation for PCTL* state formulas

\[ s \models true \]
\[ s \models a \quad \text{iff} \quad a \in L(s) \]
\[ s \models \Phi_1 \land \Phi_2 \quad \text{iff} \quad s \models \Phi_1 \quad \text{and} \quad s \models \Phi_2 \]
\[ s \models \neg \Phi \quad \text{iff} \quad s \not\models \Phi \]
\[ s \models P_I(\varphi) \quad \text{iff} \quad \text{for all schedulers } \sigma: \]
\[ \Pr^\sigma \{ \pi \in Paths(s) : \pi \models \varphi \} \in I \]

probability measure in the Markov chain induced by \( \sigma \)

semantics of path formulas as for Markov chains
PCTL* model checking for MDP
PCTL* model checking for MDP

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

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 $\psi$ of $\Phi$
PCTL* model checking for MDP

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

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 $\psi$ of $\Phi$

treatment of the propositional logic fragment: $\checkmark$
Treatment of probability operator
Treatment of probability operator

upper probability bounds $P_{\leq p}(\varphi)$ or $P_{< p}(\varphi)$
Treatment of probability operator

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

- compute the maximal probabilities for $\varphi$

$$\Pr_s^{\max}(\varphi) = \sup_{D} \Pr^{D} \left\{ \pi \in \text{Paths}(s) : \pi \models \varphi \right\}$$

for all states $s$
Treatment of probability operator

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

- compute the maximal probabilities for $\varphi$

$$\operatorname{Pr}_{s}^{\max}(\varphi) = \max_{D} \operatorname{Pr}^{D}\left\{ \pi \in \text{Paths}(s) : \pi \models \varphi \right\}$$

for all states $s$

there exists optimal finite-memory schedulers
Treatment of probability operator

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

- compute the maximal probabilities for $\varphi$

\[
\Pr_s^{\text{max}}(\varphi) = \max_D \Pr^D\{ \pi \in \text{Paths}(s) : \pi \models \varphi \}
\]

for all states $s$

- return $\{s \in S : \Pr_s^{\text{max}}(\varphi) \leq p\}$
Treatment of probability operator

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

- compute the maximal probabilities for $\varphi$

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

for all states $s$

- return $\{ s \in S : \Pr^\text{max}_s(\varphi) \leq p \}$

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

analogous, but minimal probabilities for $\varphi$
Treatment of probability operator

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

compute the maximal probabilities for $\varphi$

$$\Pr_{s}^{\max}(\varphi) = \max_D \Pr^D \{ \pi \in \text{Paths}(s) : \pi \models \varphi \}$$

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

reachability condition
Treatment of probability operator

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

compute the maximal probabilities for $\varphi$

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

special case: $\varphi = \diamond \psi$

compute $\Pr_s^{\max}(\diamond \psi)$ by solving a linear program

↑

maximal reachability probabilities
Treatment of probability operator

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

compute the maximal probabilities for $\varphi$

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

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

compute $\Pr_{s}^{\max}(\Diamond \psi)$ by solving a linear program

general case:

via deterministic automaton $\mathcal{A}$ for $\varphi$ and maximal reachability probabilities in $\mathcal{M} \times \mathcal{A}$
PCTL* model checking for MDP

given: MDP $\mathcal{M} = (S, \text{Act}, P, \ldots)$
PCTL* state formula $\mathbb{P} \leq_P \phi$

task: compute $\text{Sat}(\mathbb{P} \leq_P \phi)$
PCTL* model checking for MDP

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

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

task: compute $\text{Sat}(\mathbb{P} \leq p(\varphi))$

method: compute $x_s = \text{Pr}_s^{\text{max}}(\varphi)$ via a reduction to the probabilistic reachability problem
PCTL* model checking for MDP

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

task: compute $\text{Sat}(\mathbb{P}^{\leq p}(\varphi))$

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

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

DRA: deterministic Rabin automaton
MDP $\mathcal{M}$

PCTL* path formula $\varphi$
MDP $\mathcal{M}$

PCTL* path formula $\varphi$

\[ \varphi' \]

LTL formula $\varphi'$
MDP $\mathcal{M}$

$\text{PCTL}^*$ path formula $\varphi$

$LTL$ formula $\varphi'$

$DRA \ A$
MDP $\mathcal{M}$

PCTL* path formula $\varphi$

LTL formula $\varphi'$

DRA $\mathcal{A}$

product-MDP $\mathcal{M} \times \mathcal{A}$
$$\Pr_{\mathcal{M}}^{\text{max}}(\varphi) = \Pr_{\mathcal{M} \times \mathcal{A}}^{\text{max}}( \bigvee_i (\Diamond \Box \neg L_i \land \Box \Diamond U_i) )$$

acceptance condition of $\mathcal{A}$
\[
\Pr_{\mathcal{M}}^\text{max}(\varphi) = \Pr_{\mathcal{M} \times \mathcal{A}}^\text{max}(\bigvee_i (\Diamond \Box \neg L_i \land \Box \Diamond U_i)) = \Pr_{\mathcal{M} \times \mathcal{A}}^\text{max}(\Diamond \text{accMEC})
\]
Lower probability bounds

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

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

task: compute $\mathit{Sat}(\mathbb{P} \geq p(\varphi))$
Lower probability bounds

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

PCTL* formula $P \geq_p (\varphi)$

task: compute $Sat(P \geq_p (\varphi))$

simple fact: for each scheduler $D$ and state $s$:

$$Pr^D_S(\varphi) = 1 - Pr^D_S(\neg \varphi)$$

... duality of lower and upper probability bounds
Lower probability bounds

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

PCTL* formula $P \geq p(\varphi)$

task: compute $\text{Sat}(P \geq p(\varphi))$

simple fact: for each scheduler $D$ and state $s$:

$$\Pr^D_s(\varphi) = 1 - \Pr^D_s(\neg \varphi)$$

... duality of lower and upper probability bounds

For each state $s$ and PCTL* path formula $\varphi$:

$$\Pr^\text{min}_s(\varphi) = 1 - \Pr^\text{max}_s(\neg \varphi)$$
Complexity of PCTL/PCTL* model checking
## Complexity of PCTL/PCTL* model checking

<table>
<thead>
<tr>
<th></th>
<th>PCTL</th>
<th>PCTL*</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Markov chain</strong></td>
<td><em>PTIME</em></td>
<td><em>PSPACE</em>-complete</td>
</tr>
<tr>
<td></td>
<td>[Hansson/Jonsson’94]</td>
<td>[Vardi/Wolper’86]</td>
</tr>
<tr>
<td><strong>Markov decision process</strong></td>
<td><em>PTIME</em></td>
<td><strong>2EXP</strong>-complete</td>
</tr>
<tr>
<td></td>
<td>[Bianco/deAlfaro’95]</td>
<td>[Courcoubetis/Yannakakis’88]</td>
</tr>
</tbody>
</table>
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 probabilities for MDP

for Markov decision processes:

\[
\Pr_{\mathcal{M},s}^{\max}(\varphi | \psi) = \max_{\sigma} \frac{\Pr_{s}^{\sigma}(\varphi \land \psi)}{\Pr_{s}^{\sigma}(\psi)}
\]

all schedulers \(\sigma\)

with \(\Pr_{s}^{\sigma}(\psi) > 0\)
Conditional probabilities for MDP

for Markov decision processes:

$$\Pr_{\max}^{\mathcal{M},s}(\varphi | \psi) = \max_{\sigma} \frac{\Pr_s^{\sigma}(\varphi \land \psi)}{\Pr_s^{\sigma}(\psi)}$$

exponential-time procedure for PCTL

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

PCTL probabilistic computation tree logic

[Andrés/Rossum’08]
Conditional probabilities for MDP

for Markov decision processes:

\[
\Pr_{\mathcal{M}, s}^{\max}(\varphi | \psi) = \max_{\sigma} \frac{\Pr_{s}^{\sigma}(\varphi \land \psi)}{\Pr_{s}^{\sigma}(\psi)}
\]

exponential-time procedure for PCTL

even for reachability \( \varphi = \Diamond F, \psi = \Diamond G \)

transformation-based approach for LTL

\( \mathcal{M} \leadsto \mathcal{M}_{\varphi | \psi} \) of linear size for reachability

\[
\Pr_{\mathcal{M}, s}^{\max}(\varphi | \psi) = \Pr_{\mathcal{M}_{\varphi | \psi}, s}^{\max}(\varphi')
\]

[Andrés/Rossum’08]

 transformation-based approach for LTL

\( \mathcal{M} \leadsto \mathcal{M}_{\varphi | \psi} \) of linear size for reachability

\[
\Pr_{\mathcal{M}, s}^{\max}(\varphi | \psi) = \Pr_{\mathcal{M}_{\varphi | \psi}, s}^{\max}(\varphi')
\]

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

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

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

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

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

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

three fresh trap states
Transformation-based approach for MDP

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

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

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

soundness:

$$\Pr^\text{max}_{\mathcal{M}, s}(\Diamond F \mid \Diamond G) = \Pr^\text{max}_{\mathcal{M}', s}(\Diamond \text{goal} \mid \Diamond (\text{goal} \lor \text{stop}))$$
Transformation-based approach for MDP

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

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

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

step 2: normal form MDP $\mathcal{M}' \rightsquigarrow \mathcal{M}''$ s.t. ...
Transformation-based approach for MDP

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

$$\Pr_{\mathcal{M}', S_{\text{init}}}^{\max} (\Diamond \text{goal} \mid \Diamond (\text{goal} \lor \text{stop})) = \Pr_{\mathcal{M}'', S_{\text{init}}}^{\max} (\Diamond \text{goal})$$
Transformation-based approach for MDP

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

$$\Pr_{\mathcal{M}', s_{\text{init}}}^{\text{max}}(\lozenge \text{goal} | \lozenge (\text{goal} \lor \text{stop})) = \Pr_{\mathcal{M}'', s_{\text{init}}}^{\text{max}}(\lozenge \text{goal})$$

idea: $\mathcal{M}''$ redistributes the probabilities of the paths $\pi$ with $\pi \not\models \lozenge (\text{goal} \lor \text{stop})$
Transformation-based approach for MDP

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

$$\Pr_{\mathcal{M}',s_{\text{init}}}^\max (\Diamond \text{goal} \mid \Diamond (\text{goal} \lor \text{stop})) = \Pr_{\mathcal{M}'',s_{\text{init}}}^\max (\Diamond \text{goal})$$
Transformation-based approach for MDP

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

$\Pr_{\mathcal{M}',S_{\text{init}}}^{\text{max}}(\Diamond \text{goal} | \Diamond (\text{goal} \lor \text{stop})) = \Pr_{\mathcal{M}'',S_{\text{init}}}^{\text{max}}(\Diamond \text{goal})$
Transformation-based approach for MDP

step 2: normal form MDP \( \mathcal{M}' \) \( \sim \) MDP \( \mathcal{M}'' \) s.t.

\[
\Pr_{\mathcal{M}', S_{\text{init}}}^{\max} \left( \Diamond \text{goal} \mid \Diamond (\text{goal} \lor \text{stop}) \right) = \Pr_{\mathcal{M}'', S_{\text{init}}}^{\max} \left( \Diamond \text{goal} \right)
\]
Transformation-based approach for MDP

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

$$\Pr_{\mathcal{M}',s_{\text{init}}}^{\text{max}}(\Diamond \text{goal} \mid \Diamond(\text{goal} \lor \text{stop})) = \Pr_{\mathcal{M}'',s_{\text{init}}}^{\text{max}}(\Diamond \text{goal})$$

How to deal with states that might never reach one of the trap states?
Transformation-based approach for MDP

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

$$\Pr_{\mathcal{M}',s_{\text{init}}}^{\text{max}}\left(\Diamond \text{goal} \mid \Diamond(\text{goal} \lor \text{stop})\right) = \Pr_{\mathcal{M}'',s_{\text{init}}}^{\text{max}}\left(\Diamond \text{goal}\right)$$

add reset-transitions from all end components that do not contain a trap state.
Summary: conditional probabilities for MDP

for Markov decision processes:

$$\Pr_{\mathcal{M},s}^{\max}(\varphi | \psi) = \max_{\sigma} \frac{\Pr^\sigma_s(\varphi \land \psi)}{\Pr^\sigma_s(\psi)}$$

computation by reduction to unconditional probabilities

* reset-mechanism for reachability objective and condition
* generalization for LTL objectives/conditions via $\omega$-automata
Summary: conditional probabilities for MDP

for Markov decision processes:

$$\Pr_{\mathcal{M},s}^{\max}(\varphi | \psi) = \max_{\sigma} \frac{\Pr_{s}^{\sigma}(\varphi \land \psi)}{\Pr_{s}^{\sigma}(\psi)}$$

computation by reduction to unconditional probabilities

* reset-mechanism for reachability objective and condition
* generalization for LTL objectives/conditions via $\omega$-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
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
Quantiles

well-known in statistics:

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

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

is the $q$-quantile of $f$.

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

well-known in statistics:

\[
\inf \left\{ r \in \mathbb{R} : \Pr\{f \leq r\} > q \right\}
\]

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:

\[ \Pr_s(\diamond \leq e \Rightarrow u \text{ goal}) \]

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:

\[ \Pr_s(\Diamond \leq e \; \text{goal}) \]

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:

$$\min \{ e \in \mathbb{N} : \Pr_s(\Diamond \leq^e_u \text{goal}) > 0.8 \}$$

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 $u$

---

Diagram:

- Probability vs. Energy Consumption
- $e_{\text{min}}$ at 80% probability
Examples for quantiles in Markov chains

energy-aware job scheduling:

$$\min \{ e \in \mathbb{N} : \Pr_s(\diamond \leq_e u \text{ goal}) > 0.8 \}$$

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 $u$

for fixed energy budget $e$

80%
Examples for quantiles in Markov chains

energy-aware job scheduling:

\[
\min \left\{ e \in \mathbb{N} : \Pr_s(\diamondsuit \gtrless^e_u \text{goal}) > 0.8 \right\}
\]

\[
\max \left\{ u \in \mathbb{N} : \Pr_s(\diamondsuit \gtrless^e_u \text{goal}) > 0.8 \right\}
\]

for fixed utility value \( u \)

for fixed energy budget \( e \)

\[\text{utility value} \quad u_{\text{max}}\]

\[\text{energy consumption} \quad e_{\text{min}}\]
Quantiles in Markovian models

Markov chains:

\[
\begin{align*}
\min & \{ \ r \in \mathbb{N} : \ Pr_s(\diamond \leq_r \text{goal}) > 0.8 \} \\
\max & \{ \ r \in \mathbb{N} : \ Pr_s(\diamond \geq_r \text{goal}) > 0.8 \}
\end{align*}
\]

Markov decision processes:

\[
\begin{align*}
\min & \{ \ r \in \mathbb{N} : \ Pr_s^{\max}(\diamond \leq_r \text{goal}) > 0.8 \} \\
\max & \{ \ r \in \mathbb{N} : \ Pr_s^{\max}(\diamond \geq_r \text{goal}) > 0.8 \}
\end{align*}
\]
Quantiles in Markovian models

Markov chains:

\[
\begin{align*}
\min & \left\{ r \in \mathbb{N} : \Pr_s(\diamond \leq r \text{ goal}) > 0.8 \right\} \\
\max & \left\{ r \in \mathbb{N} : \Pr_s(\diamond \geq r \text{ goal}) > 0.8 \right\}
\end{align*}
\]

Markov decision processes:

\[
\begin{align*}
\min & \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\diamond \leq r \text{ goal}) > 0.8 \right\} \\
\min & \left\{ r \in \mathbb{N} : \Pr_s^{\min}(\diamond \leq r \text{ goal}) > 0.8 \right\} \\
\max & \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\diamond \geq r \text{ goal}) > 0.8 \right\} \\
\max & \left\{ r \in \mathbb{N} : \Pr_s^{\min}(\diamond \geq r \text{ goal}) > 0.8 \right\}
\end{align*}
\]
Computing quantiles in MDP
Computing quantiles in MDP

e.g., existential quantiles

\[
\min \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond \preceq r G) > q \right\} \\
\max \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\Diamond \succeq r G) > q \right\}
\]

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

$$\min \left\{ r \in \mathbb{N} : \Pr_s^{\max} (\mathcal{G} \leq r) = 1 \right\}$$

$$\max \left\{ r \in \mathbb{N} : \Pr_s^{\max} (\mathcal{G} \geq r) > 0 \right\}$$

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

\[
\begin{align*}
\min & \{ \ r \in \mathbb{N} : \ Pr_s^{\max}(\diamond \leq r G) > q \} \\
\max & \{ \ r \in \mathbb{N} : \ Pr_s^{\max}(\diamond \geq r G) > q \}
\end{align*}
\]

results on the computation of quantiles:

- qualitative quantiles in poly-time \cite{UmmelsBaier13}
- EXP-compl. for quantitative quantiles \cite{HaaseKiefer15}
- iterative LP-approach for quantitative quantiles
Computing quantiles in MDP

e.g., existential quantiles

\[
\min \{ r \in \mathbb{N} : \Pr_s^{\max}(\diamond \leq r G) > q \} \\
\max \{ r \in \mathbb{N} : \Pr_s^{\max}(\diamond \geq r G) > q \}
\]

results on the computation of quantiles:

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

\[ \text{qu}(s_0) = \min \left\{ r \in \mathbb{N} : \Pr_{s_0}^{\text{max}}(\diamond \leq r G) > q \right\} \]

existential quantile for

- upper reward-bounded reachability
- lower probability bound

\[ \Pr_s^{\text{max}}(\varphi) > p \quad \text{iff} \quad \left\{ \begin{array}{l}
\text{there exists a scheduler } \sigma \\
\text{with } \Pr_s^{\sigma}(\varphi) > p
\end{array} \right. \]
Computing quantitative quantiles

\[ \text{qu}(s_0) = \min \{ r \in \mathbb{N} : \Pr_{s_0}^{\text{max}}(\diamond \leq r G) > q \} \]

1. compute \( p = \Pr_{s_0}^{\text{max}}(\diamond G) \)
2. return \( \text{qu}(s_0) = \infty \) if \( p \leq q \)
3. ...
Computing quantitative quantiles

\[
\text{qu}(s_0) = \min \left\{ r \in \mathbb{N} : \underbrace{\Pr_{s_0}^{\max}(\diamond \leq r \ G)}_{p_{s_0,r}} > q \right\}
\]

1. compute \( p = \Pr_{s_0}^{\max}(\diamond G) \)
2. return \( \text{qu}(s_0) = \infty \) if \( p \leq q \)
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

\[ \text{qu}(s_0) = \min \{ r \in \mathbb{N} : \Pr_{s_0}^{\text{max}}(\Diamond^r G) > q \} \]

1. compute \( p = \Pr_{s_0}^{\text{max}}(\Diamond G) \)
2. return \( \text{qu}(s_0) = \infty \) if \( p \leq q \)
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

\[ \text{qu}(s_0) = \min \{ r \in \mathbb{N} : \Pr_{s_0}^{\max}(\diamond \leq r G) > q \} \]

1. compute \( p = \Pr_{s_0}^{\max}(\diamond G) \)
2. return \( \text{qu}(s_0) = \infty \) if \( p \leq q \)
3. for \( r = 0, 1, 2, \ldots \) compute the values \( p_{s_0, r} \)
   for all states \( s \in S \) and return the smallest value \( r \) such that \( p_{s_0, r} > q \)

computation of \( p_{s_0, r} \) by an iterative linear-programming approach with back propagation
linear program for the values $p_{s,r} = \Pr_s^{\max}(\Diamond \leq_r G)$

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

$x_{s,r} = 1$ if $s \models G$

If $s \not\in G$, $s \models \exists \Diamond G$ and $\alpha \in \text{Act}(s)$ then:

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

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

solution: $x_{s,r} = p_{s,r} = \Pr_s^{\max}(\Diamond \leq_r G)$
linear program for the values \( p_{s,r} = \Pr_s^{\text{max}}(\diamond \leq r G) \)

\[
x_{s,r} = 0 \quad \text{if } s \not\models \exists \diamond G
\]

\[
x_{s,r} = 1 \quad \text{if } s \models G
\]

If \( s \not\in G, s \models \exists \diamond G \) and \( \alpha \in \text{Act}(s) \) then:

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

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

unique solution: \( x_{s,r} = p_{s,r} = \Pr_s^{\text{max}}(\diamond \leq r G) \)
linear program for the values $p_{s,r} = \Pr_{s}^{\text{max}}(\diamond \preceq r G)$

\[
x_{s,r} = \begin{cases} 
0 & \text{if } s \not\models \exists \diamond G \\
1 & \text{if } s \models \exists \diamond G
\end{cases}
\]

If $s \not\in G$, $s \models \exists \diamond G$ and $\alpha \in \text{Act}(s)$ then:

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

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

use the solutions $p_{t,i} = \Pr_{s}^{\text{max}}(\diamond \preceq i G)$ for $i < r$

computed in previous iterations
linear program for the values $p_{s,r} = \Pr_{s}^{\max}(\lozenge \leq r G)$

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

$$x_{s,r} = 1 \quad \text{if} \quad s \models \exists \lozenge G$$

If $s \not\in G$, $s \not\models \exists \lozenge G$ and $\alpha \in Act(s)$ then:

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

$$x_{s,r} \geq \text{const}$$

use the solutions $p_{t,i} = \Pr_{s}^{\max}(\lozenge \leq i G)$ for $i < r$

computed in previous iterations
linear program for the values $p_{s,r} = \Pr_s^{\text{max}}(\Diamond r G)$

$$
\begin{align*}
x_{s,r} &= 0 \quad \text{if } s \not\models \exists \Diamond G \\
x_{s,r} &= 1 \quad \text{if } s \models \exists \Diamond G
\end{align*}
$$

minimize $\sum_s x_{s,r}$

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

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

$$
x_{s,r} \geq \text{const}
$$

linear in the size of the MDP

linear program to be solved in the $r$-th iteration
Expectation quantiles
Expectation quantiles: example

\[ \mathcal{M} = (S, \text{Act}, P, \text{energy}, \text{utility}, s_0) \]

MDP with two reward functions

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

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

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

(under some scheduler)
Expectation quantiles: example

\[ M = (S, Act, P, \text{energy}, \text{utility}, s_0) \]

MDP with two reward functions

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

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

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)
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
Mean-payoff
Mean-payoff

given: a weighted graph without trap states

mean-payoff functions $\overline{\text{MP}}, \underline{\text{MP}} : \text{InfPaths} \rightarrow \mathbb{R}$:

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

\[
\underline{\text{MP}}(s_0 s_1 s_2 \ldots) = \liminf_{n \to \infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} \text{wgt}(s_i)
\]
Mean-payoff

given: a weighted graph without trap states

mean-payoff functions $\overline{\text{MP}}, \underline{\text{MP}} : \text{InfPaths} \rightarrow \mathbb{R}$:

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

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

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

\[
\underline{\text{MP}}(\pi) < 0 < \overline{\text{MP}}(\pi)
\]
Expected mean-payoff in finite MC or MDP

fundamental results:

in finite MC: \( E_s(MP) = E_s(\overline{MP}) \)

in finite MDP:

\( E_s^{\text{max}}(MP) = E_s^{\text{max}}(\overline{MP}) \)
\( E_s^{\text{min}}(MP) = E_s^{\text{min}}(\overline{MP}) \)

and optimal MD-scheduler exist

notation: \( E^*_s(MP) \) rather than \( E^*_s(MP) \) resp. \( E^*_s(\overline{MP}) \)
Expected mean-payoff in finite MC

fundamental results:

in finite MC: \( \mathbb{E}_s(\text{MP}) = \mathbb{E}_s(\overline{\text{MP}}) \)

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: \( E_s(MP) = E_s(\overline{MP}) \)

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:

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

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

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

long-run frequency of state $s$ 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:

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

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

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

long-run frequency of state $s$ in path $\pi$

$$\text{freq}(s, s_0 \ s_1 \ldots \ s_n) = \begin{cases} \text{number of occurrences of } s \text{ in the sequence } s_0 \ s_1 \ldots \ s_n \end{cases}$$
Mean-payoff in finite weighted MC

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

$$\theta^B(s) = \lim_{n \to \infty} \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr_t(\bigcirc^i s)$$

for each $t \in B$

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

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

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

$$\text{MP}(\pi) = \sum_{s \in B} \theta^B(s) \cdot \text{wgt}(s)$$
Mean-payoff in finite weighted MC

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

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

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

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

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

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

only depends on $B$
Mean-payoff in finite weighted MC

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

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

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

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

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

$$\text{MP}(\pi) = \sum_{s \in B} \theta^B(s) \cdot wgt(s) \overset{\text{def}}{=} \text{MP}(B)$$

only depends on $B$
Mean-payoff in finite weighted MC

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

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

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

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

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

$$\text{MP}(\pi) = \sum_{s \in B} \theta^B(s) \cdot \text{wgt}(s) \overset{\text{def}}{=} \text{MP}(B)$$

expected mean-payoff:

$$\sum_{B} \Pr_{s_0}(\diamond B) \cdot \text{MP}(B)$$
random variable $\overline{\text{MP}} : \text{InfPaths} \rightarrow \mathbb{R}$ defined by

$$\overline{\text{MP}}(s_0 \ s_1 \ s_2 \ldots) = \limsup_{n \to \infty} \frac{1}{n+1} \cdot \sum_{i=0}^{n} \text{wgt}(s_i)$$
Mean-payoff in MDPs

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

Random variable $\overline{\text{MP}} : \text{InfPaths} \rightarrow \mathbb{R}$ defined by

$$\overline{\text{MP}}(s_0 s_1 s_2 \ldots) = \lim_{n \to \infty} \sup \left\{ \frac{1}{n+1} \cdot \sum_{i=0}^{n} \textit{wgt}(s_i) \right\}$$
Mean-payoff in MDPs

Given MDP with weight function \( wgt : S \rightarrow \mathbb{Q} \), 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]
Given MDP with weight function $\text{wgt} : S \rightarrow \mathcal{Q}$, find a scheduler maximizing the expected mean-payoff.

**Results:**

- 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

[Howard’60], [Dernan’66], [Kallenberg’80] ...
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 $(s, \alpha)$ in MR-schedulers
linear program for the maximal expected mean-payoff:

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

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

- if $x_s \overset{\text{def}}{=} \sum_{\alpha \in \text{Act}(s)} x_{s,\alpha} > 0$ 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 $(s, \alpha)$ in MR-schedulers

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

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

- if $x_s = 0$ then $\sigma$ behaves an MD-scheduler that reaches a state $t$ with $x_t = 1$ with probability 1
linear program for the maximal expected mean-payoff:

\[
\text{maximize } \sum_{s,\alpha} x_{s,\alpha} \cdot \text{wgt}(s, \alpha)
\]

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:

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

\[
\text{mean-payoff of MR-scheduler } \sigma \text{ given by }
\]

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

for each state \( s \) with \( x_s \overset{\text{def}}{=} \sum_{\alpha \in \text{Act}(s)} x_{s, \alpha} > 0 \)
linear program for the maximal expected mean-payoff:

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

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

balance equation for state \( t \)

\[
x_t = \sum_{\beta \in \text{Act}(t)} x_{t,\beta}
\]
Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

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

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

\[
x_{s,\alpha} \geq 0 \quad \text{for } s \in S \text{ and } \alpha \in \text{Act}(s)
\]

long-run frequencies are non-negative

\[
x_t = \sum_{\beta \in \text{Act}(t)} x_{t,\beta}
\]
Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

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

\[
\begin{align*}
x_t &= \sum_{s,\alpha} x_{s,\alpha} \cdot P(s, \alpha, t) \quad \text{for } t \in S \\
x_{s,\alpha} &\geq 0 \quad \text{for } s \in S \text{ and } \alpha \in \text{Act}(s) \\
\sum_{s,\alpha} x_{s,\alpha} &= 1 \quad \text{for } t \in S
\end{align*}
\]

long-run frequencies yield a distribution

\( x_t = \sum_{\beta \in \text{Act}(t)} x_{t,\beta} \)
linear program for the maximal expected mean-payoff:

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

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

\[
x_{s,\alpha} \geq 0 \quad \text{for } s \in S \text{ and } \alpha \in \text{Act}(s)
\]

\[
\sum_{s,\alpha} x_{s,\alpha} = 1
\]

Each solution induces an optimal MR-scheduler.
Mean-payoff in strongly connected MDPs

linear program for the maximal expected mean-payoff:

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

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

\[
x_{s,\alpha} \geq 0 \quad \text{for } s \in S \text{ and } \alpha \in \text{Act}(s)
\]

\[
\sum_{s,\alpha} x_{s,\alpha} = 1
\]

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:

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

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

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

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

optimal MD-scheduler: for each state $s$ with $x_s > 0$

pick an action $\alpha$ 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 $\text{Act}(s) = \emptyset$. 
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: 

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

$\mathbf{x}_{s,\alpha}$ long-run frequency

$\mathbf{y}_{s,\alpha}$ frequency in the transient part

State $s$ is called a trap if $\text{Act}(s) = \emptyset$. 

[K ALLENBERG’80]
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

$\mathbf{x}_{s,\alpha}$ long-run frequency

$\mathbf{y}_{s,\alpha}$ frequency in the transient part

method 2:

compute the maximal expected mean-payoff of the MECs and “compose” the result for the full MDP
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 $\mathcal{E}_1, \ldots, \mathcal{E}_k$

step 2: for $i = 1, \ldots, k$, 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 $\mathcal{E}_1, \ldots, \mathcal{E}_k$

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 $\mathcal{M}'$
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 $\mathcal{M}'$

$\mathcal{M}'$ arises from $\text{MEC}(\mathcal{M})$ by adding
- a fresh trap state $\text{goal}$
- a new action symbol $\tau$
- transitions $\mathcal{E}_i \xrightarrow{\tau} \text{goal}$ for $i, \ldots, k$
Mean-payoff in MDPs: general case

step 1: compute the maximal end components $E_1, \ldots, E_k$

step 2: for $i = 1, \ldots, k$, compute the maximal expected mean-payoff $mp_i$ of $E_i$

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

step 4: compute the maximal expected total weight

\[
\Pr_{\mathcal{M}'}^{\min}(\Diamond \text{goal}) = 1
\]

maximal expected total weight and optimal MD-scheduler exist
Mean-payoff in MDPs: general case

step 1: compute the maximal end components $E_1, ..., E_k$

step 2: for $i = 1, ..., k$, compute the maximal expected mean-payoff $mp_i$ of $E_i$

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

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

$$\mathbb{E}_{\mathcal{M}'}^{\max}(\text{“total weight”}) = \mathbb{E}_{\mathcal{M}}^{\max}(\text{MP})$$
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 $\mathcal{E}_i \xrightarrow{\tau} \text{goal}$ 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 $\mathcal{E}_1, \ldots, \mathcal{E}_k$

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

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

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

optimal MD-scheduler arises by combining $\nu, \sigma_1, \ldots, \sigma_k$
Expected long-run ratios

\[ \text{ratio} = \frac{\text{cost}}{\text{util}} \] where \( \text{cost}, \text{util} \) 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

\[ \sum_B \Pr_s(\Diamond B) \cdot \frac{\text{MP}[\text{cost}](B)}{\text{MP}[\text{util}](B)} \]

where \( B \) ranges over all BSCCs of the MC

\[ \text{ratio} = \frac{\text{cost}}{\text{util}} \]

where \( \text{cost} \), \( \text{util} \) 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

• LP-based approach

\[
\text{ratio} = \frac{\text{cost}}{\text{util}} \quad \text{where cost, util are reward functions}
\]

[Gimbert’07]
[de Alfaro’98]
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 \[\text{[Gimbert'07]}\]
- LP-based approach \[\text{[de Alfaro'98]}\]

minimize \(y\) subject to

\[
x_s \geq \text{cost}(s, \alpha) - y \cdot \text{util}(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t
\]

for all states \(s\) and \(\alpha \in \text{Act}(s)\)
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 $\mathcal{M}$

$\diamondsuit \text{goal}$ accumulated weight until reaching a goal state

requirement for $\mathcal{M}$:

$\Pr_{\text{min}}(\diamondsuit \text{goal}) = 1$

best- or worst-case expectation

$\mathbb{E}_{\text{min}}(\diamondsuit \text{goal})$ or $\mathbb{E}_{\text{max}}(\diamondsuit \text{goal})$

extrema over all schedulers
Stochastic shortest/longest path problem

weighted MDP $\mathcal{M}$

Goal
accumulated weight until reaching a goal state

relaxed requirement: $\Pr_{\text{max}}(\diamond \text{goal}) = 1$

best- or worst-case expectation $\mathbb{E}_{\text{min}}(\diamond \text{goal})$ or $\mathbb{E}_{\text{max}}(\diamond \text{goal})$

extrema over all proper schedulers

Bertsekas/Tsitsiklis’91
de Alfaro’99
Maximal expected accumulated weight

given: MDP \( \mathcal{M} = (S, Act, P, wgt) \) and \( G \subseteq S \) s.t.
\[
T = \{ s \in S : \Pr_{s}^{\max}(\diamondsuit G) = 1 \} \neq \emptyset
\]

task: compute \( x_{s} = E_{s}^{\max}(\triangledown G) \) for \( s \in T \)

maximum over all proper schedulers

\( \sigma \) is proper iff \( \Pr_{s}^{\sigma}(\diamondsuit G) = 1 \) for all \( s \in T \)
Maximal expected accumulated weight

given: MDP $\mathcal{M} = (S, \text{Act}, P, \text{wgt})$ and $G \subseteq S$ s.t.
$T = \{s \in S : \Pr_s^{\max}(\lozenge G) = 1\} \neq \emptyset$

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

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

$\sigma$ is proper iff $\Pr_s^\sigma(\lozenge G) = 1$ for all $s \in T$
Maximal expected accumulated weight

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

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

task: compute $x_s = \mathbb{E}_s^{\max}(\Diamond G)$ 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 $(s, \alpha)$ where $s \in T \setminus G$, $\alpha \in \text{Act}(s)$ 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 $\mathcal{M} = (S, \text{Act}, P, \text{wgt})$ and $G \subseteq S$ s.t. $T = \{ s \in S : \Pr_s^{\text{max}}(\Diamond G) = 1 \} \neq \emptyset$

task: compute $x_s = \mathbb{E}_s^{\text{max}}(\Diamond G)$ 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 $(s, \alpha)$ where $s \in T \setminus G$, $\alpha \in \text{Act}(s)$ and

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

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

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

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

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

$\mathbb{E}_s^{max}(\Diamond goal)$ 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 accumulated weight:

\[ \mathbb{E}_s^{\max} (\Diamond \text{goal}) = +\infty \]

note that \( \mathbb{E}_s^{\sigma_n} (\Diamond \text{goal}) = n \) where \( \sigma_n \) schedules

- \( \beta \) for the first \( n \) visits of \( s \)
- \( \alpha \) 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. $\Pr_s^{\max}(\diamond G) = 1$ for all $s \in S$

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

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

$$x_s < +\infty \text{ for all } s \in S$$
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 $x_s = E^s_{\max}(\Diamond G)$ for $s \in S$

If $E^\sigma_s(“total weight”) = -\infty$ for each improper scheduler $\sigma$ then:

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

[Bertsekas/Tsitsiklis’91]
Maximal expected accumulated weight

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

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

If \( \mathbb{E}_s^{\sigma} ("\text{total weight}") = -\infty \) for each improper scheduler \( \sigma \) then:

[\text{Bertsekas/Tsitsiklis'91}]

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

\[
x_s = \max_{\alpha \in \text{Act}(s)} \left( \text{wgt}(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \right)
\]
Maximal expected accumulated weight

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

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

If $\mathbb{E}^\sigma_s (\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

[Bertsekas/Tsitsiklis’91]

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

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

... unique fixpoint
Maximal expected accumulated weight

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

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

If $\mathbb{E}^\sigma_s ("\text{total weight}") = -\infty$ for each improper scheduler $\sigma$ then:

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

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

... unique fixpoint, optimal MD-scheduler exist ...
Maximal expected accumulated weight

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

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

If $\mathbb{E}_s^{\sigma}(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

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

$$x_s \geq \max_{\alpha \in \text{Act}(s)} \left( wgt(s, \alpha) + \sum_{t \in S} P(s, \alpha, t) \cdot x_t \right)$$

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. $\Pr^\text{max}_s(\Diamond G) = 1$ for all $s \in S$

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

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

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

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

value iteration (arbitrary starting vector)
Maximal expected accumulated weight

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

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

If $\mathcal{E}_s^{\sigma}(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

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

[Bertsekas/Tsitsiklis’91]

How to compute $x_s$ if $\mathcal{E}_s^{\sigma}(\text{“total weight”}) > -\infty$ for some improper scheduler $\sigma$?
Maximal expected accumulated weight

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

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

If $E^\sigma_s(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ 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 $E^\sigma_s(\text{“total weight”}) > -\infty$ for some improper scheduler $\sigma$? How to check finiteness?
Non-negative weights

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

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

consider the case of non-negative weights, i.e., $wgt(s, \alpha) \geq 0$ for all state-action pairs
Non-negative weights

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

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

results:

- $\mathbb{E}_s^{\max}(\Diamond G) = \infty$ iff $s$ can reach a positive EC

[De Alfaro’99]

end component that contains a state-action pair with positive weight
Non-negative weights

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

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

results:

- $\mathbb{E}_{s}^\text{max}(\diamond G) = \infty$ iff $s$ can reach a positive EC

- if $\mathcal{M}$ has no positive ECs and $\mathcal{N} = \text{MEC}(\mathcal{M})$ then:
  \[ \mathbb{E}_{\mathcal{M},s}^\text{max}(\diamond G) = \mathbb{E}_{\mathcal{N},s}^\text{max}(\diamond G) \]

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. $\Pr_s^{\text{max}}(\Diamond G) = 1$ for all $s \in S$

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

results: [De Alfaro’99]

- $\mathbb{E}_s^{\text{max}}(\Diamond G) = \infty$ iff $s$ can reach a positive EC
- if $\mathcal{M}$ has no positive ECs and $\mathcal{N} = \text{MEC}(\mathcal{M})$ then:
  $$\mathbb{E}_{\mathcal{M}, s}^{\text{max}}(\Diamond G) = \mathbb{E}_{\mathcal{N}, s}^{\text{max}}(\Diamond G)$$

Hence: $\mathbb{E}_{\mathcal{M}, s}^{\text{max}}(\Diamond G)$ is computable in polynomial time.
Non-positive weights

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

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

results: [De Alfaro’99]

- $\mathbb{E}^\text{max}_s(\Diamond G)$ is finite ... and non-positive
Non-positive weights

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

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

results: [De Alfaro’99]

- $\mathbb{E}_s^{\text{max}}(\Diamond G)$ 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 $(s, \alpha)$ in $\mathcal{E}$
Non-positive weights

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

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

results: \cite{DeAlfaro'99}

- $\mathbb{E}_s^{\max}(\diamond G)$ is finite ... and non-positive
- if $\mathcal{N}$ is the MDP arising from $\mathcal{M}$ by collapsing all zero-ECs then $\mathbb{E}_{\mathcal{M},s}^{\max}(\diamond G) = \mathbb{E}_{\mathcal{N},s}^{\max}(\diamond G)$

end component $\mathcal{E}$ with $wgt(s, \alpha) = 0$ for all state-action pairs $(s, \alpha)$ in $\mathcal{E}$
Non-positive weights

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

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

results: [De Alfaro’99]

- \( \mathbb{E}^\text{max}_s(\Diamond G) \) is finite ... and non-positive

- if \( \mathcal{N} \) is the MDP arising from \( \mathcal{M} \) by collapsing all zero-ECs then 
  \[ \mathbb{E}^\text{max}_{\mathcal{M},s}(\Diamond G) = \mathbb{E}^\text{max}_{\mathcal{N},s}(\Diamond G) \]

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.
$Pr_s^{\text{max}}(\Diamond G) = 1$ for all $s \in S$

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

results: [De Alfaro’99]

- $\mathbb{E}^\text{max}_s(\Diamond G)$ is finite ... and non-positive
- if $\mathcal{N}$ is the MDP arising from $\mathcal{M}$ by collapsing all zero-ECs then $\mathbb{E}^\text{max}_{\mathcal{M},s}(\Diamond G) = \mathbb{E}^\text{max}_{\mathcal{N},s}(\Diamond G)$
- $\mathbb{E}^\text{max}_{\mathcal{N},s}(\Diamond G)$ computable via Bellman equations

... expected total weight of each improper scheduler is $-\infty$
Maximal expected accumulated weight

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

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

If $\mathbb{E}^\sigma_s(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

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

Treatment of non-negative or non-positive weights: $\sqrt{\checkmark}$

General case: $???
Maximal expected accumulated weight

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

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

If $\mathbb{E}^\sigma_s(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

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

[Bertsekas/Tsitsiklis’91]

Treatment of non-negative or non-positive weights: √

General case: … consider the MECs separately …
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathcal{E}_{\mathcal{E}}^{\max}(\bigotimes G) = \infty$$

iff ...
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathcal{E}_E^{\max}(\otimes G) = \infty$$

iff $\mathcal{E}$ is weight-divergent
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}^\text{max}_\mathcal{E}(\Diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr^{\text{max}}_{\mathcal{E},s}(\Diamond (\text{wgt} \geq r)) = 1 \right\} = \infty$$
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$E_{\mathcal{E}}^{\max}(\Diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr_{\mathcal{E},s}^{\max}(\Diamond(wgt \geq r)) = 1 \right\} = \infty$$

iff $\Pr_{\mathcal{E}}^{\max}\left\{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \right\} = 1$

$\text{pref}(\pi, n) = \text{prefix of } \pi \text{ of length } n$
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$E_{\mathcal{E}}^{\max}(\Diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr_{\mathcal{E},s}^{\max}(\Diamond(wgt \geq r)) = 1 \} = \infty$$

iff $\Pr_{\mathcal{E}}^{\max}\{ \pi : \limsup_{n \to \infty} wgt(pref(\pi, n)) = \infty \} = 1$

iff $E_{\mathcal{E}}^{\max}(MP) > 0$ or ...

$pref(\pi, n) = \text{prefix of } \pi \text{ of length } n$
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\max_{\mathcal{E}} (\Diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr_{\mathcal{E},s} (\Diamond (\text{wgt} \geq r)) = 1 \} = \infty$$

iff $\Pr_{\mathcal{E}} \{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \} = 1$

iff $\max_{\mathcal{E}} (\text{MP}) > 0$ or $\max_{\mathcal{E}} (\text{MP}) = 0$ & $\mathcal{E}$ is gambling

$\text{pref}(\pi, n) = \text{prefix of } \pi \text{ of length } n$
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$\mathbb{E}_{\mathcal{E}}^{\max}(\Diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \{ r \in \mathbb{N} : \Pr_{\mathcal{E},s}^{\max}(\Diamond (\text{wgt} \geq r)) = 1 \} = \infty$$

iff $\Pr_{\mathcal{E}}^{\max}\{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \} = 1$

iff $\mathbb{E}_{\mathcal{E}}^{\max}(\text{MP}) > 0$ or $\mathbb{E}_{\mathcal{E}}^{\max}(\text{MP}) = 0$ & $\mathcal{E}$ is gambling

there exists scheduler s.t. almost surely:

$$\limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = +\infty$$

$$\liminf_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = -\infty$$
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$E_{\mathcal{E}}^\text{max}(\diamondsuit G) = \infty$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$\sup \left\{ r \in \mathbb{N} : \Pr_{s}^\text{max}(\diamondsuit (\text{wgt} \geq r)) = 1 \right\} = \infty$

iff $\Pr_{\mathcal{E}}^\text{max} \left\{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \right\} = 1$

iff $E_{\mathcal{E}}^\text{max}(\text{MP}) > 0$ or $E_{\mathcal{E}}^\text{max}(\text{MP}) = 0$ & $\mathcal{E}$ is gambling

can be checked in polynomial time

exists scheduler s.t. almost surely:

$\limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = +\infty$

$\liminf_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = -\infty$
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$E^\text{max}_\mathcal{E}(\Diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr^\text{max}_s(\Diamond (\text{wgt} \geq r)) = 1 \right\} = \infty$$

iff $\Pr^\text{max}_\mathcal{E} \left\{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \right\} = 1$

iff $E^\text{max}_\mathcal{E}(\text{MP}) > 0$ or $E^\text{max}_\mathcal{E}(\text{MP}) = 0$ & $\mathcal{E}$ is gambling

how to check whether an EC is gambling?

can be checked in polynomial time
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$
\mathcal{E}^{\max}(\diamond G) = \infty
$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$
\sup \left\{ r \in \mathbb{N} : \Pr_s^{\max}(\diamond (\text{wgt} \geq r)) = 1 \right\} = \infty
$$

iff $\Pr_{\mathcal{E}}^{\max}\left\{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \right\} = 1$

iff $\mathcal{E}^{\max}(\text{MP}) > 0$ or $\mathcal{E}^{\max}(\text{MP}) = 0$ & $\mathcal{E}$ is gambling

The problem to check whether a given EC is gambling is NP-hard
Let $\mathcal{E}$ be an end component of $\mathcal{M}$.

$$E^\text{max}_\mathcal{E}(\diamond G) = \infty$$

iff $\mathcal{E}$ is weight-divergent, i.e., for all states $s$ in $\mathcal{E}$:

$$\sup \left\{ r \in \mathbb{N} : \Pr^\text{max}_s(\diamond (\text{wgt} \geq r)) = 1 \right\} = \infty$$

iff $\Pr^\text{max}_\mathcal{E}\left\{ \pi : \limsup_{n \to \infty} \text{wgt}(\text{pref}(\pi, n)) = \infty \right\} = 1$

iff $E^\text{max}_\mathcal{E}(\text{MP}) > 0$ or $E^\text{max}_\mathcal{E}(\text{MP}) = 0$ & $\mathcal{E}$ is gambling

The problem to check whether a given EC is gambling

- is NP-hard
- solvable in polynomial-time if $E^\text{max}_\mathcal{E}(\text{MP}) = 0$
Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $\mathbb{E}_{\mathcal{E}}^\text{max}(\text{MP}) = 0$. 
Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $E^\text{max}_\mathcal{E}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E^\sigma_{\mathcal{E},s}(\text{MP}) = 0$ for $s \in \mathcal{E}$.
Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $E_{\mathcal{E}}^{\max}(MP) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E_{\mathcal{E},s}^\sigma(MP) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$.

$\mathcal{E}'$ 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 $\mathbb{E}_\mathcal{E}^{\max}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $\mathbb{E}_{\mathcal{E},s}^{\sigma}(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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 $\mathbf{E}_{\mathcal{E}}^{\text{max}}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $\mathbf{E}_{\mathcal{E},s}^\sigma(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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 $E_{\mathcal{E}}^{\max}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E_{\mathcal{E},s}^\sigma(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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 $E_{\mathcal{E}}^{\max}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E_{\mathcal{E},s}^{\sigma}(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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.

![Diagram](attachment:image.png)

$\text{wgt}(s, \alpha) = +1$

$\text{wgt}(t, \beta) = -1$
Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $E_{\mathcal{E}}^{\text{max}}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E_{\mathcal{E},s}^{\sigma}(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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 $E_{\mathcal{E}}^{\text{max}}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E_{\mathcal{E},s}^{\sigma}(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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:

$$w(s, t) = wgt(\pi)$$ for all paths $\pi$ from $s$ to $t$
Non-gambling EC with zero mean-payoff

Let $\mathcal{E}$ be an end component of $\mathcal{M}$ with $E^\text{max}_\mathcal{E}(\text{MP}) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E^\sigma_{\mathcal{E},s}(\text{MP}) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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:

$$w(s, t) = \text{wgt}(\pi)$$

for all paths $\pi$ from $s$ 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 $E^\max_\mathcal{E}(MP) = 0$. Pick an MD-scheduler $\sigma$ s.t. $E^\sigma_{\mathcal{E},s}(MP) = 0$ for $s \in \mathcal{E}$ and a BSCC $\mathcal{E}'$ of $\sigma$. 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:

$$w(s, t) = wgt(\pi)$$

for all paths $\pi$ from $s$ 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

![Diagram showing the construction process](image-url)
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: $\operatorname{Act}(s) \cap \operatorname{Act}(t) = \emptyset$ 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 $M$ and a zero-EC $E$ of $M$

1. pick a state $s$ in $E$

2. remove all state-action pairs in $E$
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 $\text{wgt}(t, \tau) = -w(s, t)$
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 $\mathcal{E}$

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 $\text{wgt}(t, \tau) = -w(s, t)$

4. replace each state-action pair $(t, \beta)$ in $\mathcal{M} \setminus \mathcal{E}$
   
   where $t \neq s$ with the pair $(s, \beta)$

$$s \xrightarrow{\text{in } \mathcal{E}} t \xrightarrow{\beta} \ldots$$
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 \( \text{wgt}(t, \tau) = -w(s, t) \)

4. replace each state-action pair \( (t, \beta) \) in \( \mathcal{M} \setminus \mathcal{E} \)
   where \( t \neq s \) with the pair \( (s, \beta) \):
   \[
   \text{wgt}(s, \beta) = w(s, t) + \text{wgt}(t, \beta)
   \]

\[ s \xrightarrow{\text{in } \mathcal{E}} t \xrightarrow{\beta} \ldots \]
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 $\text{wgt}(t, \tau) = -w(s, t)$
4. replace each state-action pair $(t, \beta)$ in $\mathcal{M} \setminus \mathcal{E}$
   where $t \neq s$ with the pair $(s, \beta)$:
   
   $\text{wgt}(s, \beta) = w(s, t) + \text{wgt}(t, \beta)$

   $\mathcal{P}(s, \beta, u) = \mathcal{P}(t, \beta, u)$ for all states $u$ 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

- $\mathcal{E}^{\max}_{\mathcal{M},s}(\diamondsuit G) = \mathcal{E}^{\max}_{\mathcal{N},s}(\diamondsuit G)$ 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 \( \mathcal{N} = \mathcal{M} \setminus \mathcal{E} \)

- \( \mathcal{M} \) is weight-divergent iff \( \mathcal{N} \) is weight-divergent
- \( \mathbb{E}_{\mathcal{M},s}(\Theta G) = \mathbb{E}_{\mathcal{N},s}(\Theta G) \) for all states \( s \) in \( \mathcal{M} \)
- \( \|\mathcal{N}\| \leq \|\mathcal{M}\| - 1 \)

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
- $\mathcal{E}_{\mathcal{M},s}^{\max}(\varnothing G) = \mathcal{E}_{\mathcal{N},s}^{\max}(\varnothing G)$ for all states $s$ in $\mathcal{M}$
- $\|\mathcal{N}\| \leq \|\mathcal{M}\| - 1$

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 $\mathbb{E}_{\mathcal{M}}^{\text{max}}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent
Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $E^\text{max}_\mathcal{M}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $E^\text{max}_\mathcal{M}(\text{MP})$ and an optimal MD-scheduler $\sigma$
Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^{\text{max}}_{\mathcal{M}}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^{\text{max}}_{\mathcal{M}}(\text{MP})$ and an optimal MD-scheduler $\sigma$

2. if $\mathbb{E}^{\text{max}}_{\mathcal{M}}(\text{MP}) < 0$ 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 $M$ with $E^\text{max}_M(MP) \leq 0$

task: check if $M$ is weight-divergent

1. compute $E^\text{max}_M(MP)$ and an optimal MD-scheduler $\sigma$

2. if $E^\text{max}_M(MP) < 0$ then return “no”

3. pick a BSCC $\mathcal{E}$ of the MC induced by $\sigma$

strongly connected MC with expected mean-payoff 0
Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $\mathbb{E}^\text{max}_\mathcal{M}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $\mathbb{E}^\text{max}_\mathcal{M}(\text{MP})$ and an optimal MD-scheduler $\sigma$

2. if $\mathbb{E}^\text{max}_\mathcal{M}(\text{MP}) < 0$ then return “no”

3. pick a BSCC $\mathcal{E}$ of the MC induced by $\sigma$

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 $E^\text{max}_\mathcal{M}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $E^\text{max}_\mathcal{M}(\text{MP})$ and an optimal MD-scheduler $\sigma$

2. if $E^\text{max}_\mathcal{M}(\text{MP}) < 0$ then return “no”

3. pick a BSCC $\mathcal{E}$ of the MC induced by $\sigma$

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 $E_{\mathcal{M}}^{\text{max}}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $E_{\mathcal{M}}^{\text{max}}(\text{MP})$ and an optimal MD-scheduler $\sigma$

2. if $E_{\mathcal{M}}^{\text{max}}(\text{MP}) < 0$ then return “no”

3. pick a BSCC $\mathcal{E}$ of the MC induced by $\sigma$

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 $E_M^{\max}(MP) \leq 0$
task: check if $\mathcal{M}$ is weight-divergent

1. compute $E_M^{\max}(MP)$ and an optimal MD-scheduler $\sigma$

2. if $E_M^{\max}(MP) < 0$ then return “no”

If $\mathcal{M}$ is not weight-divergent then the algorithm has generated an MDP $\mathcal{N}$ with $E_{M,s}^{\max}(\emptyset G) = E_{N,s}^{\max}(\emptyset G)$ recursively to the MDP $\mathcal{M}\setminus \epsilon$.

Otherwise return “yes, $\mathcal{M}$ is weight-divergent”.
Checking weight-divergence

given: strongly connected MDP \( \mathcal{M} \) with \( E^\mathcal{M}_{\text{max}}(\text{MP}) \leq 0 \)

task: check if \( \mathcal{M} \) is weight-divergent

1. compute \( E^\mathcal{M}_{\text{max}}(\text{MP}) \) and an optimal MD-scheduler \( \sigma \)

2. if \( E^\mathcal{M}_{\text{max}}(\text{MP}) < 0 \) then return “no”

If \( \mathcal{M} \) is not weight-divergent then the algorithm has generated an MDP \( \mathcal{N} \) with \( E^\mathcal{M}_{\sigma,\text{s}}(\mathcal{G}) = E^\mathcal{N}_{\sigma,\text{s}}(\mathcal{G}) \) and \( E^\sigma_{\mathcal{N},\text{s}}(\text{“total weight”}) = -\infty \) for each improper scheduler \( \sigma \). ... as \( \mathcal{N} \) has no zero-ECs ...
Checking weight-divergence

given: strongly connected MDP $\mathcal{M}$ with $E_{\mathcal{M}}^{\max}(\text{MP}) \leq 0$

task: check if $\mathcal{M}$ is weight-divergent

1. compute $E_{\mathcal{M}}^{\max}(\text{MP})$ and an optimal MD-scheduler $\sigma$

2. if $E_{\mathcal{M}}^{\max}(\text{MP}) < 0$ then return “no”

If $\mathcal{M}$ is not weight-divergent then the algorithm has generated an MDP $\mathcal{N}$ with $E_{\mathcal{M},s}(\otimes G) = E_{\mathcal{N},s}(\otimes G)$ and $E_{\mathcal{N},s}^{\sigma}(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$.

... $E_{\mathcal{N},s}^{\max}(\otimes G)$ computable via Bellman equations ...
Maximal expected accumulated weight

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

s.t. $Pr_s^{\text{max}}(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = E_s^{\text{max}}(\Diamond G)$ for $s \in S$

If $E_s(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

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

[ Bertsekas/Tsitsiklis’91 ]
Maximal expected accumulated weight

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

s.t. $\Pr_s^{\text{max}}(\Diamond G) = 1$ for all $s \in S$

task: compute $x_s = E_s^{\text{max}}(\bigoplus G)$ for $s \in S$

If $E_s^\sigma(\text{“total weight”}) = -\infty$ for each improper scheduler $\sigma$ then:

- $x_s < +\infty$ 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 $\mathcal{M} = (S, \text{Act}, P, wgt)$ and $G \subseteq S$

s.t. $\Pr^\text{max}_s(\diamondsuit G) = 1$ for all $s \in S$

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

If $E^\sigma_s(\text{"total weight"}) = -\infty$ for each improper scheduler $\sigma$ then:

- $x_s < +\infty$ 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 $x_s = E^\text{max}_{\mathcal{N}, s}(\diamondsuit G)$

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 $\mathcal{M}$
- iamond $\mathcal{G}$: goal
  accumulated weight until reaching a goal state

relaxed requirement:

$\Pr^{\text{max}}(\lozenge \text{goal}) = 1$

maximal expectation

$\mathbb{E}^{\text{max}}(\lozenge \text{goal})$

maximum over all proper schedulers

- Bertsekas/Tsitsiklis’91
- de Alfaro’99
Maximal conditional expectations

- weighted MDP $\mathcal{M}$
- $\Diamond \text{goal}$
- accumulated weight until reaching a goal state

Relaxed requirement:

$$\Pr_{\text{max}}(\Diamond \text{goal}) > 0$$

Maximal conditional expectation

$$\mathbb{E}_{\text{max}}( \Diamond \text{green} \mid \Diamond \text{goal} )$$

Maximum over all positive schedulers
Maximal conditional expectations

**weighted MDP** $\mathcal{M}$

**$\Diamond \textit{goal}$**
accumulated weight until reaching a goal state

relaxed requirement:
$\Pr^{\text{max}}(\Diamond \textit{goal}) > 0$

assumption:
non-negative weights

**maximal conditional expectation**
$\mathbb{E}^{\text{max}}(\Diamond \textit{green} \mid \Diamond \textit{goal})$

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 exist 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 exist 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

Maximal conditional expected reward:

\[
\mathbb{E}^{\text{max}}(\Diamond \text{goal} \mid \Diamond \text{goal}) = ???
\]

\[
\text{rew}(s_1, \gamma) = r
\]
\[
\text{rew}(s_2, \beta) = 1
\]
\[
\text{rew}(s_i, \alpha) = 0
\]
Maximal conditional expected reward

```
\text{rew}(s_1, \gamma) = r
\text{rew}(s_2, \beta) = 1
\text{rew}(s_i, \alpha) = 0
```

“choose always $\alpha$ in state $s_2$”:

\[
\frac{1}{2} \cdot r + \frac{1}{2} \cdot 0 = \frac{r}{2}
\]
Maximal conditional expected reward

\[
\text{Rew}(s_1, \gamma) = r \\
\text{Rew}(s_2, \beta) = 1 \\
\text{Rew}(s_i, \alpha) = 0
\]

“choose always \(\alpha\) in state \(s_2\)”: \[
\frac{1}{2} r + \frac{1}{2} \cdot 0 = \frac{r}{2}
\]

“choose always \(\beta\) in state \(s_2\)”: \[
\frac{1}{2} r + 0 = r
\]
Maximal conditional expected reward

 rew(s₁, γ) = r
 rew(s₂, β) = 1
 rew(sᵢ, α) = 0

“choose β exactly for the first $n$ visits of $s₂$”

$$\frac{1}{2} \cdot r + \frac{1}{2} \cdot \frac{1}{2^n} \cdot n$$

$$\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2^n}$$
Maximal conditional expected reward

\[ \text{rew}(s_1, \gamma) = r \]
\[ \text{rew}(s_2, \beta) = 1 \]
\[ \text{rew}(s_i, \alpha) = 0 \]

“choose \( \beta \) exactly 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

\[ \text{rew}(s_1, \gamma) = r \]
\[ \text{rew}(s_2, \beta) = 1 \]
\[ \text{rew}(s_i, \alpha) = 0 \]

“choose \( \beta \) exactly for the first \( n \) visits of \( s_2 \)”

\[
\frac{1/2 \cdot r + 1/2 \cdot \frac{1}{2^n} \cdot n}{1/2 + 1/2 \cdot \frac{1}{2^n}} = r + \frac{n-r}{2^n+1} > r \quad \text{iff} \quad n > r
\]
Maximal conditional expected reward

\[
\alpha, \frac{1}{2} \quad \gamma \quad \alpha, \frac{1}{2} \quad \beta, \frac{1}{2} \quad 1/2
\]

\[
\begin{align*}
\text{rew}(s_1, \gamma) &= r \\
\text{rew}(s_2, \beta) &= 1 \\
\text{rew}(s_i, \alpha) &= 0
\end{align*}
\]

“choose \( \beta \) exactly for the first \( n \) visits of \( s_2 \)”

\[
\frac{1}{2} \cdot r + \frac{1}{2} \cdot \frac{1}{2^n} \cdot n \quad = \quad r + \frac{n - r}{2^n + 1} > r \quad \text{iff} \quad n > r
\]

optimal value is achieved for \( n = r + 2 \)
Maximal conditional expected reward

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 $r$

rew($s_1, \gamma$) = $r$
rew($s_2, \beta$) = 1
rew($s_i, \alpha$) = 0
Maximal conditional expected reward

\[
\begin{align*}
rew(s_1, \gamma) &= r \\
rew(s_2, \beta) &= 1 \\
rew(s_i, \alpha) &= 0
\end{align*}
\]

maximal conditional reward until \textit{goal}

... is finite for state \( s_0 \), namely \( r + \frac{2}{2^{r+2}+1} \)
Maximal conditional expected reward

\[ \text{maximal conditional reward until } \text{goal} \]

... is finite for state \( s_0 \), namely \( r + \frac{2}{2^r+2+1} \)

... but infinite for \( s_2 \)

\[ \sup_{n \in \mathbb{N}} \frac{n}{2^n} = \infty \]
Problem statement

given: MDP $\mathcal{M} = (S, Act, P, rew, s_0)$ and $F, G \subseteq S$ such that $\Pr_{s_0}^{\max} (\Diamond F \mid \Diamond G) = 1$

task: ...

$\Pr_{s_0}^{\max} (\Diamond F \mid \Diamond G) = 1$ iff there is scheduler $\sigma$ s.t.

1. $\Pr_{s_0}^{\sigma} (\Diamond G) > 0$ and
2. $\Pr_{s_0}^{\sigma} (\Diamond F \mid \Diamond G) = 1$
Problem statement

given: MDP \( \mathcal{M} = (S, Act, P, rew, s_0) \) and \( F, G \subseteq S \)
such that \( \Pr_{s_0}^{\text{max}}(\Diamond F \mid \Diamond G) = 1 \)

task: compute \( \mathbb{E}_{s_0}^{\text{max}}(\Diamond F \mid \Diamond G) \)

maximal conditional accumulated reward to reach \( F \)
under all schedulers \( \sigma \) where \( \Pr_{s_0}^{\sigma}(\Diamond G) > 0 \)
and \( \Pr_{s_0}^{\sigma}(\Diamond F \mid \Diamond G) = 1 \)
Problem statement

given: MDP $\mathcal{M} = (S, Act, P, rew, s_0)$ and $F, G \subseteq S$
such that $\Pr^\max_{s_0}(\Diamond F | \Diamond G) = 1$

task: compute $\mathbb{E}^\max_{s_0}(\Diamond F | \Diamond G)$

after some preprocessing and cleaning-up:

1. all states are reachable from $s_0$

2. $F = G = \{\text{goal}\}$ for a trap state $\text{goal}$

3. there is another trap state $\text{fail}$ with
   $\Pr^\min_s(\Diamond (\text{goal} \lor \text{fail})) = 1$ for all states $s$
Given a scheduler $\sigma$ with $\Pr_{s_0}^\sigma(\Diamond\text{goal}) > 0$, let:

$$\text{CE}^\sigma = \mathbb{E}_{s_0}^\sigma(\Diamond\text{goal} | \Diamond\text{goal})$$

Maximal conditional expectation:

$$\text{CE}^{\max} = \sup_{\sigma} \text{CE}^\sigma$$

ranging over all schedulers $\sigma$ with $\Pr_{s_0}^\sigma(\Diamond\text{goal}) > 0$
Given a scheduler $\sigma$ with $\Pr_{s_0}^{\sigma}(\Diamond goal) > 0$, let:

$$CE^\sigma = E_{s_0}(\Diamond goal \mid \Diamond goal)$$

Maximal conditional expectation:

$$CE_{\text{max}} = \sup_{\sigma} CE^\sigma$$

supremum over all deterministic reward-based schedulers

$\sigma : S \times \mathbb{N} \rightarrow \text{Act}$
Checking finiteness

Given a scheduler $\sigma$ with $\Pr_{s_0}^{\sigma}(\Diamond \text{goal}) > 0$, let:

$$CE^\sigma = E_{s_0}^{\sigma}(\diamond \text{goal} | \Diamond \text{goal})$$

Maximal conditional expectation:

$$CE^{\max} = \sup_{\sigma} CE^\sigma$$

Checking finiteness in polynomial time:

$$CE^{\max} < \infty \iff \{ \text{there is no scheduler } \sigma \text{ s.t. } \Pr_{s_0}^{\sigma}(\Diamond \text{goal}) = 0 \text{ and there is a reachable positive } \sigma\text{-cycle} \}$$
If $CE^{\text{max}} < \infty$ then ...
If \( C\!E^{\text{max}} < \infty \) then ...

- pseudo-polynomial algorithm to compute an upper bound \( C\!E^{\text{ub}} \) for \( C\!E^{\text{max}} \)

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 $\text{CE}^{\text{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\text{CE}^{\text{ub}}$ for $\text{CE}^{\text{max}}$

- threshold problem “is $\text{CE}^{\text{max}} \geq \vartheta$?” 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 $\geq \in \{>, \geq\}$
task: check whether $\text{CE}^{\text{max}} \geq \vartheta$
If $\mathbf{CE}^{\text{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbf{CE}^{\text{ub}}$ for $\mathbf{CE}^{\text{max}}$

- threshold problem “is $\mathbf{CE}^{\text{max}} \geq \vartheta$?” is PSPACE-hard, and PSPACE-complete for acyclic MDPs

- there exists a saturation point $\vartheta$ such that optimal schedulers behave memoryless from reward $\vartheta$ on

... and maximize the probability to reach the goal state
If $CE^{\text{max}} < \infty$ then ... 

- pseudo-polynomial algorithm to compute an upper bound $CE^{ub}$ for $CE^{\text{max}}$

- threshold problem “is $CE^{\text{max}} \geq \emptyset$?” is PSPACE-hard, and PSPACE-complete for acyclic MDPs

- there exists a saturation point $\emptyset$ such that optimal schedulers behave memoryless from reward $\emptyset$ on

- pseudo-polynomial threshold algorithm
If $CE_{\text{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $CE_{\text{ub}}$ for $CE_{\text{max}}$

- threshold problem “is $CE_{\text{max}} \geq \vartheta$?” is PSPACE-hard, and PSPACE-complete for acyclic MDPs

- there exists a saturation point \( \varrho \) such that optimal schedulers behave memoryless from reward \( \varrho \) on

- pseudo-polynomial threshold algorithm: generates a scheduler \( \sigma \) s.t. $CE_{\sigma} > \vartheta$ or $CE_{\text{max}} = CE_{\sigma} = \vartheta$ (if existent)
If $\mathbf{CE}^{\text{max}} < \infty$ then ...

- pseudo-polynomial algorithm to compute an upper bound $\mathbf{CE}^{\text{ub}}$ for $\mathbf{CE}^{\text{max}}$

- threshold problem “is $\mathbf{CE}^{\text{max}} \geq \vartheta$?” is PSPACE-hard, and PSPACE-complete for acyclic MDPs

- there exists a saturation point $\varrho$ such that optimal schedulers behave memoryless from reward $\varrho$ on

- pseudo-polynomial threshold algorithm: generates a scheduler $\sigma$ s.t. $\mathbf{CE}^\sigma > \vartheta$ or $\mathbf{CE}^{\text{max}} = \mathbf{CE}^\sigma = \vartheta$

- exponential-time algorithm to compute $\mathbf{CE}^{\text{max}}$ interleaves scheduler-improvement steps with threshold algorithm
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 $\Diamond \text{goal}$
Computing an upper bound

unconditional total expected reward in a new MDP $\mathcal{N}$ that simulates $\mathcal{M}$ under the condition $\Diamond \text{goal}$

**first mode:**

* augments states with the reward accumulated so far up to $R^{\text{max}} = \sum_s \max_\alpha \text{rew}(s, \alpha)$
* reward 0 for all state-actions in the first mode
* mode switch from $(s, r)$ via action $\alpha$ with reward $r'$
  if $r' \overset{\text{def}}{=} r + \text{rew}(s, \alpha) > R^{\text{max}}$

**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 $\diamond \text{goal}$

**first mode:**

* augments states with the reward accumulated so far up to $R^{\text{max}} = \sum_s \max_\alpha \text{rew}(s, \alpha)$

* reward 0 for all state-actions in the first mode

* mode switch from $(s, r)$ via action $\alpha$ with reward $r'$ if $r' \overset{\text{def}}{=} r + \text{rew}(s, \alpha) > R^{\text{max}}$

**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 $\varphi$ and optimal decisions for state-reward pairs $(s, r)$ with $r \geq \varphi$

FOR $r = \varphi - 1, \varphi - 2, \ldots, 0$ 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 $CE^\sigma \geq \psi$ for the generated scheduler $\sigma$
Sketch of the threshold algorithm

compute the saturation point \( \varphi \) and optimal decisions for state-reward pairs \((s, r)\) with \( r \geq \varphi \)

FOR \( r = \varphi - 1, \varphi - 2, \ldots, 0 \) 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 \( CE^\sigma \succeq \psi \) for the generated scheduler \( \sigma \)
Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0, 1]$ such that $y > z$ and $x + p y > 0$, $x + p z > 0$.

\[
\text{CE}^\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad \text{CE}^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}
\]
Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0, 1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

\[
\begin{align*}
\text{CE}^\sigma &= \frac{\rho + p(ry + \theta)}{x + py} \\
\text{CE}^\tau &= \frac{\rho + p(rz + \zeta)}{x + pz}
\end{align*}
\]

\[
\text{CE}^\sigma > \text{CE}^\tau \text{ iff } r + \frac{\theta - \zeta}{y - z} > \max \left\{ \text{CE}^\sigma, \text{CE}^\tau \right\}
\]
Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0, 1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$CE^\sigma = \frac{\rho + p(ry + \theta)}{x + py}$$

$$CE^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$CE^\sigma > CE^\tau$ iff

$$r + \frac{\theta - \zeta}{y - z} > \max \left\{ CE^\sigma, CE^\tau \right\}$$

does not depend on $\rho, x, p$
Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0, 1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$Ce^\sigma = \frac{\rho + p(ry + \theta)}{x + py}$$

$$Ce^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$Ce^\sigma > Ce^\tau \iff r + \frac{\theta - \zeta}{y - z} > \max \{ Ce^\sigma, Ce^\tau \}$$

threshold algorithm:

$$r + \frac{\theta - \zeta}{y - z} \geq \vartheta \iff \theta - (\vartheta - r)y \geq \zeta - (\vartheta - r)z$$
Let $\rho, \theta, \zeta, r \in \mathbb{R}$, $p, x, y, z \in [0, 1]$ such that $y > z$ and $x + py > 0$, $x + pz > 0$.

$$CE^\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad CE^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$CE^\sigma > CE^\tau \iff r + \frac{\theta - \zeta}{y - z} > \max \left\{ CE^\sigma, CE^\tau \right\}$$

threshold algorithm:

$$r + \frac{\theta - \zeta}{y - z} \geq \vartheta \iff \theta - (\vartheta - r)y \geq \zeta - (\vartheta - r)z$$

... 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 $x + py > 0, x + pz > 0$.

$$\text{CE}^\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad \quad \text{CE}^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$\text{CE}^\sigma > \text{CE}^\tau \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max \left\{ \text{CE}^\sigma, \text{CE}^\tau \right\}$$

saturation point: smallest value $r$ such that

$$r + \frac{\theta - \zeta}{y - z} \geq \text{CE}^\max \quad \text{for all } \tau$$

where $\sigma$ 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 $x + py > 0$, $x + pz > 0$.

$$CE^\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad \text{and} \quad CE^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$$CE^\sigma > CE^\tau \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max \{CE^\sigma, CE^\tau\}$$

saturation point: smallest value $r$ such that

$$r + \frac{\theta - \zeta}{y - z} \geq CE^{\text{ub}}$$

for all $\tau$.

where $\sigma$ 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 $x + py > 0$, $x + pz > 0$.

$$\text{CE}_\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad \text{CE}_\tau = \frac{\rho + p(rz + \zeta)}{x + pz}$$

$\text{CE}_\sigma > \text{CE}_\tau$ iff

$$r + \frac{\theta - \zeta}{y - z} > \max \left\{ \text{CE}_\sigma, \text{CE}_\tau \right\}$$

saturation point: smallest value $r$ such that

$$r + \frac{\theta - \zeta}{y - z} \geq \text{CE}^{\text{ub}}$$

for all $\tau$

... it suffices to consider “one-step variants” $\tau$ of $\sigma$
Computing the maximal conditional expectation using a scheduler-improvement approach with iterative calls of the threshold algorithm

If $CE_{\text{max}} \geq \vartheta$ then the threshold algorithm generates a scheduler $\sigma$ s.t. $CE^\sigma > \vartheta$ or $CE_{\text{max}} = CE^\sigma = \vartheta$. 
Computing the maximal conditional expectation using a scheduler-improvement approach with iterative calls of the threshold algorithm

let $\sigma$ be an arbitrary scheduler;

REPEAT

$\vartheta := \text{CE}_\sigma$;

$\sigma := \text{outcome of the algorithm for threshold } \vartheta$

UNTIL $\text{CE}_\sigma = \vartheta$

If $\text{CE}^{\text{max}} \geq \vartheta$ then the threshold algorithm generates a scheduler $\sigma$ s.t. $\text{CE}_\sigma > \vartheta$ or $\text{CE}^{\text{max}} = \text{CE}_\sigma = \vartheta$. 
Computing the maximal conditional expectation

using a scheduler-improvement approach with iterative calls of the threshold algorithm

\[
\text{let } \sigma \text{ be } \ldots \\
\text{REPEAT} \\
\quad \vartheta := \text{CE}^\sigma; \\
\quad \sigma := \text{outcome of the algorithm for threshold } \vartheta \\
\text{UNTIL } \text{CE}^\sigma = \vartheta
\]

time complexity: double exponential

If \( \text{CE}^\text{max} \geq \vartheta \) then the threshold algorithm generates a scheduler \( \sigma \) s.t. \( \text{CE}^\sigma > \vartheta \) or \( \text{CE}^\text{max} = \text{CE}^\sigma = \vartheta \).
Computing the maximal conditional expectation using a scheduler-improvement approach with iterative calls of the threshold algorithm

let \( \sigma \) be . . . 

**REPEAT**

\[ \vartheta := CE^\sigma; \]

\[ \sigma := \text{outcome of the algorithm for threshold } \vartheta \]

**UNTIL** \( CE^\sigma = \vartheta \)

time complexity: double exponential

in the worst-case: \( |MD|^\varphi \) iterations where the saturation point \( \varphi \) can be exponential in \( \text{size}(\mathcal{M}) \)
Computing the maximal conditional expectation

exponential-time algorithm for computing $\text{CE}^{\text{max}}$

* 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 $CE^{\text{max}}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

\[
CE^\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad \text{and} \quad CE^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}
\]

\[
CE^\sigma > CE^\tau \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} > \max \left\{ CE^\sigma, CE^\tau \right\}
\]

If this holds for all $\tau$ then $\sigma$ is optimal for level $r$. 
Computing the maximal conditional expectation

exponential-time algorithm for computing $\mathbf{CE}^{\text{max}}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

\[
\mathbf{CE}^\sigma = \frac{\rho + p(ry + \theta)}{x + py} \quad \mathbf{CE}^\tau = \frac{\rho + p(rz + \zeta)}{x + pz}
\]

$\mathbf{CE}^\sigma > \mathbf{CE}^\tau$ iff

\[
r + \frac{\theta - \zeta}{y - z} > \max \left\{ \mathbf{CE}^\sigma, \mathbf{CE}^\tau \right\}
\]

use these values as threshold values
Computing the maximal conditional expectation

exponential-time algorithm for computing $\mathbf{CE}^{\text{max}}$

* freezes level-wise optimal decisions
* uses threshold algorithm for scheduler-improvement steps
* maintains an interval of feasible threshold candidates

\[
\mathbf{CE}^{\sigma} = \frac{\rho + p(ry + \theta)}{x + py} \quad \text{and} \quad \mathbf{CE}^{\tau} = \frac{\rho + p(rz + \zeta)}{x + pz}
\]

$\mathbf{CE}^{\sigma} > \mathbf{CE}^{\tau}$ iff $r + \frac{\theta - \zeta}{y - z} > \max \{ \mathbf{CE}^{\sigma}, \mathbf{CE}^{\tau} \}$

in total: $O(\wp \cdot |\text{MD}|)$ 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

<table>
<thead>
<tr>
<th>Tool</th>
<th>Features</th>
<th>Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRISM</td>
<td>various models and logics, symbolic, explicit</td>
<td>(Oxford, Birmingham)</td>
</tr>
<tr>
<td></td>
<td>and hybrid engines</td>
<td></td>
</tr>
<tr>
<td>STORM</td>
<td>PCTL, bisimulation, parametric models</td>
<td>(Aachen)</td>
</tr>
<tr>
<td>Modest</td>
<td>MDPs (with clocks)</td>
<td>(Saarbrücken, Twente)</td>
</tr>
<tr>
<td>PARAM</td>
<td>parametric models</td>
<td>(Saarbrücken)</td>
</tr>
<tr>
<td>ProbDiVinE</td>
<td>parallel LTL model checker</td>
<td>(Brno)</td>
</tr>
<tr>
<td>iscasMC</td>
<td>lazy determinization</td>
<td>(Beijing, Liverpool)</td>
</tr>
</tbody>
</table>
Thank You