Analyse eines Forschungsthemas
Stochastic Shortest Paths

Maximilian Starke

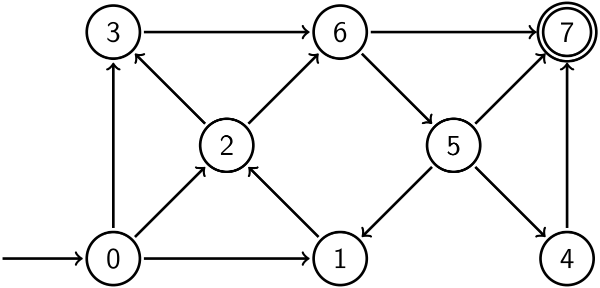
Fakultät für Informatik
Technische Universität Dresden

3. Januar 2023

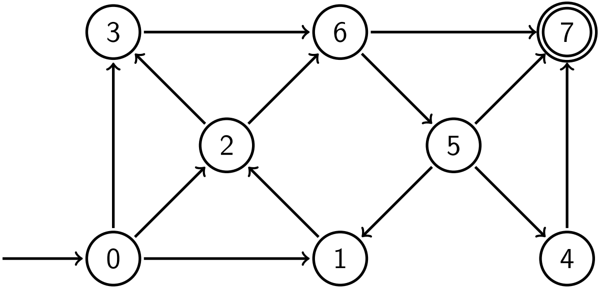
Section 1

Introduction

- The *simplest* shortest path problem

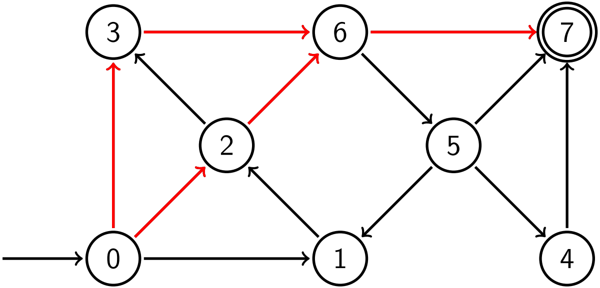


- The *simplest* shortest path problem



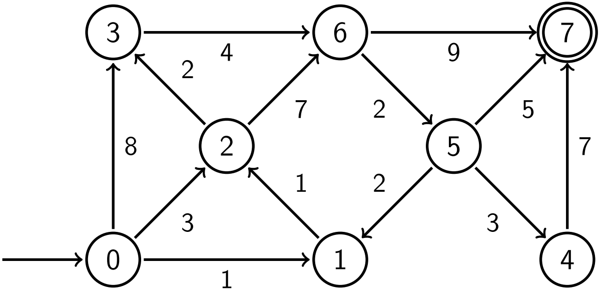
- Task
  - Find the shortest path (*number of hops*)!

- The *simplest* shortest path problem

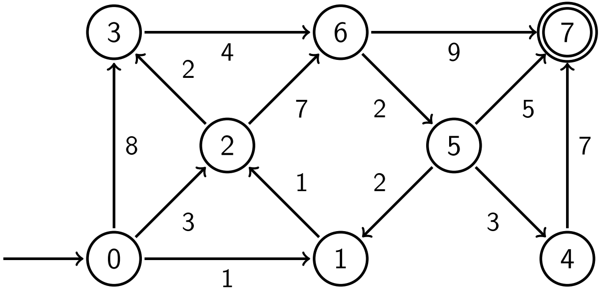


- Task
  - Find the shortest path (*number of hops*)!

- The *classical*, *non-stochastic*, *deterministic* shortest path problem

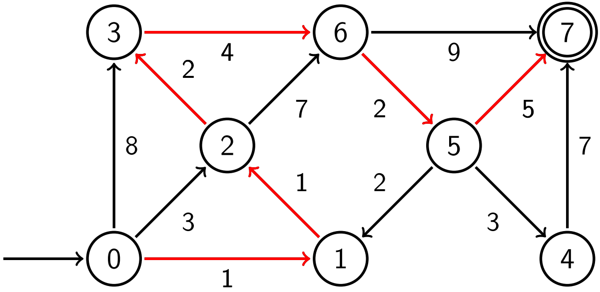


- The *classical*, *non-stochastic*, *deterministic* shortest path problem



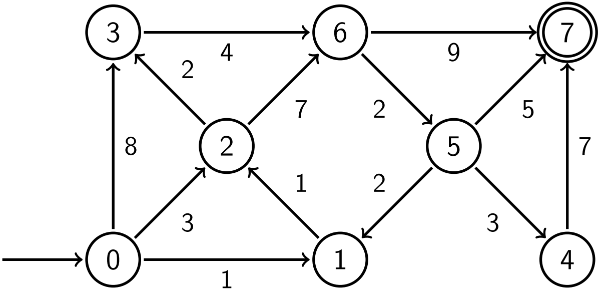
- Task
  - Find the path with the minimal weight sum!

- The *classical*, *non-stochastic*, *deterministic* shortest path problem



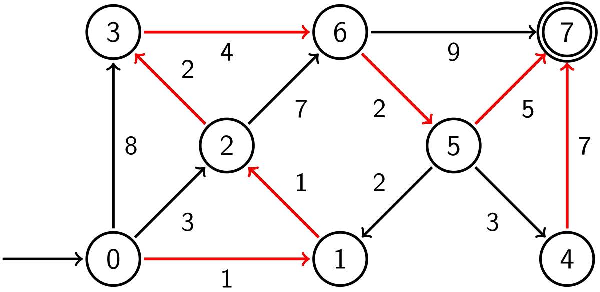
- Task
  - Find the path with the minimal weight sum!

- The *classical, non-stochastic, deterministic* shortest path problem



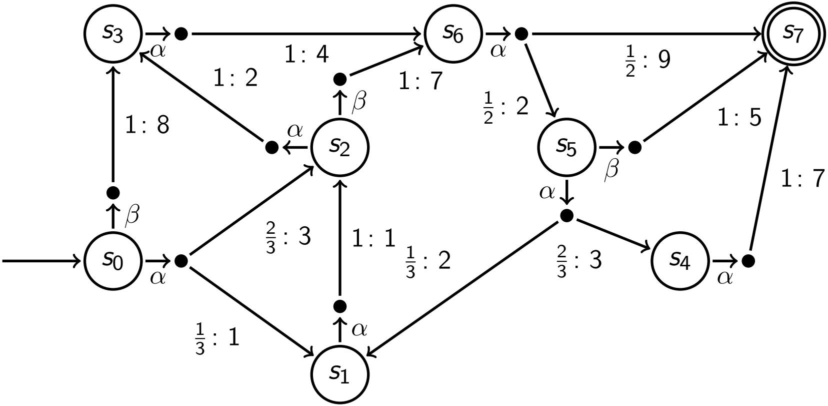
- Task
  - Find the path with the minimal weight sum!
  - Give a strategy to always reach the goal while collecting minimal weight!

- The *classical, non-stochastic, deterministic* shortest path problem

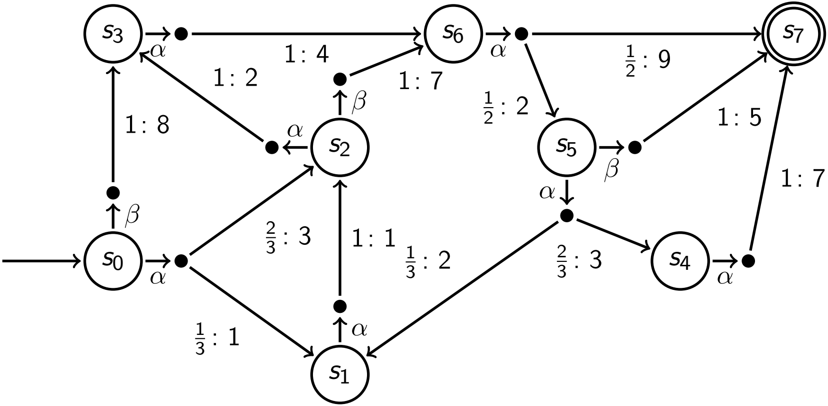


- Task
  - Find the path with the minimal weight sum!
  - Give a strategy to always reach the goal while collecting minimal weight!

- The *stochastic* shortest path problem

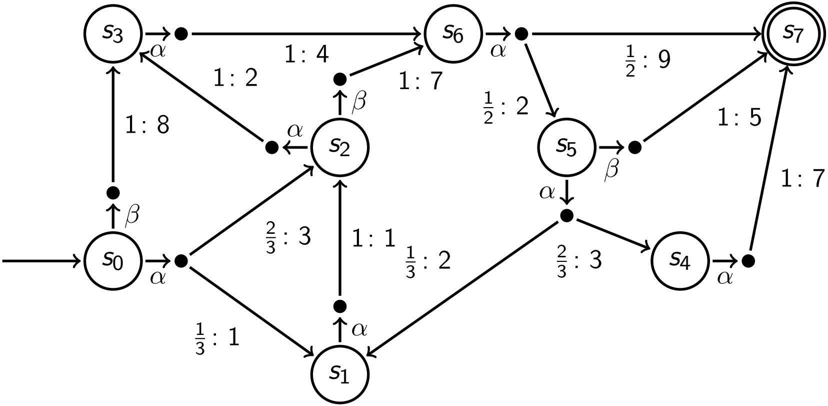


- The *stochastic* shortest path problem



- Markov Decision Process (MDP)

- The *stochastic* shortest path problem



- Markov Decision Process (MDP)
- Task
  - Give a strategy to reach the goal with minimal *expected* accumulated weights!

MDP - Markov Decision Process (I)

definition

An MDP is a tuple

$$\mathcal{M} = (S, Act, P, s_{init}, wgt)$$

MDP - Markov Decision Process (I)

definition

An MDP is a tuple

$$\mathcal{M} = (S, Act, P, s_{init}, wgt)$$

where

- S is a finite set of states.

MDP - Markov Decision Process (I)

definition

An MDP is a tuple

$$\mathcal{M} = (S, Act, P, s_{init}, wgt)$$

where

- S is a finite set of states.
- Act is a finite set of actions.

MDP - Markov Decision Process (I)

definition

An MDP is a tuple

$$\mathcal{M} = (S, Act, P, s_{init}, wgt)$$

where

- S is a finite set of states.
- Act is a finite set of actions.
- $P : S \times Act \dashrightarrow Distr(S)$ maps every enabled state - action - pair (s, α) to a probability distribution over the states.

MDP - Markov Decision Process (I)

definition

An MDP is a tuple

$$\mathcal{M} = (S, Act, P, s_{init}, wgt)$$

where

- S is a finite set of states.
- Act is a finite set of actions.
- $P : S \times Act \dashrightarrow Distr(S)$ maps every enabled state - action - pair (s, α) to a probability distribution over the states.
- $s_{init} \in S$ is some designated initial state.

MDP - Markov Decision Process (I)

definition

An MDP is a tuple

$$\mathcal{M} = (S, Act, P, s_{init}, wgt)$$

where

- S is a finite set of states.
- Act is a finite set of actions.
- $P : S \times Act \dashrightarrow Distr(S)$ maps every enabled state - action - pair (s, α) to a probability distribution over the states.
- $s_{init} \in S$ is some designated initial state.
- $wgt : S \times Act \to \mathbb{Z}$ is some weight (reward) function.

MDP - Markov Decision Process (II)

Paths

$$\pi \in \mathrm{Paths}(\mathcal{M})$$

MDP - Markov Decision Process (II)

Paths

$$\pi \in \mathrm{Paths}(\mathcal{M})$$

- $\pi = (s_0, \alpha_0, s_1, \alpha_1, \dots)$

MDP - Markov Decision Process (II)

Paths

$$\pi \in \mathrm{Paths}(\mathcal{M})$$

- $\pi = (s_0, \alpha_0, s_1, \alpha_1, \dots)$
- π is some maximal finite or infinite path

MDP - Markov Decision Process (II)

Paths

$$\pi \in \mathrm{Paths}(\mathcal{M})$$

- $\pi = (s_0, \alpha_0, s_1, \alpha_1, \ldots)$
- π is some maximal finite or infinite path

Let $F \subseteq S$ be some set of goal states...

MDP - Markov Decision Process (II)

Paths

$$\pi \in \mathrm{Paths}(\mathcal{M})$$

- $\pi = (s_0, \alpha_0, s_1, \alpha_1, \dots)$
- π is some maximal finite or infinite path

Let $F \subseteq S$ be some set of goal states...

accumulated weights until reaching a goal

$$\oplus F : \mathrm{Paths}(\mathcal{M}^{\mathfrak{S}}) \to \mathbb{Q} :$$

$$\oplus F(\pi) = \begin{cases} wgt(\hat{\pi}), & \hat{\pi} \text{ is shortest prefix of } \pi \text{ s.t. } \mathrm{last}(\hat{\pi}) \in F \\ \infty, & \text{otherwise, i.e. } \pi \not\models \Diamond F \end{cases}$$

Section 2

The classic stochastic shortest path problem

The classic stochastic shortest path problem

- given:
  - a single goal state

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\diamondplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\diamondplus goal)$$

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\oplus goal)$$

- Well known for a long time:
  - There exists an optimal memoryless deterministic scheduler $\mathfrak{S}$.

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\oplus goal)$$

- Well known for a long time:
  - There exists an optimal memoryless deterministic scheduler $\mathfrak{S}$.
  - $\mathfrak{S}$ is computable by solving a LP

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\oplus goal)$$

- Well known for a long time:
  - There exists an optimal memoryless deterministic scheduler $\mathfrak{S}$.
  - $\mathfrak{S}$ is computable by solving a LP
  - iterative algorithm:

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\oplus goal)$$

- Well known for a long time:
  - There exists an optimal memoryless deterministic scheduler $\mathfrak{S}$.
  - $\mathfrak{S}$ is computable by solving a LP
  - iterative algorithm:
    - start at any feasible scheduler

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\oplus goal)$$

- Well known for a long time:
  - There exists an optimal memoryless deterministic scheduler $\mathfrak{S}$.
  - $\mathfrak{S}$ is computable by solving a LP
  - iterative algorithm:
    - start at any feasible scheduler
    - iterative improvement

The classic stochastic shortest path problem

- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$
  - goal is reachable from each state
- objective: Minimize the expected accumulated weight until reaching goal state.

$$\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus goal) := \inf_{\mathfrak{S} \in \mathrm{Schedulers}(M)} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\oplus goal)$$

- Well known for a long time:
  - There exists an optimal memoryless deterministic scheduler $\mathfrak{S}$.
  - $\mathfrak{S}$ is computable by solving a LP
  - iterative algorithm:
    - start at any feasible scheduler
    - iterative improvement
    - stop at an optimal vertex of the LP (corresponding to some MD scheduler)

The classic stochastic shortest path problem

- Can we do it better?

The classic stochastic shortest path problem

- Can we do it better? $\longrightarrow$ YES! - using **spider construction**!

The classic stochastic shortest path problem

- Can we do it better? $\longrightarrow$ YES! - using **spider construction**!
- given:
  - a single goal state
  - positive cycle condition: There is no cycle π with $\mathrm{wgt}(\pi) \leq 0$

The classic stochastic shortest path problem

- Can we do it better? $\longrightarrow$ YES! - using **spider construction**!
- given:
  - a single goal state
  - $\mathcal{M}$ is an MDP with arbitrary integer weights

The classic stochastic shortest path problem

- Can we do it better? $\longrightarrow$ YES! - using **spider construction**!
- given:
  - a single goal state
  - $\mathcal{M}$ is an MDP with arbitrary integer weights
  - goal is reachable from each state

The classic stochastic shortest path problem

- Can we do it better? $\longrightarrow$ YES! - using **spider construction**!
- given:
  - a single goal state
  - $\mathcal{M}$ is an MDP with arbitrary integer weights
  - goal is reachable from each state
- The following can be solved in polynomial time:
  - Check: $\mathbb{E}^{inf}_{\mathcal{M},s}(\oplus \mathrm{goal}) > -\infty$?
  - Compute $\mathbb{E}^{inf}_{\mathcal{M},s}$ if it is finite

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$
- Pick a 0-BSCC $\mathcal{E}$ of $\mathcal{M}$ and some vertex s_0 in $\mathcal{E}$.

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$
- Pick a 0-BSCC $\mathcal{E}$ of $\mathcal{M}$ and some vertex s_0 in $\mathcal{E}$.
- $\mathcal{M} \mapsto \mathcal{N} := \mathrm{Spider}_{\mathcal{E},s_0}(\mathcal{M})$

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$
- Pick a 0-BSCC $\mathcal{E}$ of $\mathcal{M}$ and some vertex s_0 in $\mathcal{E}$.
- $\mathcal{M} \mapsto \mathcal{N} := \mathrm{Spider}_{\mathcal{E},s_0}(\mathcal{M})$
- The spider construction is done by applying the following steps:
  1. Remove all actions $(s, \alpha_s) \in \mathcal{E}$

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$
- Pick a 0-BSCC $\mathcal{E}$ of $\mathcal{M}$ and some vertex s_0 in $\mathcal{E}$.
- $\mathcal{M} \mapsto \mathcal{N} := \mathrm{Spider}_{\mathcal{E},s_0}(\mathcal{M})$
- The spider construction is done by applying the following steps:
  1. Remove all actions $(s, \alpha_s) \in \mathcal{E}$
  2. Add actions (s, τ) for all $s \in \mathcal{E} \setminus \{s_0\}$ such that
     - $P_{\mathcal{N}}(s, \tau, s_0) := 1$

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$
- Pick a 0-BSCC $\mathcal{E}$ of $\mathcal{M}$ and some vertex s_0 in $\mathcal{E}$.
- $\mathcal{M} \mapsto \mathcal{N} := \mathrm{Spider}_{\mathcal{E},s_0}(\mathcal{M})$
- The spider construction is done by applying the following steps:
  1. Remove all actions $(s, \alpha_s) \in \mathcal{E}$
  2. Add actions (s, τ) for all $s \in \mathcal{E} \setminus \{s_0\}$ such that
     - $P_{\mathcal{N}}(s, \tau, s_0) := 1$
     - $wgt_{\mathcal{N}}(s, \tau) := wgt(s, s_0)$

Spider Construction

- Idea: construct a new MDP $\mathcal{N}$ from the given MDP $\mathcal{M}$
- Pick a 0-BSCC $\mathcal{E}$ of $\mathcal{M}$ and some vertex s_0 in $\mathcal{E}$.
- $\mathcal{M} \mapsto \mathcal{N} := \mathrm{Spider}_{\mathcal{E},s_0}(\mathcal{M})$
- The spider construction is done by applying the following steps:
  1. Remove all actions $(s, \alpha_s) \in \mathcal{E}$
  2. Add actions (s, τ) for all $s \in \mathcal{E} \setminus \{s_0\}$ such that
     - $P_{\mathcal{N}}(s, \tau, s_0) := 1$
     - $wgt_{\mathcal{N}}(s, \tau) := wgt(s, s_0)$
  3. For each $s \in \mathcal{E} \setminus \{s_0\}$ and $\beta \in \mathrm{Act}_{\mathcal{M}}(s) \setminus \{\alpha_s\}$ let us replace (s, β) by (s_0, β) where
     - $P_{\mathcal{N}}(s_0, \beta, u) := P_{\mathcal{M}}(s, \beta, u)$
     - $wgt_{\mathcal{N}}(s_0, \beta) + wgt(s, s_0) = wgt_{\mathcal{M}}(s, \beta)$

Classification of paths

A path $\pi \in \mathrm{InfPaths}(\mathcal{M})$ is called

- pumping $:\Leftrightarrow \liminf_{n\to\infty}(\mathrm{wgt}(\mathrm{pref}(\pi,\mathrm{n}))) = \infty$

Classification of paths

A path $\pi \in \mathrm{InfPaths}(\mathcal{M})$ is called

- pumping $:\Leftrightarrow \liminf_{n\to\infty}(\mathrm{wgt}(\mathrm{pref}(\pi,\mathrm{n}))) = \infty$
- $\begin{matrix}\text{(positively)}\\ \text{negatively}\end{matrix}$ weight divergent $:\Leftrightarrow \begin{matrix}\limsup_{n\to\infty}\\ \liminf_{n\to\infty}\end{matrix} = \begin{matrix}\infty\\ -\infty\end{matrix}$

Classification of paths

A path $\pi \in \mathrm{InfPaths}(\mathcal{M})$ is called

- pumping $:\Leftrightarrow \liminf_{n\to\infty}(\mathrm{wgt}(\mathrm{pref}(\pi, \mathrm{n}))) = \infty$
- $\begin{matrix}\text{(positively)}\\ \text{negatively}\end{matrix}$ weight divergent $:\Leftrightarrow \begin{matrix}\limsup_{n\to\infty}\\ \liminf_{n\to\infty}\end{matrix} = \begin{matrix}\infty\\ -\infty\end{matrix}$
- gambling $:\Leftrightarrow \pi$ is positively and negatively weight divergent

Classification of paths

A path $\pi \in \mathrm{InfPaths}(\mathcal{M})$ is called

- pumping $:\Leftrightarrow \liminf_{n\to\infty}(\mathrm{wgt}(\mathrm{pref}(\pi, \mathrm{n}))) = \infty$
- $\begin{array}{c}\text{(positively)}\\ \text{negatively}\end{array}$ weight divergent $:\Leftrightarrow \begin{array}{c}\limsup_{n\to\infty} \\ \liminf_{n\to\infty}\end{array} = \begin{array}{c}\infty \\ -\infty\end{array}$
- gambling $:\Leftrightarrow$ π is positively and negatively weight divergent
- bounded from below $:\Leftrightarrow \liminf_{n\to\infty} \mathrm{wgt}(\mathrm{pref}(\pi, n)) \in \mathbb{Z}$

Classification of end components

We distinguish end components by the following types

- pumping ECs: $\exists$ scheduler $\mathfrak{S}$: $\mathbb{Pr}(\pi \text{ is pumping}) = 1$

Classification of end components

We distinguish end components by the following types

- pumping ECs: $\exists$ scheduler $\mathfrak{S} : \mathbb{Pr}(\pi \text{ is pumping}) = 1$
- $\begin{matrix}\text{(positively)}\\ \text{negatively}\end{matrix}$ weight divergent ECs:

$$\exists \text{ scheduler } \mathfrak{S} : \mathbb{Pr}\left(\pi \text{ is } \begin{matrix}\text{(positively)}\\ \text{negatively}\end{matrix} \text{ weight divergent}\right) = 1$$

Classification of end components

We distinguish end components by the following types

- pumping ECs: $\exists$ scheduler $\mathfrak{S}$: $\mathbb{Pr}(\pi \text{ is pumping}) = 1$
- $\begin{matrix}\text{(positively)}\\ \text{negatively}\end{matrix}$ weight divergent ECs:

$$\exists \text{ scheduler } \mathfrak{S} : \mathbb{Pr}\Big(\pi \text{ is } \begin{matrix}\text{(positively)}\\ \text{negatively}\end{matrix} \text{ weight divergent } \Big) = 1$$

- gambling ECs: There is a scheduler s.t. $\mathbb{E}(\mathrm{MP}) = 0$ and it is positively and negatively weight divergent

Classification of end components

We distinguish end components by the following types

- pumping ECs: $\exists$ scheduler $\mathfrak{S} : \mathbb{Pr}(\pi \text{ is pumping}) = 1$
- $\begin{array}{c}\text{(positively)}\\ \text{negatively}\end{array}$ weight divergent ECs:

$$\exists \text{ scheduler } \mathfrak{S} : \mathbb{Pr}\left(\pi \text{ is } \begin{array}{c}\text{(positively)}\\ \text{negatively}\end{array} \text{ weight divergent }\right) = 1$$

- gambling ECs: There is a scheduler s.t. $\mathbb{E}(\mathrm{MP}) = 0$ and it is positively and negatively weight divergent
- bounded EC: There exists an upper bound and a lower bound

Check Weight-Divergence of a SCC

- can be done in **PTime**

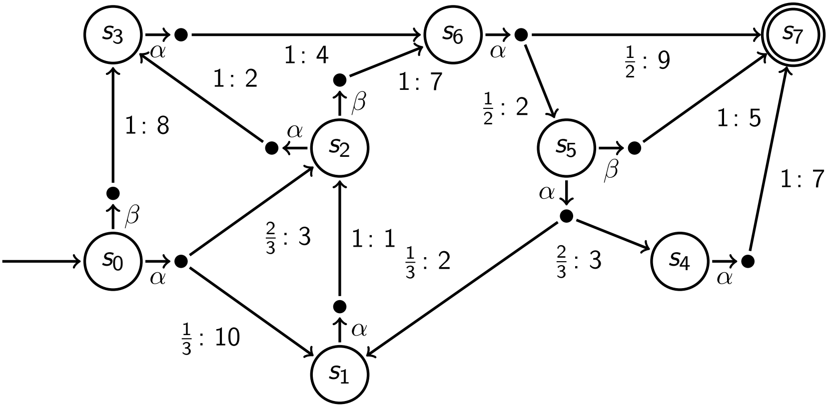
Check Weight-Divergence of a SCC

- can be done in **PTime**
- two cases:
  - "yes, weight-divergent": finds a gambling or a pumping scheduler

Check Weight-Divergence of a SCC

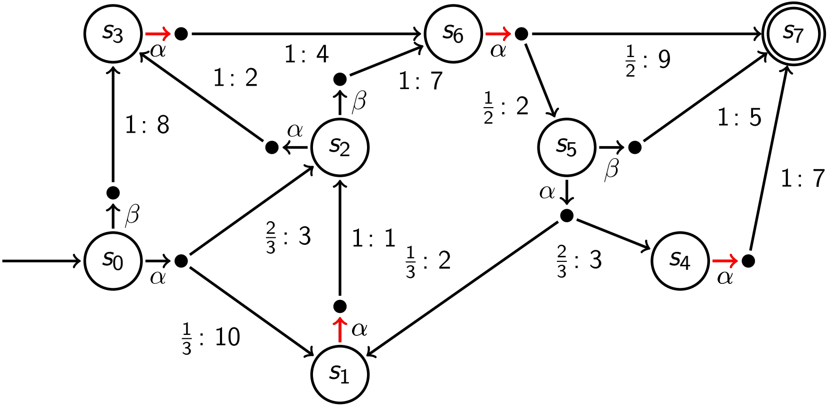
- can be done in **PTime**
- two cases:
  - "yes, weight-divergent": finds a gambling or a pumping scheduler
  - "no": returns an *equivalent* MDP $\mathcal{N}$ without 0-ECs

- The *stochastic* shortest path problem



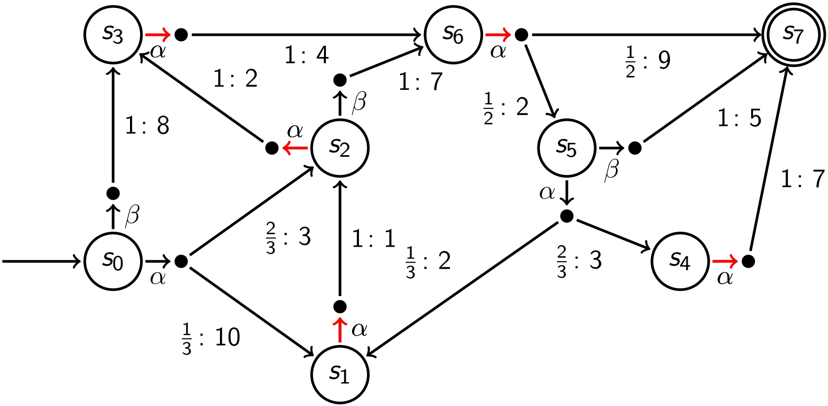
- Markov Decision Process (MDP)
- Task
  - Give a strategy to reach the goal with minimal *expected* accumulated weights!

- The *stochastic* shortest path problem



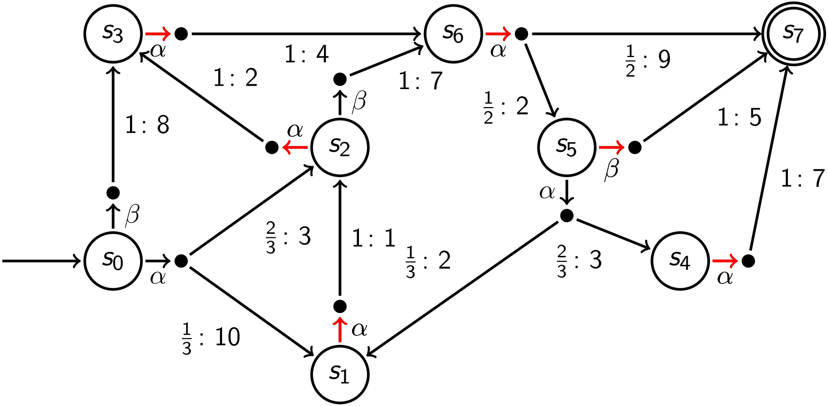
- Markov Decision Process (MDP)
- Task
  - Give a strategy to reach the goal with minimal *expected* accumulated weights!

- The *stochastic* shortest path problem



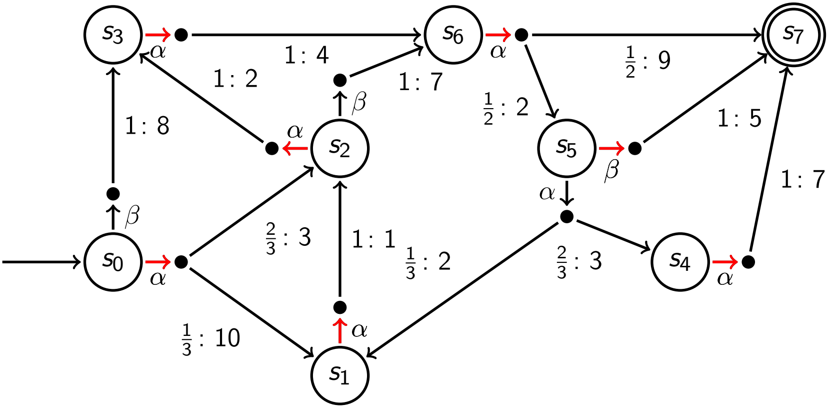
- Markov Decision Process (MDP)
- Task
    - Give a strategy to reach the goal with minimal *expected* accumulated weights!

- The *stochastic* shortest path problem



- Markov Decision Process (MDP)
- Task
  - Give a strategy to reach the goal with minimal *expected* accumulated weights!

- The *stochastic* shortest path problem



- Markov Decision Process (MDP)
- Task
  - Give a strategy to reach the goal with minimal *expected* accumulated weights!

Section 3

Different variants of the stochastic shortest path problem

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1
- ...having non-negative integer weights

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1
- ...having non-negative integer weights
- ...having two sets of states $F, G \subseteq S$

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1
- ...having non-negative integer weights
- ...having two sets of states $F, G \subseteq S$

Maximizing the conditional expected accumulated weight

$$\mathbb{CE} := \mathbb{E}(\oplus \mathrm{goal} \mid \Diamond \mathrm{goal})$$

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1
- ...having non-negative integer weights
- ...having two sets of states $F, G \subseteq S$

Maximizing the conditional expected accumulated weight

$$\mathbb{CE} := \mathbb{E}(\oplus\text{goal} \mid \Diamond\text{goal})$$

$$\mathbb{CE} := \mathbb{E}(\oplus F \mid \Diamond G)$$

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1
- ...having non-negative integer weights
- ...having two sets of states $F, G \subseteq S$

Maximizing the conditional expected accumulated weight

$$\mathbb{CE} := \mathbb{E}(\oplus\mathrm{goal} \mid \Diamond\mathrm{goal})$$

$$\mathbb{CE} := \mathbb{E}(\oplus F \mid \Diamond G)$$

$$\mathbb{CE}^{max} := \sup_{\mathfrak{S}\in\mathfrak{V}} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\oplus F \mid \Diamond G)$$

conditional expected accumulated weights

Assume:

- ...to not reach goal with probability 1
- ...having non-negative integer weights
- ...having two sets of states $F, G \subseteq S$

Maximizing the conditional expected accumulated weight

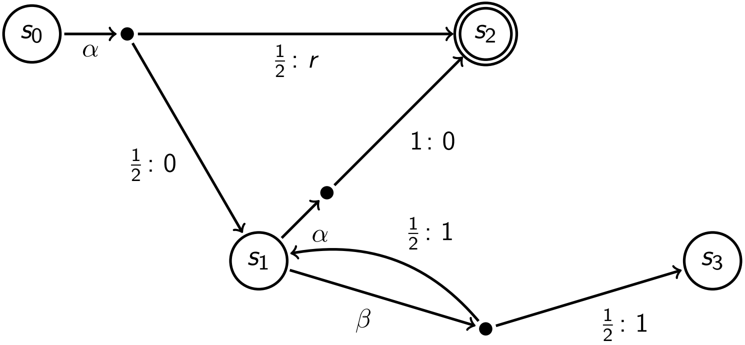
$$\mathbb{CE} := \mathbb{E}(\oplus\mathrm{goal} \mid \Diamond\mathrm{goal})$$

$$\mathbb{CE} := \mathbb{E}(\oplus F \mid \Diamond G)$$

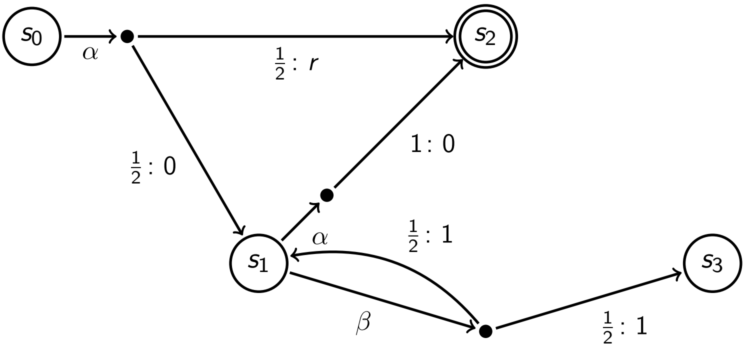
$$\mathbb{CE}^{max} := \sup_{\mathfrak{S} \in \mathfrak{V}} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M}, s_{init}}(\oplus F \mid \Diamond G)$$

$$\mathfrak{V} := \{\mathfrak{S} \in \mathrm{Schedulers}(\mathcal{M}) \mid \mathbb{Pr}^{\mathfrak{S}}_{\mathcal{M}, s_{init}}(\Diamond G) > 0 \text{ and } \mathbb{Pr}^{\mathfrak{S}}_{\mathcal{M}, s_{init}}(\Diamond F \mid \Diamond G) = 1\}$$

conditional expected accumulated reward



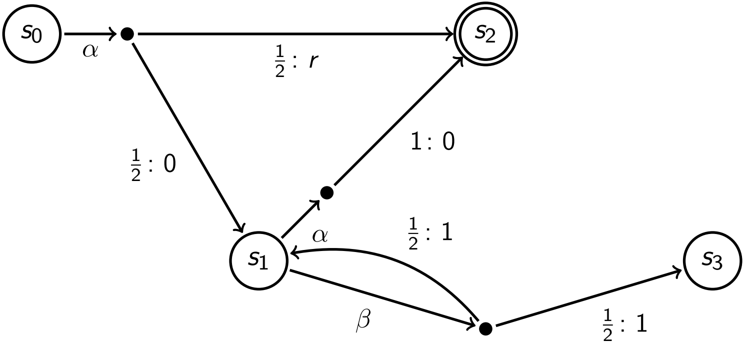
conditional expected accumulated reward



Finding the best scheduler...

$\mathfrak{S}_n$: select $\beta^n\alpha$, $n \in \mathbb{N} \cup \{\infty\}$

conditional expected accumulated reward



Finding the best scheduler...

$\mathfrak{S}_n$: select $\beta^n\alpha$, $n \in \mathbb{N} \cup \{\infty\}$

$$\mathbb{CE} = \frac{\frac{r}{2} + \frac{1}{2} \cdot \frac{1}{2}^n \cdot n}{\frac{1}{2} + \frac{1}{2}^{n+1}}$$

choose scheduler $\mathfrak{S}_{r+2}$

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`
- There is a **pseudo-PTime algorithm** to `calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`
- There is a **pseudo-PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- *If we have* $F = G$ *and* $\forall s \in \mathrm{States}(\mathcal{M}) : s \vDash \exists \Diamond G \Rightarrow \mathbb{Pr}^{min}_{\mathcal{M},s}(\Diamond G) > 0$ there is a **PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`
- There is a **pseudo-PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- *If we have $F = G$ and* $\forall s \in \mathrm{States}(\mathcal{M}) : s \vDash \exists\Diamond G \Rightarrow \mathbb{P}\mathrm{r}^{min}_{\mathcal{M},s}(\Diamond G) > 0$ there is a **PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- The problem `Decide if` $\mathbb{CE}^{max} \bowtie t$ where we have
  - $t \in \mathbb{Q} \ldots$ some rational threshold

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`
- There is a **pseudo-PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- *If we have* $F = G$ *and* $\forall s \in \mathrm{States}(\mathcal{M}) : s \vDash \exists\Diamond G \Rightarrow \mathbb{P}\mathrm{r}^{min}_{\mathcal{M},s}(\Diamond G) > 0$ there is a **PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- The problem `Decide if` $\mathbb{CE}^{max} \bowtie t$ where we have
  - $t \in \mathbb{Q} \ldots$ some rational threshold
  - $\bowtie \in \{<, \leq, \geq, >\}$

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`
- There is a **pseudo-PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- *If we have $F = G$ and* $\forall s \in \mathrm{States}(\mathcal{M}) : s \models \exists \Diamond G \Rightarrow \mathbb{Pr}^{min}_{\mathcal{M},s}(\Diamond G) > 0$ there is a **PTime algorithm** `to calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- The problem `Decide if` $\mathbb{CE}^{max} \bowtie t$ where we have
  - $t \in \mathbb{Q} \ldots$ some rational threshold
  - $\bowtie \in \{<, \leq, \geq, >\}$

  is **PSpace-hard**, solvable in **ExpTime** and *for acyclic MDPs* **PSpace-complete**

complexity: conditional expected accumulated rewards

- There is a **PTime algorithm** to decide: `Is` $\mathbb{CE}^{max}$ `finite?`
- There is a **pseudo-PTime algorithm** to `calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- *If we have* $F = G$ *and* $\forall s \in \mathrm{States}(\mathcal{M}) : s \models \exists \Diamond G \Rightarrow \mathbb{P}\mathrm{r}^{min}_{\mathcal{M},s}(\Diamond G) > 0$ there is a **PTime algorithm** to `calculate an upperbound` $\mathbb{CE}^{ub} \geq \mathbb{CE}^{max}$
- The problem `Decide if` $\mathbb{CE}^{max} \bowtie t$ where we have
  - $t \in \mathbb{Q} \ldots$ some rational threshold
  - $\bowtie \in \{<, \leq, \geq, >\}$

  is **PSpace-hard**, solvable in **ExpTime** and *for acyclic MDPs* **PSpace-complete**
- In **ExpTime** we can compute $\mathbb{CE}^{max}$ `together with an optimal scheduler`

Checking finiteness

- **PTime** algorithm: given $\mathcal{M}, F, G$, two possible outcomes:
  (1) $\mathbb{CE}^{max} = \infty$
  (2) $\mathbb{CE}^{max}$ is finite, equivalent MDP $\mathcal{N}$ with two trap states *goal*, *fail*

Checking finiteness

- **PTime** algorithm: given $\mathcal{M}, F, G$, two possible outcomes:
  (1) $\mathbb{CE}^{max} = \infty$
  (2) $\mathbb{CE}^{max}$ is finite, equivalent MDP $\mathcal{N}$ with two trap states *goal*, *fail*

equivalence of $\mathcal{M}$ and $\mathcal{N}$, properties of $\mathcal{N}$

$$\mathbb{E}^{max}_{\mathcal{M},s_{init}}(\oplus F \mid \Diamond G) = \mathbb{E}^{max}_{\mathcal{N},s_{init}}(\oplus goal \mid \Diamond goal)$$

Checking finiteness

- **PTime** algorithm: given $\mathcal{M}, F, G$, two possible outcomes:
  (1) $\mathbb{CE}^{max} = \infty$
  (2) $\mathbb{CE}^{max}$ is finite, equivalent MDP $\mathcal{N}$ with two trap states $goal, fail$

equivalence of $\mathcal{M}$ and $\mathcal{N}$, properties of $\mathcal{N}$

$$\mathbb{E}^{max}_{\mathcal{M},s_{init}}(\oplus F \mid \Diamond G) = \mathbb{E}^{max}_{\mathcal{N},s_{init}}(\oplus goal \mid \Diamond goal)$$

in $\mathcal{N}$ the state $goal$ is reachable from all states $s \in S_{\mathcal{N}} \setminus \{fail\}$ and

$$\mathbb{Pr}^{min}_{\mathcal{N},s}\Big(\Diamond(goal \vee fail)\Big) = 1$$

Checking finiteness

- **PTime** algorithm: given $\mathcal{M}, F, G$, two possible outcomes:
  (1) $\mathbb{CE}^{max} = \infty$
  (2) $\mathbb{CE}^{max}$ is finite, equivalent MDP $\mathcal{N}$ with two trap states *goal*, *fail*

equivalence of $\mathcal{M}$ and $\mathcal{N}$, properties of $\mathcal{N}$

$$\mathbb{E}^{max}_{\mathcal{M},s_{init}}(\oplus F \mid \Diamond G) = \mathbb{E}^{max}_{\mathcal{N},s_{init}}(\oplus goal \mid \Diamond goal)$$

in $\mathcal{N}$ the state *goal* is reachable from all states $s \in S_{\mathcal{N}} \setminus \{fail\}$ and

$$\mathbb{Pr}^{min}_{\mathcal{N},s}\Big(\Diamond(goal \vee fail)\Big) = 1$$

$\mathcal{N}$ has no *critical* scheduler $\mathfrak{S}$: $\mathbb{Pr}^{\mathfrak{S}}(\Diamond fail) = 1$ and there is a reachable positive $\mathfrak{S}$-cycle.

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Observation

- $\exists$ saturation point $t \in \mathbb{N}$ such that

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Observation

- $\exists$ saturation point $t \in \mathbb{N}$ such that
  - after π with $\mathrm{wgt}(\pi) \geq t$ we can rely on a memoryless, deterministic scheduler maximizing the probability to reach *goal*.

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Observation

- $\exists$ saturation point $t \in \mathbb{N}$ such that
  - after π with $\mathrm{wgt}(\pi) \geq t$ we can rely on a memoryless, deterministic scheduler maximizing the probability to reach *goal*.
  - Until reaching t a deterministic reward-based scheduler is sufficient.

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Observation

- $\exists$ saturation point $t \in \mathbb{N}$ such that
  - after π with $\mathrm{wgt}(\pi) \geq t$ we can rely on a memoryless, deterministic scheduler maximizing the probability to reach *goal*.
  - Until reaching t a deterministic reward-based scheduler is sufficient.

Threshold Algorithm

- input: MDP $\mathcal{N}$ as before, threshold $t \in \mathbb{Q}_{\geq 0}$

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Observation

- $\exists$ saturation point $t \in \mathbb{N}$ such that
  - after π with $\mathrm{wgt}(\pi) \geq t$ we can rely on a memoryless, deterministic scheduler maximizing the probability to reach *goal*.
  - Until reaching t a deterministic reward-based scheduler is sufficient.

Threshold Algorithm

- input: MDP $\mathcal{N}$ as before, threshold $t \in \mathbb{Q}_{\geq 0}$
- output:

case (1) "no", we do not have $\mathbb{CE}^{max} > t$
case (2) "yes", $\mathbb{CE}^{max} > t$ and we found a deterministic, reward-based scheduler $\mathfrak{S}$ s.t.

Threshold Algorithm

We assume that we have such an MDP $\mathcal{N}$.

Observation

- $\exists$ saturation point $t \in \mathbb{N}$ such that
  - after π with $\mathrm{wgt}(\pi) \geq t$ we can rely on a memoryless, deterministic scheduler maximizing the probability to reach *goal*.
  - Until reaching t a deterministic reward-based scheduler is sufficient.

Threshold Algorithm

- input: MDP $\mathcal{N}$ as before, threshold $t \in \mathbb{Q}_{\geq 0}$
- output:

case (1) "no", we do not have $\mathbb{CE}^{max} > t$

case (2) "yes", $\mathbb{CE}^{max} > t$ and we found a deterministic, reward-based scheduler $\mathfrak{S}$ s.t. $\mathfrak{S}$ is memoryless after some saturation point.

What about paths not reaching *goal*?

conditional expectation	partial expectation

What about paths not reaching *goal*?

conditional expectation	partial expectation
$\mathbb{CE}$	$\mathbb{PE}$

What about paths not reaching *goal*?

conditional expectation	partial expectation
$\mathbb{CE}$	$\mathbb{PE}$
$\mathbb{CE} = \mathbb{E}(\oplus\text{goal} \mid \Diamond\text{goal})$	$\pi \not\models \Diamond\text{goal} \Rightarrow \text{wgt}'(\pi) := 0$

What about paths not reaching *goal*?

$$\oplus goal(\pi) := \begin{cases} \oplus(\pi), & \pi \models \Diamond F \\ 0, & \text{otherwise, i.e. } \pi \not\models \Diamond F \end{cases}$$

conditional expectation	partial expectation
$\mathbb{CE}$	$\mathbb{PE}$
$\mathbb{CE} = \mathbb{E}(\oplus\mathrm{goal} \mid \Diamond\mathrm{goal})$	$\pi \not\models \Diamond\mathrm{goal} \Rightarrow \mathrm{wgt}'(\pi) := 0$

What about paths not reaching *goal*?

$$\oplus goal(\pi) := \begin{cases} \oplus(\pi), & \pi \models \Diamond F \\ 0, & \text{otherwise, i.e. } \pi \not\models \Diamond F \end{cases}$$

conditional expectation	partial expectation
$\mathbb{CE}$	$\mathbb{PE}$
$\mathbb{CE} = \mathbb{E}(\oplus\mathrm{goal} \mid \Diamond\mathrm{goal})$	$\pi \not\models \Diamond\mathrm{goal} \Rightarrow \mathrm{wgt}'(\pi) := 0$
may lead to quite high $\mathbb{CE}$ paired with a low probability of reaching goal	good approximation for maximizing probability of reaching goal and reward until goal

What about paths not reaching *goal*?

$$\oplus goal(\pi) := \begin{cases} \oplus(\pi), & \pi \models \Diamond F \\ 0, & \text{otherwise, i.e. } \pi \not\models \Diamond F \end{cases}$$

conditional expectation	partial expectation
$\mathbb{CE}$	$\mathbb{PE}$
$\mathbb{CE} = \mathbb{E}(\oplus\mathrm{goal} \mid \Diamond\mathrm{goal})$	$\pi \not\models \Diamond\mathrm{goal} \Rightarrow \mathrm{wgt}'(\pi) := 0$
may lead to quite high $\mathbb{CE}$ paired with a low probability of reaching goal	good approximation for maximizing probability of reaching goal and reward until goal

For both we know...

- **PTime** algorithm to check finiteness of $\mathbb{CE}^{max}$ ($\mathbb{PE}^{max}$)

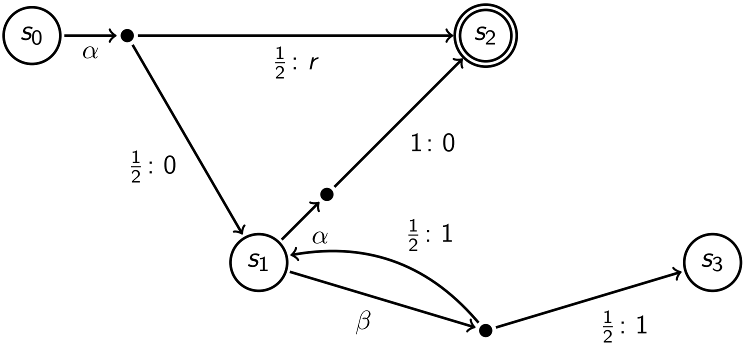
What about paths not reaching *goal*?

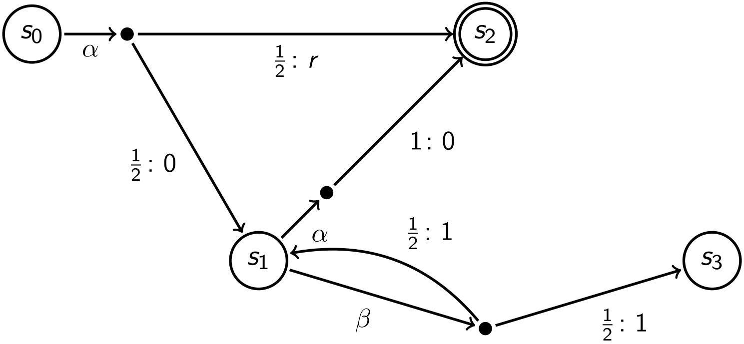
$$\oplus goal(\pi) := \begin{cases} \oplus(\pi), & \pi \models \Diamond F \\ 0, & \text{otherwise, i.e. } \pi \not\models \Diamond F \end{cases}$$

conditional expectation	partial expectation
$\mathbb{CE}$	$\mathbb{PE}$
$\mathbb{CE} = \mathbb{E}(\oplus \mathrm{goal} \mid \Diamond \mathrm{goal})$	$\pi \not\models \Diamond \mathrm{goal} \Rightarrow \mathrm{wgt}'(\pi) := 0$
may lead to quite high $\mathbb{CE}$ paired with a low probability of reaching goal	good approximation for maximizing probability of reaching goal and reward until goal

For both we know...

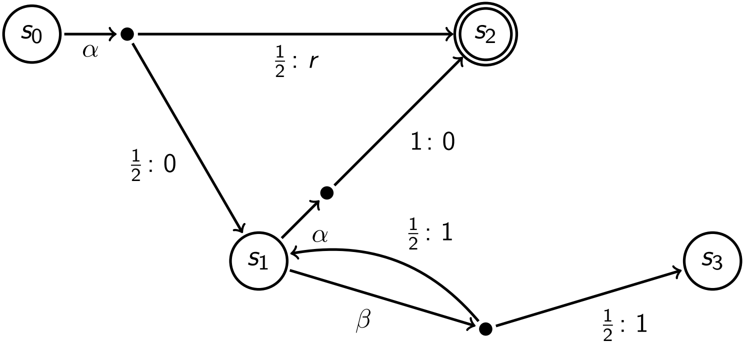
- **PTime** algorithm to check finiteness of $\mathbb{CE}^{max}$ ($\mathbb{PE}^{max}$)
- Both have a saturation point: reward-based schedulers vs. memoryless, det. schedulers


s_0
α
$\frac{1}{2}: r$
s_2
$\frac{1}{2}: 0$
$1: 0$
α
$\frac{1}{2}: 1$
s_1
s_3
β
$\frac{1}{2}: 1$



Finding the best scheduler...

$\mathfrak{S}_n$: select $\beta^n\alpha$, $n \in \mathbb{N} \cup \{\infty\}$

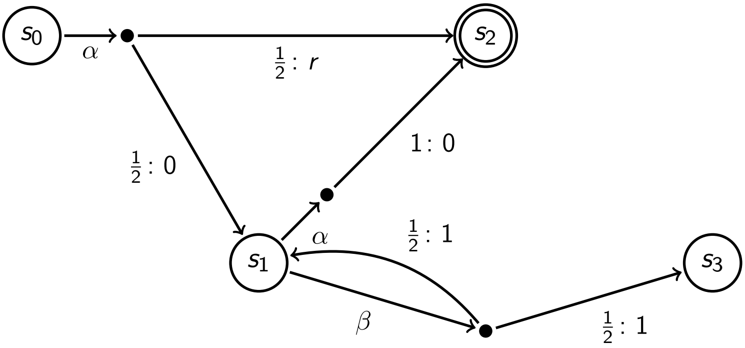


Finding the best scheduler...

$\mathfrak{S}_n$: select $\beta^n \alpha$, $n \in \mathbb{N} \cup \{\infty\}$

$$\mathbb{CE} = \frac{\frac{r}{2} + \frac{1}{2} \cdot \frac{1}{2}^{n} \cdot n}{\frac{1}{2} + \frac{1}{2}^{n+1}}$$

choose scheduler $\mathfrak{S}_{r+2}$



Finding the best scheduler...

$\mathfrak{S}_n$: select $\beta^n\alpha$, $n \in \mathbb{N} \cup \{\infty\}$

$$\mathbb{CE} = \frac{\frac{r}{2} + \frac{1}{2} \cdot \frac{1}{2}^n \cdot n}{\frac{1}{2} + \frac{1}{2}^{n+1}}$$

choose scheduler $\mathfrak{S}_{r+2}$

$$\mathbb{PE} = \frac{1}{2}r + \frac{1}{2} \cdot \frac{1}{2}^n \cdot n$$

choose scheduler $\mathfrak{S}_1$ or $\mathfrak{S}_2$

Switching to integer weights...

- It's not ensured that a saturation point exists

Switching to integer weights...

- It's not ensured that a saturation point exists
- Optimal schedulers may need infinite memory

Switching to integer weights...

- It's not ensured that a saturation point exists
- Optimal schedulers may need infinite memory
- Optimal values may even become irrational

Switching to integer weights...

- It's not ensured that a saturation point exists
- Optimal schedulers may need infinite memory
- Optimal values may even become irrational
- LPs are not anymore sufficient for solving

Switching to integer weights...

- It's not ensured that a saturation point exists
- Optimal schedulers may need infinite memory
- Optimal values may even become irrational
- LPs are not anymore sufficient for solving
- There are approximation methods using LPs

Section 4

Keep an eye on the variance

Variance-penalized expectation

Given:

- MDP $\mathcal{M}$ with non-negative integer weights

Variance-penalized expectation

Given:

- MDP $\mathcal{M}$ with non-negative integer weights
- only one trap state *goal* which is reachable from all other states

Variance-penalized expectation

Given:

- MDP $\mathcal{M}$ with non-negative integer weights
- only one trap state *goal* which is reachable from all other states
- all states are reachable from s_{init}

Variance-penalized expectation

Given:

- MDP $\mathcal{M}$ with non-negative integer weights
- only one trap state *goal* which is reachable from all other states
- all states are reachable from s_{init}

Variance-penalized expectation

$$\mathbb{VPE}[\lambda]^{\mathfrak{S}}_{\mathcal{M}} := \mathbb{E}^{\mathfrak{S}}_{\mathcal{M}}(\oplus goal) - \lambda \cdot \mathbb{V}^{\mathfrak{S}}_{\mathcal{M}}(\oplus goal)$$

Variance-penalized expectation

Given:

- MDP $\mathcal{M}$ with non-negative integer weights
- only one trap state *goal* which is reachable from all other states
- all states are reachable from s_{init}

Variance-penalized expectation

$$\mathbb{VPE}[\lambda]^{\mathfrak{S}}_{\mathcal{M}} := \mathbb{E}^{\mathfrak{S}}_{\mathcal{M}}(\oplus goal) - \lambda \cdot \mathbb{V}^{\mathfrak{S}}_{\mathcal{M}}(\oplus goal)$$

$$\mathbb{VPE}[\lambda]^{max}_{\mathcal{M}} := \sup_{\mathfrak{S}} \mathbb{VPE}[\lambda]^{\mathfrak{S}}_{\mathcal{M}}$$

complexity result

- Computing a variance-minimal scheduler among all $\mathbb{E}$-optimal schedulers is doable in **PTime**. The scheduler can be chosen memoryless.

complexity result

- Computing a variance-minimal scheduler among all $\mathbb{E}$-optimal schedulers is doable in **PTime**. The scheduler can be chosen memoryless.
- In **ExpSpace** we can compute $\mathbb{VPE}^{max}$ and a deterministic scheduler as witness.

complexity result

- Computing a variance-minimal scheduler among all $\mathbb{E}$-optimal schedulers is doable in **PTime**. The scheduler can be chosen memoryless.
- In **ExpSpace** we can compute $\mathbb{VPE}^{max}$ and a deterministic scheduler as witness.
- The threshold problem, i.e. checking $\mathbb{VPE}^{max} \geq t$ is decidable in **NExpTime** and is known to be **ExpTime-hard**.

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that

PTime algorithm for chosing the best$_{(\mathit{variance})}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved
2. Transformation $\mathcal{M}' \mapsto \mathcal{M}''$ such that

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved
2. Transformation $\mathcal{M}' \mapsto \mathcal{M}''$ such that
   - All actions not leading to $\mathbb{E}^{max}$ are removed.

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved
2. Transformation $\mathcal{M}' \mapsto \mathcal{M}''$ such that
   - All actions not leading to $\mathbb{E}^{max}$ are removed.
   - $\mathcal{M}''$ has no end components

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved
2. Transformation $\mathcal{M}' \mapsto \mathcal{M}''$ such that
   - All actions not leading to $\mathbb{E}^{max}$ are removed.
   - $\mathcal{M}''$ has no end components
   - All schedulers of $\mathcal{M}''$ have equal value for $\mathbb{E}$.

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved
2. Transformation $\mathcal{M}' \mapsto \mathcal{M}''$ such that
   - All actions not leading to $\mathbb{E}^{max}$ are removed.
   - $\mathcal{M}''$ has no end components
   - All schedulers of $\mathcal{M}''$ have equal value for $\mathbb{E}$.
3. Solve a system of linear equations to find $\mathbb{V}^{min}$ + witnessing scheduler

PTime algorithm for chosing the best$_{(variance)}$ scheduler among all $\mathbb{E}$-optimal ones.

1. PTime transformation $\mathcal{M} \mapsto \mathcal{M}'$ such that
   - $\mathcal{M}'$ has no 0-ECs
   - There are mappings from schedulers of $\mathcal{M}$ to $\mathcal{M}'$ and vice versa s.t.
     - $\mathbb{E}, \mathbb{V}$ are preserved
2. Transformation $\mathcal{M}' \mapsto \mathcal{M}''$ such that
   - All actions not leading to $\mathbb{E}^{max}$ are removed.
   - $\mathcal{M}''$ has no end components
   - All schedulers of $\mathcal{M}''$ have equal value for $\mathbb{E}$.
3. Solve a system of linear equations to find $\mathbb{V}^{min}$ + witnessing scheduler
   - The scheduler can be chosen memoryless and deterministic

Applications

- Worst-case expected termination times of probabilistic programs

Applications

- Worst-case expected termination times of probabilistic programs
- Finding optimal controls for a motion planning scenario having random external influences

Applications

- Worst-case expected termination times of probabilistic programs
- Finding optimal controls for a motion planning scenario having random external influences
- Traffic control systems, energy grids

Applications

- Worst-case expected termination times of probabilistic programs
- Finding optimal controls for a motion planning scenario having random external influences
- Traffic control systems, energy grids
- Decision making in financial markets

Applications

- Worst-case expected termination times of probabilistic programs
- Finding optimal controls for a motion planning scenario having random external influences
- Traffic control systems, energy grids
- Decision making in financial markets

Limits of applicability?

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$
- weight threshold $t \in \mathcal{M}$

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$
- weight threshold $t \in \mathcal{M}$
- probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$
- weight threshold $t \in \mathcal{M}$
- probability threshold $\alpha \in [0,1] \cap \mathbb{Q}$

Decide: `Is there a scheduler such that` $\mathbb{Pr}(\bigoplus(\pi) \leq t) \geq \alpha$?

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$
- weight threshold $t \in \mathcal{M}$
- probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$

Decide: `Is there a scheduler such that` $\mathbb{P}\mathrm{r}(\oplus(\pi) \leq t) \geq \alpha$?

- ...**PSpace-hard**

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$
- weight threshold $t \in \mathcal{M}$
- probability threshold $\alpha \in [0,1] \cap \mathbb{Q}$

Decide: `Is there a scheduler such that` $\mathbb{P}\mathrm{r}(\oplus(\pi) \leq t) \geq \alpha$?

- ...**PSpace-hard**
- ...decidable in **pseudo-PTime**

Percentile queries

The problem

Given a

- $\mathbb{Z}$-weighted MDP $\mathcal{M}$
- weight threshold $t \in \mathcal{M}$
- probability threshold $\alpha \in [0, 1] \cap \mathbb{Q}$

Decide: `Is there a scheduler such that` $\mathbb{Pr}(\oplus(\pi) \leq t) \geq \alpha$?

- ...**PSpace-hard**
- ...decidable in **pseudo-PTime**
- ...There is an optimal deterministic scheduler, computable in **ExpTime**

Tools

- Storm model checker
- Prism model checker