Markov Decision Processes

Based on slides by Nils Jansen

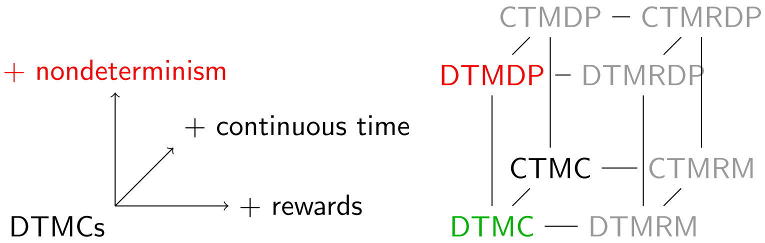
Eline Bovy

February 28, 2025

Markov Decision Processes

- Nondeterminism
- MDPs
- Schedulers
- Probabilistic Reachability
- Memoryless Schedulers Suffice
- Computing Reachability Probabilities

The probabilistic model space



DTMC	=	Discrete-time Markov chain
DTMRM	=	Discrete-time Markov reward model
DTMDP	=	Discrete-time Markov decision process
DTMRDP	=	Discrete-time Markow reward decision process
CTMC	=	Continuous-time Markov chain
CTMRM	=	Continuous-time Markov reward model
CTMDP	=	Continuous-time Markov decision process
CTMRDP	=	Continuous-time Markov reward decision process

Nondeterminism

Some aspects of a system may not be probabilistic and should not be modelled probabilistically, for example:

Nondeterminism

Some aspects of a system may not be probabilistic and should not be modelled probabilistically, for example:

- Concurrency – scheduling of parallel components
  - e. g., randomized distributed algorithms – multiple probabilistic components operating asynchronously

Nondeterminism

Some aspects of a system may not be probabilistic and should not be modelled probabilistically, for example:

- Concurrency – scheduling of parallel components
  - e. g., randomized distributed algorithms – multiple probabilistic components operating asynchronously
- Unknown environments
  - e. g., probabilistic security protocols – unknown adversary
  - e. g., partial information in reinforcement learning

Nondeterminism

Some aspects of a system may not be probabilistic and should not be modelled probabilistically, for example:

- Concurrency – scheduling of parallel components
  - e. g., randomized distributed algorithms – multiple probabilistic components operating asynchronously
- Unknown environments
  - e. g., probabilistic security protocols – unknown adversary
  - e. g., partial information in reinforcement learning
- Underspecification – unknown model parameters
  - e. g., a probabilistic communication protocol designed for message propagation delays of between $d_{\min}$ and $d_{\max}$.
  - e. g., not enough data to sufficiently describe behavior in a stochastic manner

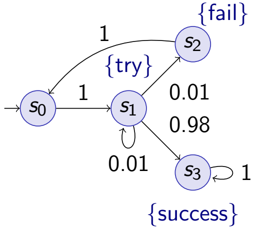
Nondeterminism

Some aspects of a system may not be probabilistic and should not be modelled probabilistically, for example:

- Concurrency – scheduling of parallel components
  - e. g., randomized distributed algorithms – multiple probabilistic components operating asynchronously
- Unknown environments
  - e. g., probabilistic security protocols – unknown adversary
  - e. g., partial information in reinforcement learning
- Underspecification – unknown model parameters
  - e. g., a probabilistic communication protocol designed for message propagation delays of between $d_{\min}$ and $d_{\max}$.
  - e. g., not enough data to sufficiently describe behavior in a stochastic manner
- Abstraction
  - e. g., partition a DTMC into similar (but not identical) states

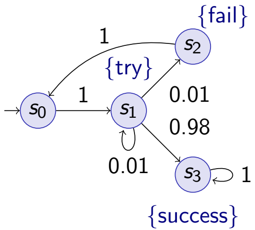
Probability vs. nondeterminism

- Discrete-time Markov chain
  - $(S, s_{\text{init}}, P, L)$ where $P : S \times S \to [0, 1]$
  - choice is probabilistic

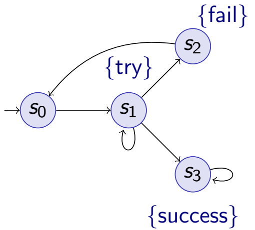


Probability vs. nondeterminism

- Discrete-time Markov chain
  - $(S, s_{\text{init}}, P, L)$ where $P : S \times S \to [0, 1]$
  - choice is probabilistic

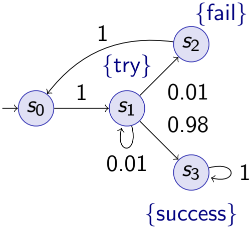


- Labeled transition system
  - $(S, s_{\text{init}}, R, L)$ where $R \subseteq S \times S$
  - choice is non-deterministic

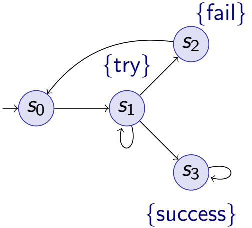


Probability vs. nondeterminism

- Discrete-time Markov chain
  - $(S, s_{\text{init}}, P, L)$ where $P : S \times S \to [0, 1]$
  - choice is probabilistic



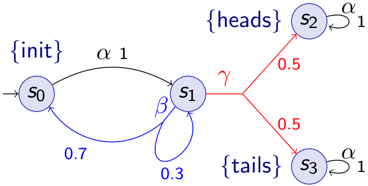
- Labeled transition system
  - $(S, s_{\text{init}}, R, L)$ where $R \subseteq S \times S$
  - choice is non-deterministic



- How to combine?

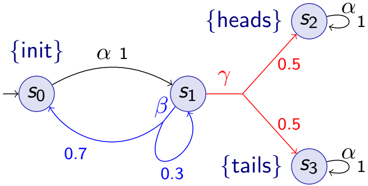
Markov decision processes

- Markov decision processes (MDPs)
  - extension of DTMCs with nondeterministic choices



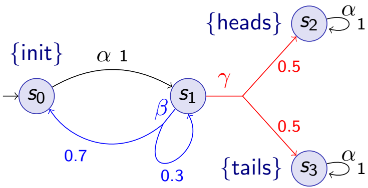
Markov decision processes

- Markov decision processes (MDPs)
  - extension of DTMCs with nondeterministic choices
- Like DTMCs
  - discrete set of states representing possible configurations of the system being modelled
  - transitions between states occur in discrete time steps



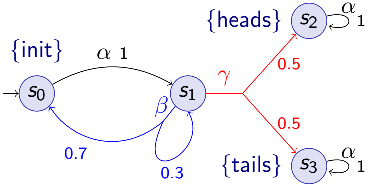
Markov decision processes

- Markov decision processes (MDPs)
  - extension of DTMCs with nondeterministic choices
- Like DTMCs
  - discrete set of states representing possible configurations of the system being modelled
  - transitions between states occur in discrete time steps
- Probabilities and nondeterminism
  - In each state, a nondeterministic choice between several discrete probability distributions over successor states is made.



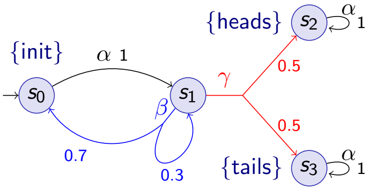
Markov decision processes

- An MDP M is a tuple $(S, s_{\mathsf{init}}, Act, P, L)$ where:
  - S is a non-empty set of states ("state space"),
  - $s_{\mathsf{init}} \in S$ is the initial state,
  - Act is a set of actions,
  - $P : S \times Act \times S \rightarrow [0,1]$ is the transition probability function, where:
  $$\forall s \in S, \forall \alpha \in Act : \sum_{s' \in S} P(s, \alpha, s') \in \{0,1\},$$
  - $L : S \rightarrow 2^{AP}$ is a labeling with atomic propositions.



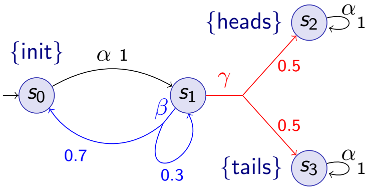
Markov decision processes

- A finite MDP M is a tuple $(S, s_{\text{init}}, Act, P, L)$ where:
  - S is a finite non-empty set of states ("state space"),
  - $s_{\text{init}} \in S$ is the initial state,
  - Act is a finite set of actions,
  - $P : S \times Act \times S \to [0, 1]$ is the transition probability function, where:
  $$\forall s \in S, \forall \alpha \in Act : \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\},$$
  - $L : S \to 2^{AP}$ is a labeling with atomic propositions (finite set).



Markov decision processes

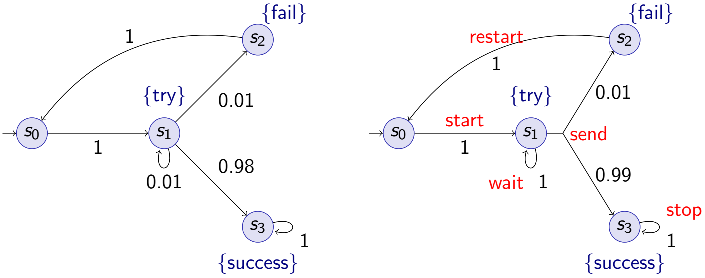
- A finite MDP M is a tuple $(S, s_{\text{init}}, Act, P, L)$ where:
  - S is a finite non-empty set of states ("state space"),
  - $s_{\text{init}} \in S$ is the initial state,
  - Act is a finite set of actions,
  - $P : S \times Act \times S \to [0,1]$ is the transition probability function, where:
$$\forall s \in S, \forall \alpha \in Act : \sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\},$$
  - $L : S \to 2^{AP}$ is a labeling with atomic propositions (finite set).
- Notes:
  - an action α is enabled in a state s iff $\sum_{s' \in S} P(s, \alpha, s') = 1$.
  - $Act(s) \subseteq Act$ denotes the non-empty set of enabled actions in s.



Simple MDP example 1

Modification of the simple DTMC communication protocol

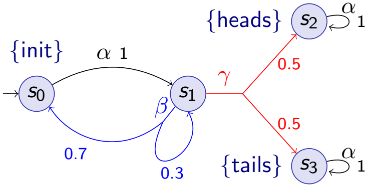
- after one step, process starts trying to send a message
- then, a nondeterministic choice between (a) waiting a step because the channel is unready, and (b) sending the message
- if the latter, with probability 0.99 send successfully and stop
- and with probability 0.01, message sending fails, restart.



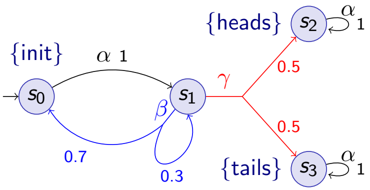
Simple MDP example 2

Another simple MDP example with four states

- from state s_0, move directly to state s_1 (action α)
- in state s_1, nondeterministic choice between actions β and γ.
- action β gives probabilistic choice: self-loop or return to s_0
- action γ gives a 0.5/0.5 random choice between heads/tails.

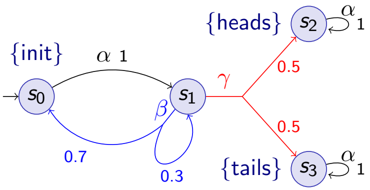


Simple MDP example 2



$$M = (S, s_{\text{init}}, Act, P, L) \quad AP = \{\text{init}, \text{heads}, \text{tails}\}$$

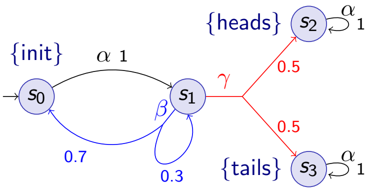
Simple MDP example 2



$$M = (S, s_{\text{init}}, Act, P, L) \quad AP = \{\text{init}, \text{heads}, \text{tails}\}$$

$$S = \{s_0, s_1, s_2, s_3\}$$
$$s_{\text{init}} = s_0$$
$$Act = \{\alpha, \beta, \gamma\}$$

Simple MDP example 2

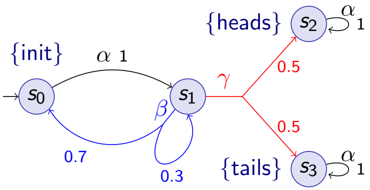


$M = (S, s_{\text{init}}, Act, P, L) \quad AP = \{\text{init}, \text{heads}, \text{tails}\}$

$S = \{s_0, s_1, s_2, s_3\}$
$s_{\text{init}} = s_0$
$Act = \{\alpha, \beta, \gamma\}$

$P(s_0, \alpha) = [s_1 \mapsto 1]$
$P(s_1, \beta) = [s_0 \mapsto 0.7, s_1 \mapsto 0.3]$
$P(s_1, \gamma) = [s_2 \mapsto 0.5, s_3 \mapsto 0.5]$
$P(s_2, \alpha) = [s_2 \mapsto 1]$
$P(s_3, \alpha) = [s_3 \mapsto 1]$

Simple MDP example 2



$M = (S, s_{\text{init}}, Act, P, L) \quad AP = \{\text{init}, \text{heads}, \text{tails}\}$

$S = \{s_0, s_1, s_2, s_3\}$

$s_{\text{init}} = s_0$

$Act = \{\alpha, \beta, \gamma\}$

$L(s_0) = \{\text{init}\}$

$L(s_1) = \emptyset$

$L(s_2) = \{\text{heads}\}$

$L(s_3) = \{\text{tails}\}$

$P(s_0, \alpha) = [s_1 \mapsto 1]$

$P(s_1, \beta) = [s_0 \mapsto 0.7, s_1 \mapsto 0.3]$

$P(s_1, \gamma) = [s_2 \mapsto 0.5, s_3 \mapsto 0.5]$

$P(s_2, \alpha) = [s_2 \mapsto 1]$

$P(s_3, \alpha) = [s_3 \mapsto 1]$

MDPs are compositional

- Compositionality: Combine MDPs for small components into an MDP for the whole system.
- Communication: between components via synchronization
- Synchronization: Involved components execute the same action simultaneously
- Non-synchronized actions are executed in an interleaved way.

Heavily exploited in PRISM's input language (details later).

Paths and probabilities

A (finite or infinite) path through an MDP

- is a sequence of states and actions,
- e. g., $s_0\alpha_0 s_1\alpha_1 s_2 \ldots$,
- such that $P(s_i, \alpha_i, s_{i+1}) > 0$ for all $i \geq 0$.

Paths and probabilities

A (finite or infinite) path through an MDP

- is a sequence of states and actions,
- e. g., $s_0\alpha_0 s_1\alpha_1 s_2 \ldots$,
- such that $P(s_i, \alpha_i, s_{i+1}) > 0$ for all $i \geq 0$.

A path represents an execution (i. e., one possible behavior) of the system which the MDP is modelling.

Paths and probabilities

A (finite or infinite) path through an MDP

- is a sequence of states and actions,
- e. g., $s_0\alpha_0 s_1\alpha_1 s_2\ldots$,
- such that $P(s_i, \alpha_i, s_{i+1}) > 0$ for all $i \geq 0$.

A path represents an execution (i. e., one possible behavior) of the system which the MDP is modelling.

Notation:

- $\mathsf{Paths}_{\mathsf{inf}}(s)$ = set of all infinite paths through the MDP starting in state s.
- $\mathsf{Paths}_{\mathsf{fin}}(s)$ = set of all finite paths from s.

Paths and probabilities

A (finite or infinite) path through an MDP

- is a sequence of states and actions,
- e. g., $s_0 \alpha_0 s_1 \alpha_1 s_2 \ldots$,
- such that $P(s_i, \alpha_i, s_{i+1}) > 0$ for all $i \geq 0$.

A path represents an execution (i. e., one possible behavior) of the system which the MDP is modelling.

Notation:

- $\mathrm{Paths}_{\mathrm{inf}}(s) =$ set of all infinite paths through the MDP starting in state s.
- $\mathrm{Paths}_{\mathrm{fin}}(s) =$ set of all finite paths from s.

Paths resolve both nondeterministic and probabilistic choices.

- How to reason about probabilities?

Schedulers

- To consider the probability of some behavior of the MDP
  - We first need to resolve the nondeterministic choices
  - ... which results in a DTMC
  - ... for which we can define a probability measure over paths.

Schedulers

- To consider the probability of some behavior of the MDP
  - We first need to resolve the nondeterministic choices
  - ... which results in a DTMC
  - ... for which we can define a probability measure over paths.
- An scheduler resolves non-deterministic choice in an MDP.
  - also known as "adversary", "policy", "strategy"

Schedulers

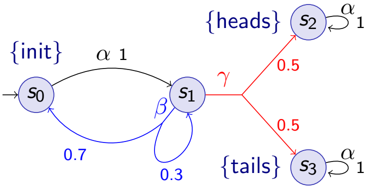
- To consider the probability of some behavior of the MDP
  - We first need to resolve the nondeterministic choices
  - ... which results in a DTMC
  - ... for which we can define a probability measure over paths.
- An scheduler resolves non-deterministic choice in an MDP.
  - also known as "adversary", "policy", "strategy"
- Formally:

- A scheduler σ of an MDP M is a function mapping every finite path $\omega = s_0\alpha_0 s_1 \ldots s_n$ to an element $\sigma(\omega) \in Act(s_n)$.
- i. e., it resolves the nondeterminism based on the execution history.

- Sched (or Sched_M) denotes the set of all schedulers.

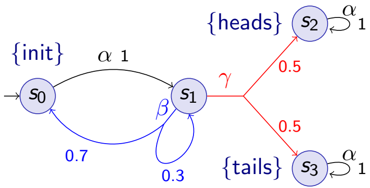
Schedulers: Examples

- Consider the previous MDP
  - note that s_1 is the only state for which $|Act(s)| > 1$.
  - i. e., s_1 is the only state for which a scheduler makes a choice



Schedulers: Examples

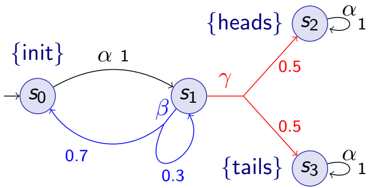
- Consider the previous MDP
  - note that s_1 is the only state for which $|Act(s)| > 1$.
  - i. e., s_1 is the only state for which a scheduler makes a choice
- Scheduler σ_1
  - picks action γ the first time
  - $\sigma_1(s_0 s_1) = \gamma$



(Note: actions omitted from paths for clarity.)

Schedulers: Examples

- Consider the previous MDP
  - note that s_1 is the only state for which $|Act(s)| > 1$.
  - i. e., s_1 is the only state for which a scheduler makes a choice
- Scheduler σ_1
  - picks action γ the first time
  - $\sigma_1(s_0 s_1) = \gamma$
- Scheduler σ_2
  - picks action β the first time, then γ
  - $\sigma_2(s_0 s_1) = \beta$, $\sigma_2(s_0 s_1 s_1) = \gamma$, $\sigma_2(s_0 s_1 s_0 s_1) = \gamma$.



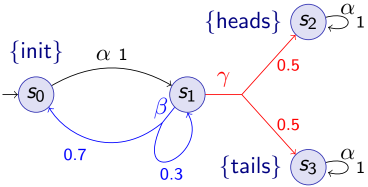
(Note: actions omitted from paths for clarity.)

Schedulers and paths

- $\text{Paths}^{\sigma}_{\text{inf}}(s) \subseteq \text{Paths}_{\text{inf}}(s)$
  - infinite paths from s where non-determinism resolved by σ
  - i. e., paths $\omega = s_0\alpha_0 s_1\alpha_1 s_2 \ldots$
  - for which $\sigma(s_0\alpha_0 s_1 \ldots s_n) = \alpha_n$

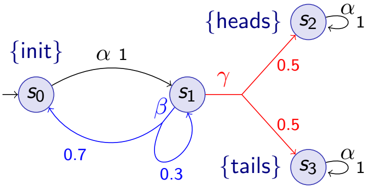
Schedulers and paths

- $\mathrm{Paths}^{\sigma}_{\mathrm{inf}}(s) \subseteq \mathrm{Paths}_{\mathrm{inf}}(s)$
  - infinite paths from s where non-determinism resolved by σ
  - i. e., paths $\omega = s_0\alpha_0 s_1\alpha_1 s_2 \ldots$
  - for which $\sigma(s_0\alpha_0 s_1 \ldots s_n) = \alpha_n$
- Scheduler σ_1:
  (pick γ the first time)
  - $\mathrm{Paths}^{\sigma_1}_{\mathrm{inf}}(s_0) = \{s_0 s_1 s_2^{\omega}, s_0 s_1 s_3^{\omega}\}$



Schedulers and paths

- $\text{Paths}^{\sigma}_{\text{inf}}(s) \subseteq \text{Paths}_{\text{inf}}(s)$
  - infinite paths from s where non-determinism resolved by σ
  - i. e., paths $\omega = s_0\alpha_0 s_1\alpha_1 s_2 \ldots$
  - for which $\sigma(s_0\alpha_0 s_1 \ldots s_n) = \alpha_n$
- Scheduler σ_1:
  (pick γ the first time)
  - $\text{Paths}^{\sigma_1}_{\text{inf}}(s_0) =$
    $\{s_0 s_1 s_2^{\omega}, s_0 s_1 s_3^{\omega}\}$
- Scheduler σ_2:
  (pick β the first time, then γ)
  - $\text{Paths}^{\sigma_2}_{\text{inf}}(s_0) =$
    $\{s_0 s_1 s_0 s_1 s_2^{\omega}, s_0 s_1 s_1 s_2^{\omega}, s_0 s_1 s_0 s_1 s_3^{\omega}, s_0 s_1 s_1 s_3^{\omega}\}$



Induced DTMCs

- Scheduler σ for MDP M induces infinite-state DTMC M^σ:

- $M^\sigma = (\text{Paths}^\sigma_{\text{fin}}(s_{\text{init}}), s_{\text{init}}, P^\sigma_{s_{\text{init}}}, L^\sigma)$ where:
  - states of the DTMC are the finite paths of σ starting in the initial state of M.
  - initial state is s_{init} (path of length 0 starting in s_{init})

And for $\omega = s_0 \alpha_0 s_1 \ldots s_n$:

- $P^\sigma_{s_{\text{init}}}(\omega, \omega') = \begin{cases} P(s_n, \alpha, s_{n+1}) & \text{if } \omega' = \omega \alpha_n s_{n+1} \wedge \sigma(\omega) = \alpha_n, \\ 0 & \text{otherwise.} \end{cases}$
- $L^\sigma(\omega) = L(s_n)$.

Induced DTMCs

- Scheduler σ for MDP M induces infinite-state DTMC M^σ:

- $M^\sigma = (\mathrm{Paths}^\sigma_{\mathrm{fin}}(s_{\mathrm{init}}), s_{\mathrm{init}}, P^\sigma_{s_{\mathrm{init}}}, L^\sigma)$ where:
  - states of the DTMC are the finite paths of σ starting in the initial state of M.
  - initial state is s_{init} (path of length 0 starting in s_{init})

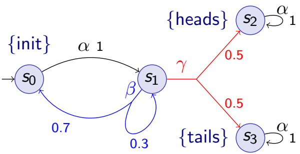
  And for $\omega = s_0\alpha_0 s_1 \ldots s_n$:
  - $P^\sigma_{s_{\mathrm{init}}}(\omega, \omega') = \begin{cases} P(s_n, \alpha, s_{n+1}) & \text{if } \omega' = \omega\alpha_n s_{n+1} \wedge \sigma(\omega) = \alpha_n, \\ 0 & \text{otherwise.} \end{cases}$
  - $L^\sigma(\omega) = L(s_n)$.

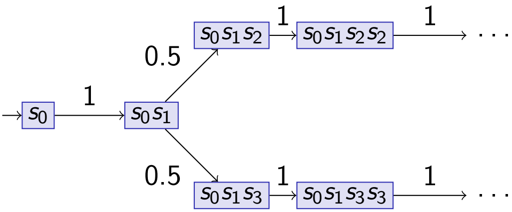
- 1-to-1 correspondence between $\mathrm{Paths}^\sigma_{\mathrm{inf}}(s_{\mathrm{init}})$ and paths of M^σ.

- This gives us a probability measure $\mathrm{Pr}^\sigma(s_{\mathrm{init}})$ over $\mathrm{Paths}^\sigma_{\mathrm{inf}}(s_{\mathrm{init}})$.
  - From probability measure over paths of M^σ.

Schedulers: Example

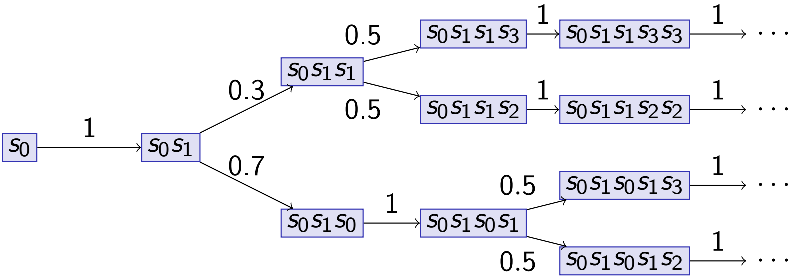
- Fragment of induced DTMC for scheduler σ_1:
  - σ_1 picks γ the first time.

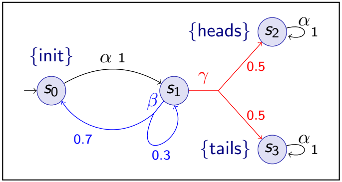




Schedulers: Example

- Fragment of the induced DTMC for scheduler σ_2
  - pick in s_1 β first, then γ





MDPs and probabilities

- $\mathsf{Pr}^{\sigma}(s, \psi) = \mathsf{Pr}^{\sigma}_{s}\{\omega \in \mathsf{Paths}^{\sigma}_{\mathsf{inf}}(s) \mid \omega \vDash \psi\}$
  - for some path formula ψ
  - and a scheduler σ,
  - e. g., $\mathsf{Pr}^{\sigma}(s, \mathbf{F}\,\mathsf{fail})$.

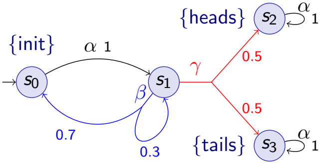
MDPs and probabilities

- $\Pr^{\sigma}(s, \psi) = \Pr^{\sigma}_{s}\{\omega \in \mathrm{Paths}^{\sigma}_{\mathrm{inf}}(s) \mid \omega \vDash \psi\}$
  - for some path formula ψ
  - and a scheduler σ,
  - e. g., $\Pr^{\sigma}(s, \mathbf{F}\,\mathrm{fail})$.
- MDP provides best-/worst-case analysis:
  - based on upper/lower bounds on probabilities
  - over all possible schedulers

$$p_{\min}(s, \psi) = \inf_{\sigma \in \mathrm{Sched}} \Pr^{\sigma}(s, \psi)$$
$$p_{\max}(s, \psi) = \sup_{\sigma \in \mathrm{Sched}} \Pr^{\sigma}(s, \psi)$$

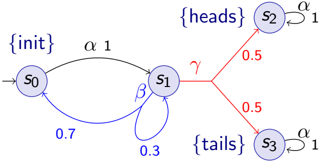
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) =$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) =$
  - where σ_i picks β $(i-1)$ times, then γ.



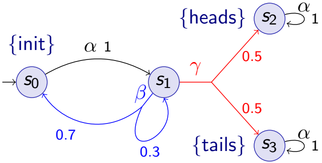
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) =$
  - where σ_i picks β $(i-1)$ times, then γ.



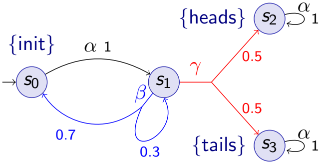
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
  - where σ_i picks β $(i-1)$ times, then γ.



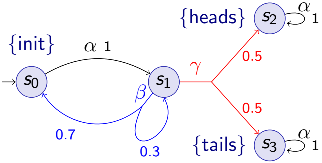
Examples

- $\mathrm{Pr}^{\sigma_1}(s_0, \mathbf{F}\,\mathrm{tails}) = 0.5$
- $\mathrm{Pr}^{\sigma_2}(s_0, \mathbf{F}\,\mathrm{tails}) = 0.5$
    - where σ_i picks β $(i-1)$ times, then γ.
- $p_{\max}(s_0, \mathbf{F}\,\mathrm{tails}) =$



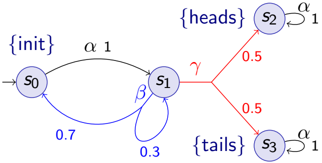
Examples

- $\mathrm{Pr}^{\sigma_1}(s_0, \mathbf{F}\,\mathrm{tails}) = 0.5$
- $\mathrm{Pr}^{\sigma_2}(s_0, \mathbf{F}\,\mathrm{tails}) = 0.5$
  - where σ_i picks β $(i-1)$ times, then γ.
- $p_{\max}(s_0, \mathbf{F}\,\mathrm{tails}) = 0.5$



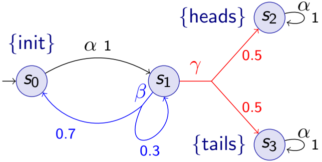
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
  - where σ_i picks β $(i-1)$ times, then γ.
- $p_{\max}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
- $p_{\min}(s_0, \mathbf{F}\,\text{tails}) =$



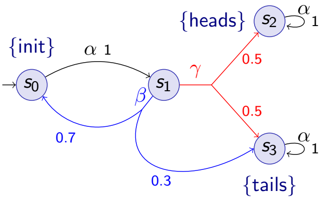
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
    - where σ_i picks β $(i-1)$ times, then γ.
- $p_{\max}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
- $p_{\min}(s_0, \mathbf{F}\,\mathsf{tails}) = 0$



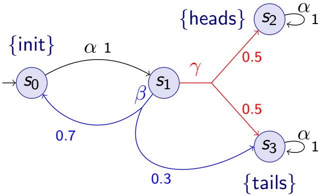
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) =$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) =$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\,\text{tails}) =$



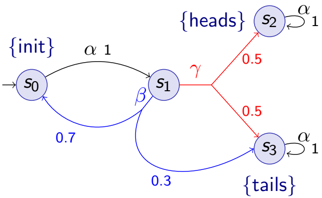
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\mathsf{tails}) =$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\,\mathsf{tails}) =$



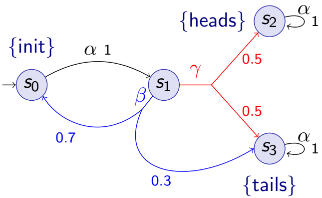
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\,\text{tails}) =$



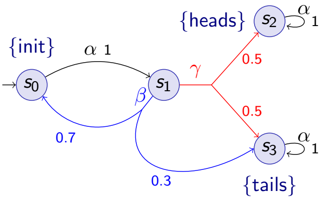
Examples

- $\mathrm{Pr}^{\sigma_1}(s_0, \mathbf{F}\,\mathrm{tails}) = 0.5$
- $\mathrm{Pr}^{\sigma_2}(s_0, \mathbf{F}\,\mathrm{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\mathrm{Pr}^{\sigma_3}(s_0, \mathbf{F}\,\mathrm{tails}) =$
  $0.3 + 0.7 \cdot 0.3 + 0.7^2 \cdot 0.5 = 0.755$



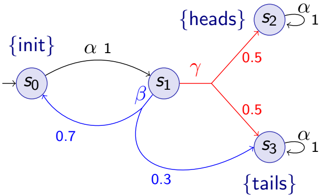
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\mathsf{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\,\mathsf{tails}) =$
  $0.3 + 0.7 \cdot 0.3 + 0.7^2 \cdot 0.5 = 0.755$
- ...



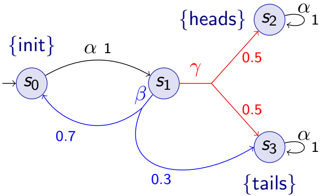
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\, \mathsf{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\, \mathsf{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\, \mathsf{tails}) =$
  $0.3 + 0.7 \cdot 0.3 + 0.7^2 \cdot 0.5 = 0.755$
- ...
- $p_{\max}(s_0, \mathbf{F}\, \mathsf{tails}) =$



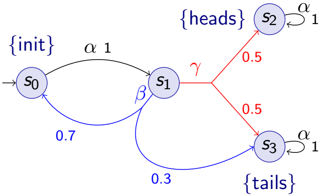
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\,\text{tails}) =$
  $0.3 + 0.7 \cdot 0.3 + 0.7^2 \cdot 0.5 = 0.755$
- ...
- $p_{\max}(s_0, \mathbf{F}\,\text{tails}) = 1$



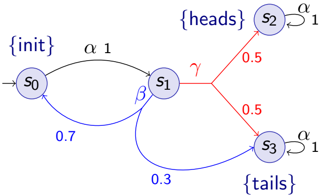
Examples

- $\Pr^{\sigma_1}(s_0, \mathbf{F}\,\text{tails}) = 0.5$
- $\Pr^{\sigma_2}(s_0, \mathbf{F}\,\text{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\Pr^{\sigma_3}(s_0, \mathbf{F}\,\text{tails}) =$
  $0.3 + 0.7 \cdot 0.3 + 0.7^2 \cdot 0.5 = 0.755$
- ...
- $p_{\max}(s_0, \mathbf{F}\,\text{tails}) = 1$
- $p_{\min}(s_0, \mathbf{F}\,\text{tails}) =$



Examples

- $\mathrm{Pr}^{\sigma_1}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$
- $\mathrm{Pr}^{\sigma_2}(s_0, \mathbf{F}\,\mathsf{tails}) =$
  $0.3 + 0.7 \cdot 0.5 = 0.65$
- $\mathrm{Pr}^{\sigma_3}(s_0, \mathbf{F}\,\mathsf{tails}) =$
  $0.3 + 0.7 \cdot 0.3 + 0.7^2 \cdot 0.5 = 0.755$
- ...
- $p_{\max}(s_0, \mathbf{F}\,\mathsf{tails}) = 1$
- $p_{\min}(s_0, \mathbf{F}\,\mathsf{tails}) = 0.5$

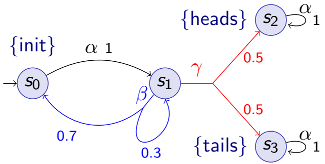


Memoryless schedulers

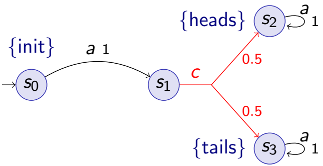
- Memoryless schedulers always pick the same choice in a state
  - also known as: positional, Markov, simple
  - formally: $\sigma(s_0\alpha_0 s_1 \ldots s_n)$ depends only on s_n
  - can be written as a mapping from states, i. e., $\sigma(s)$ for each $s \in S$
  - induced DTMC can be mapped to a $|S|$-state DTMC

Memoryless schedulers

- Memoryless schedulers always pick the same choice in a state
  - also known as: positional, ~~Markov~~, simple, stationary
  - formally: $\sigma(s_0\alpha_0 s_1 \ldots s_n)$ depends only on s_n
  - can be written as a mapping from states, i. e., $\sigma(s)$ for each $s \in S$
  - induced DTMC can be mapped to a $|S|$-state DTMC
- From previous example:
  - scheduler σ_1 (picks γ in S_1) is memoryless; σ_2 is not.



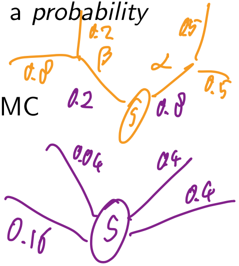
MDP



induced DTMC

Other classes of schedulers

- Finite-memory schedulers
  - finite number of *modes*, which can govern choices made
  - formally defined by a deterministic finite automaton
  - induced DTMC (for finite MDP) again mapped to a finite DTMC
- Randomized schedulers
  - maps finite paths $s_0\alpha_0 s_1 \dots s_n$ in MDP to a *probability distribution* over $Act(s_n)$
  - generalizes deterministic schedulers
  - still induces a (possibly infinite state) DTMC



Summary so far

- Nondeterminism
  - concurrency, unknown environments/parameters, abstraction
- Markov decision processes (MDPs)
  - discrete time + probability and nondeterminism
  - nondeterministic choice between multiple probability distributions
- Schedulers
  - resolution of nondeterminism only
  - induced set of paths and (infinite state) DTMCs
  - induced DTMC yields probability measure for a scheduler
  - best-/worst-case analysis: minimum/maximum probabilities
  - memoryless schedulers

Recall: MDPs

- Markov decision process: $M = (S, s_{\text{init}}, Act, P, L)$
- Scheduler $\sigma \in Sched_M$ resolves nondeterminism
- σ induces set of paths $\text{Paths}^\sigma(s)$ and DTMC M^σ
- M^σ yields probability space $\Pr_s^\sigma$ over $\text{Paths}^\sigma(s)$.
- $\Pr^\sigma(s, \psi) = \Pr_s^\sigma(\{\omega \in \text{Paths}^\sigma(s) \mid \omega \vDash \psi\})$
- MDP yields minimum/maximum probabilities:

$$p_{\min}(s, \psi) = \inf_{\sigma \in Sched_M} \Pr^\sigma(s, \psi),$$
$$p_{\max}(s, \psi) = \sup_{\sigma \in Sched_M} \Pr^\sigma(s, \psi).$$

Probabilistic reachability

- Minimum and maximum probability of reaching a target set $T \subseteq S$
- We assume, all states in T are marked by $a \in AP$.

$$p_{\min}(s, \mathbf{F}\, a) = \inf_{\sigma \in \mathrm{Sched}_M} \mathrm{Pr}^{\sigma}(s, \mathbf{F}\, a),$$
$$p_{\max}(s, \mathbf{F}\, a) = \sup_{\sigma \in \mathrm{Sched}_M} \mathrm{Pr}^{\sigma}(s, \mathbf{F}\, a).$$

- Vectors: $p_{\min}(\mathbf{F}\, a)$ and $p_{\max}(\mathbf{F}\, a)$
  - minimum/maximum probabilities for all states of the MDP

Qualitative probabilistic reachability

- Consider the problem of determining the states for which $p_{\min}(s, \mathbf{F}\, a)$ or $p_{\max}(s, \mathbf{F}\, a)$ is zero (or non-zero).
  - max case: $S^{\max=0} = \{s \in S \mid p_{\max}(s, \mathbf{F}\, a) = 0\}$.
  - this is just (non-probabilistic) reachability

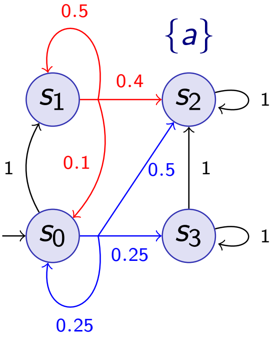
Qualitative probabilistic reachability

- Consider the problem of determining the states for which $p_{\min}(s, \mathbf{F}\, a)$ or $p_{\max}(s, \mathbf{F}\, a)$ is zero (or non-zero).
  - max case: $S^{\max=0} = \{s \in S \mid p_{\max}(s, \mathbf{F}\, a) = 0\}$.
  - this is just (non-probabilistic) reachability
- Pseudocode:

$R := \mathrm{Sat}(a)$
$\mathit{done} := \mathit{false}$
while $(\mathit{done} = \mathit{false})$ **do**
    $R' := R \cup \{s \in S \mid \exists \alpha \in \mathit{Act}(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
    if $(R' = R)$ **then** $\mathit{done} := \mathit{true}$
    $R := R'$
end while
return $S \setminus R$

Example max case

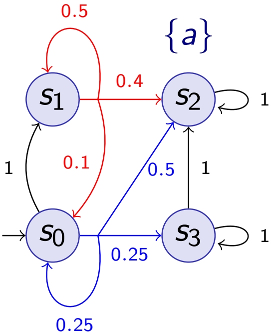
$R :=$ Sat(a)
done $:=$ *false*
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* $:=$ *true*
  $R := R'$
end while
return $S \setminus R$



Example max case

$R := \mathrm{Sat}(a)$
done $:=$ *false*
while (*done* $=$ *false*) **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* $:=$ *true*
  $R := R'$
end while
return $S \setminus R$

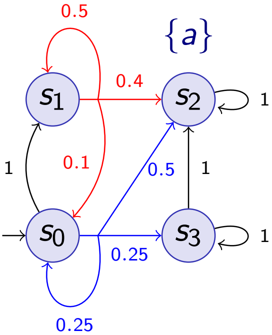
$\mathrm{Sat}(a) = \{s_2\}$



Example max case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

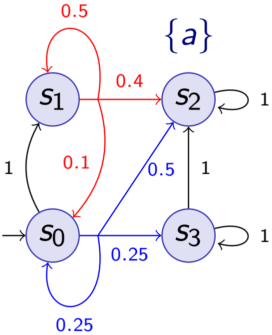
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$



Example max case

$R :=$ Sat(a)
done := *false*
while (*done* = *false*) **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* := *true*
  $R := R'$
end while
return $S \setminus R$

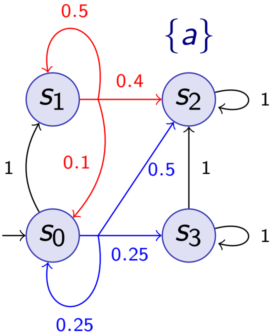
$$\text{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' =$$



Example max case

$R := \mathrm{Sat}(a)$
done $:=$ *false*
while $(done = false)$ **do**
$\quad R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
$\quad$ **if** $(R' = R)$ **then** *done* $:=$ *true*
$\quad R := R'$
end while
return $S \setminus R$

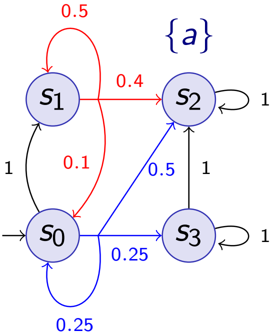
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_0, s_1, s_2, s_3\}$$



Example max case

$R := \text{Sat}(a)$
done $:=$ *false*
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* $:=$ *true*
  $R := R'$
end while
return $S \setminus R$

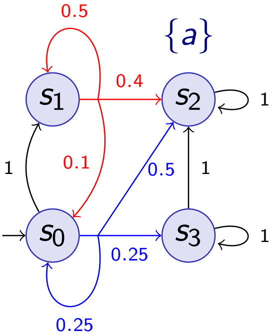
$$\text{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_0, s_1, s_2, s_3\}$$
$$R'' =$$



Example max case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

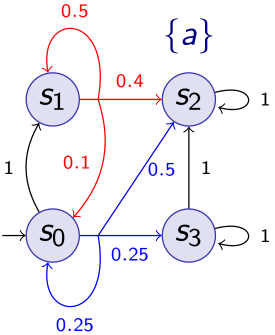
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_0, s_1, s_2, s_3\}$$
$$R'' = \{s_0, s_1, s_2, s_3\}$$



Example max case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \exists \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_0, s_1, s_2, s_3\}$$
$$R'' = \{s_0, s_1, s_2, s_3\}$$
$$S^{max=0} = \emptyset$$



Qualitative probabilistic reachability

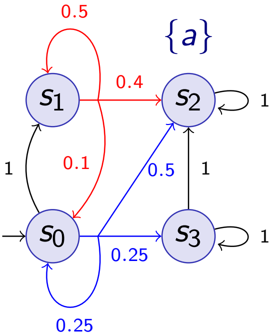
- min case: $S^{\min=0} = \{s \in S \mid p_{\min}(s, \mathbf{F}\, a) = 0\}$.
- Pseudocode:

$R := \mathrm{Sat}(a)$
$\mathit{done} := \mathit{false}$
while $(\mathit{done} = \mathit{false})$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in \mathit{Act}(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $\mathit{done} := \mathit{true}$
  $R := R'$
end while
return $S \setminus R$

- Note: Universal quantification over all choices

Example min case

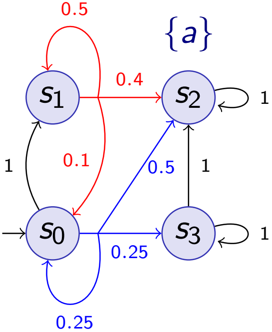
$R := \mathrm{Sat}(a)$
done := *false*
while (*done* = *false*) **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* := *true*
  $R := R'$
end while
return $S \setminus R$



Example min case

$R := \mathrm{Sat}(a)$
done $:=$ *false*
while (*done* $=$ *false*) **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* $:=$ *true*
  $R := R'$
end while
return $S \setminus R$

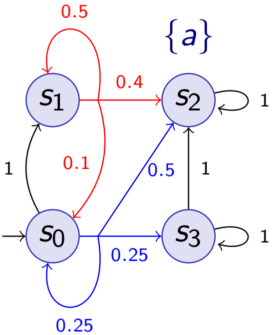
$\mathrm{Sat}(a) = \{s_2\}$



Example min case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

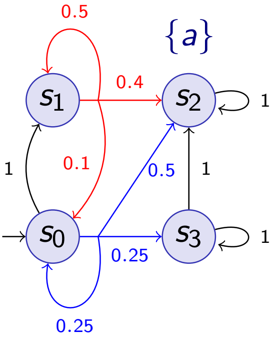
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$



Example min case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

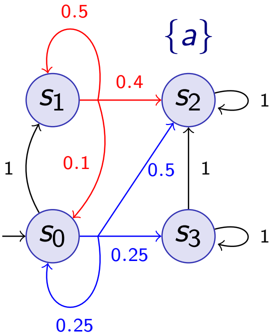
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' =$$



Example min case

$R := \mathrm{Sat}(a)$
done $:=$ *false*
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** *done* $:=$ *true*
  $R := R'$
end while
return $S \setminus R$

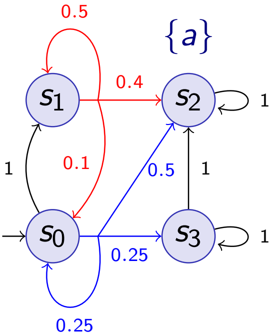
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_1, s_2\}$$



Example min case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

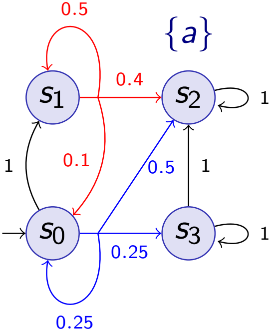
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_1, s_2\}$$
$$R'' =$$



Example min case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

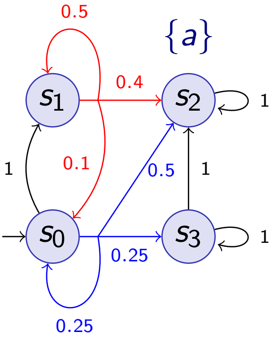
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_1, s_2\}$$
$$R'' = \{s_0, s_1, s_2\}$$



Example min case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

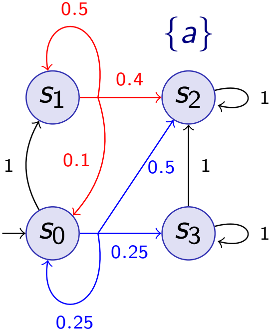
$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_1, s_2\}$$
$$R'' = \{s_0, s_1, s_2\}$$
$$R''' =$$



Example min case

$R := \text{Sat}(a)$
$done := false$
while $(done = false)$ **do**
  $R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
  if $(R' = R)$ **then** $done := true$
  $R := R'$
end while
return $S \setminus R$

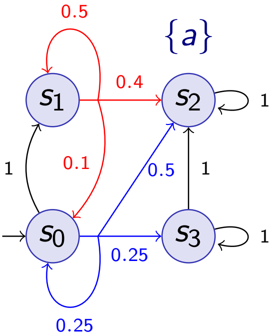
$$\begin{aligned} \text{Sat}(a) &= \{s_2\} \\ R &= \{s_2\} \\ R' &= \{s_1, s_2\} \\ R'' &= \{s_0, s_1, s_2\} \\ R''' &= \{s_0, s_1, s_2\} \end{aligned}$$



Example min case

$R := \mathrm{Sat}(a)$
$done := false$
while $(done = false)$ **do**
$\quad R' := R \cup \{s \in S \mid \forall \alpha \in Act(s),\ \exists s' \in R : P(s, \alpha, s') > 0\}$
$\quad$ **if** $(R' = R)$ **then** $done := true$
$\quad R := R'$
end while
return $S \setminus R$

$$\mathrm{Sat}(a) = \{s_2\}$$
$$R = \{s_2\}$$
$$R' = \{s_1, s_2\}$$
$$R'' = \{s_0, s_1, s_2\}$$
$$R''' = \{s_0, s_1, s_2\}$$
$$S^{min=0} = \{s_3\}$$



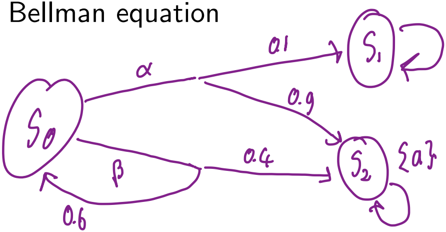
Quantitative reachability: min-optimality

The values $p_{\min}(s, \mathbf{F}\, a)$ are the unique solution of the following equations:

$$x_s = \begin{cases} 1 & \text{if } s \in \mathsf{Sat}(a), \\ 0 & \text{if } s \in S^{\min=0}, \\ \min\Big\{ \sum_{s' \in S} P(s,\alpha,s') \cdot x_{s'} \,\Big|\, \alpha \in Act(s) \Big\} & \text{otherwise.} \end{cases}$$

0 1

→ Bellman equation



$\vec{x} = x_0, x_1, x_2$

$x_0 = \min(0.1 \cdot 0 + 0.9 \cdot 1,$
$\quad 0.4 \cdot 1 + 0.6 \cdot x_0)$

$= \min(0.9, 0.4 + 0.6 \cdot x_0)$

$= 0.9$

Quantitative reachability: max-optimality

The values $p_{\max}(s, \mathbf{F}\, a)$ are the unique solution of the following equations:

$$x_s = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(a), \\ 0 & \text{if } s \in S^{\max=0}, \\ \max\Big\{ \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} \,\Big|\, \alpha \in Act(s) \Big\} & \text{otherwise.} \end{cases}$$

$\rightarrow$ Bellman equation

Memoryless schedulers

Theorem: For each MDP M with state space S there exists a memoryless scheduler $\sigma^{\max}$ which yields $p_{\max}(s, \mathbf{F}\, a)$ for all states $s \in S$.

Memoryless schedulers

Theorem: For each MDP M with state space S there exists a memoryless scheduler $\sigma^{\max}$ which yields $p_{\max}(s, \mathbf{F}\, a)$ for all states $s \in S$.

Proof: Let M be a finite MDP with state space S and $x_s = \Pr^{\max}(s, \mathbf{F}\, a)$. We prove the theorem by constructing a memoryless scheduler $\sigma^{\max}$ such that $\Pr^{\sigma^{\max}}(s, \mathbf{F}\, a) = x_s$.

Memoryless schedulers

Theorem: For each MDP M with state space S there exists a memoryless scheduler $\sigma^{\max}$ which yields $p_{\max}(s, \mathbf{F}\, a)$ for all states $s \in S$.

Proof: Let M be a finite MDP with state space S and $x_s = \Pr^{\max}(s, \mathbf{F}\, a)$. We prove the theorem by constructing a memoryless scheduler $\sigma^{\max}$ such that $\Pr^{\sigma^{\max}}(s, \mathbf{F}\, a) = x_s$.

1. For states $s \in \mathrm{Sat}(a)$ and states $s \in S^{\max=0}$ choose an arbitrary element of $Act(s)$. This does not influence the reachability probability.

Proof (cont.)

2. For states $s \in S \setminus (\mathrm{Sat}(a) \cup S^{\max=0})$ let $Act^{\max}(s) \subseteq Act(s)$ be the set such that
$$x_s = \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'}$$
for all $\alpha \in Act^{\max}(s)$.

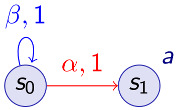
Proof (cont.)

2. For states $s \in S \setminus (\mathrm{Sat}(a) \cup S^{\max=0})$ let $Act^{\max}(s) \subseteq Act(s)$ be the set such that

$$x_s = \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'}$$

for all $\alpha \in Act^{\max}(s)$.

Observation: It does not suffice to select an arbitrary element of $Act^{\max}(s)$.



$$\begin{aligned} x_{s_1} &= 1 \\ x_{s_0} &= max\left\{1 \cdot x_{s_1},\ 1 \cdot x_{s_0}\right\} = 1 \end{aligned}$$

$Act^{\max}(s_0) = \{\alpha, \beta\}$. By choosing β we cannot reach s_1!

Proof (cont.)

We need a selection of actions which ensures the reachability of the target states $\mathrm{Sat}(a)$ in the induced DTMC.

Proof (cont.)

We need a selection of actions which ensures the reachability of the target states Sat(a) in the induced DTMC.

Consider the MDP $M^{\max}$ which results from M by removing all entries $\alpha \in Act(s) \setminus Act^{\max}(s)$ for all $s \in S \setminus S^{\max=0}$. This does not change the reachability probabilities.

Proof (cont.)

We need a selection of actions which ensures the reachability of the target states $\mathsf{Sat}(a)$ in the induced DTMC.

Consider the MDP M^{max} which results from M by removing all entries $\alpha \in Act(s) \setminus Act^{\mathsf{max}}(s)$ for all $s \in S \setminus S^{\mathsf{max}=0}$. This does not change the reachability probabilities.

For $s \in S \setminus S^{\mathsf{max}=0}$ let $\|s\|$ be the length of the shortest path from s to a target state in M^{max}. Then $\|s\| = 0$ iff $s \in \mathsf{Sat}(a)$.

Proof (cont.)

We need a selection of actions which ensures the reachability of the target states $\mathrm{Sat}(a)$ in the induced DTMC.

Consider the MDP M^{max} which results from M by removing all entries $\alpha \in Act(s) \setminus Act^{\mathrm{max}}(s)$ for all $s \in S \setminus S^{\mathrm{max}=0}$. This does not change the reachability probabilities.

For $s \in S \setminus S^{\mathrm{max}=0}$ let $\|s\|$ be the length of the shortest path from s to a target state in M^{max}. Then $\|s\| = 0$ iff $s \in \mathrm{Sat}(a)$.

Construction of the scheduler σ^{max} by induction on $\|s\|$.

Proof (cont.)

$\|s\| = 0$: Take an arbitrary entry of ~~*Steps*~~(s) *Act*(s)

$\|s\| > 0$: Let $\sigma^{\max}(s) = \alpha \in Act^{\max}(s)$ such that there is $s' \in S$ with $P(s, \alpha, s') > 0$ and $\|s'\| = \|s\| - 1$.

Proof (cont.)

$\|s\| = 0$: Take an arbitrary entry of $Steps(s)$
$\|s\| > 0$: Let $\sigma^{\max}(s) = \alpha \in Act^{\max}(s)$ such that there is $s' \in S$ with $P(s, \alpha, s') > 0$ and $\|s'\| = \|s\| - 1$.

Consider the induced DTMC $M^{\sigma^{\max}}$:

- memoryless scheduler $\sigma^{\max}$
- state space S
- reachability probability in $M^{\sigma^{\max}}$ is unique solution of

$$y_s = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(a), \\ 0 & \text{if } \mathrm{Sat}(a) \text{ not reachable from } s, \\ \sum\limits_{s' \in S} P^{\sigma^{\max}}(s, s') \cdot y_{s'} & \text{otherwise.} \end{cases}$$

$P^{\sigma^{\max}}(s, s') = P(s, \alpha, s')$ if $\sigma^{\max}(s) = \alpha$.
$\mathrm{Sat}(a)$ is not reachable from s if $s \in S^{\max=0}$.

Proof (cont.)

Optimality equation:

$$x_s = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(a), \\ 0 & \text{if } s \in S^{\max=0}, \\ \max\left\{ \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} \,\middle|\, \alpha \in Act(s) \right\} & \text{otherwise.} \end{cases}$$

Equation for our induced DTMC:

$$y_s = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(a), \\ 0 & \text{if } \mathrm{Sat}(a) \text{ not reachable from } s, \\ \sum_{s' \in S} P^{\sigma^{\max}}(s, s') \cdot y_{s'} & \text{otherwise.} \end{cases}$$

$P^{\sigma^{\max}}(s, s') = P(s, \alpha, s')$ if $\sigma^{\max}(s) = \alpha \in Act^{\max}(s)$.

Proof (cont.)

Optimality equation:

$$x_s = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(a), \\ 0 & \text{if } s \in S^{\max=0}, \\ \max\Big\{ \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'} \Big| \alpha \in Act(s) \Big\} & \text{otherwise.} \end{cases}$$

Equation for our induced DTMC:

$$y_s = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(a), \\ 0 & \text{if } \mathrm{Sat}(a) \text{ not reachable from } s, \\ \sum_{s' \in S} P^{\sigma^{\max}}(s, s') \cdot y_{s'} & \text{otherwise.} \end{cases}$$

$P^{\sigma^{\max}}(s, s') = P(s, \alpha, s')$ if $\sigma^{\max}(s) = \alpha \in Act^{\max}(s)$.

$\Rightarrow$ y_s is a solution of the optimality equation.
Since its solution is unique, $y_s = x_s = \mathrm{Pr}^{\max}(s, \mathbf{F}\, a)$. $\square$

Computing reachability probabilities

Several approaches:

1. Value iteration
   - approximate with iterative solution method
   - corresponds to a fixed point computation
   - preferable in practice, implemented in PRISM

2. Reduction to a linear programming (LP) problem
   - solve with linear optimization techniques (Simplex algorithm)
   - exact solution using well-known methods
   - better (theoretical) complexity, good for small examples

3. Policy iteration
   - iteration over adversaries.

Method 1: Value iteration

For minimum probabilities, it can be shown that:

$$p_{\min}(s, \mathbf{F}\, a) = \lim_{n\to\infty} x_s^{(n)}$$

where for $n \geq 0$

$$x_s^{(n+1)} = \begin{cases} 1 & \text{if } s \in \mathsf{Sat}(a) \\ 0 & \text{if } s \in S^{\min=0} \\ \min\Big\{ \sum_{s' \in S} P(s, \alpha, s') \cdot x_{s'}^{(n)} \Big| \alpha \in Act(s) \Big\} & \text{otherwise.} \end{cases}$$

and

$$x_s^{(0)} = \begin{cases} 1 & \text{if } s \in \mathsf{Sat}(a) \\ 0 & \text{otherwise.} \end{cases}$$

Analogue to the Jacobi method for linear equation systems.

Method 1: Value iteration

For maximum probabilities, it can be shown that:

$$p_{\max}(s, \mathbf{F}\, a) = \lim_{n\to\infty} x_s^{(n)}$$

where for $n \geq 0$

$$x_s^{(n+1)} = \begin{cases} 1 & \text{if } s \in \mathsf{Sat}(a) \\ 0 & \text{if } s \in S^{\max=0} \\ \max\left\{ \sum_{s' \in S} P(s,\alpha,s') \cdot x_{s'}^{(n)} \,\middle|\, \alpha \in Act(s) \right\} & \text{otherwise.} \end{cases}$$

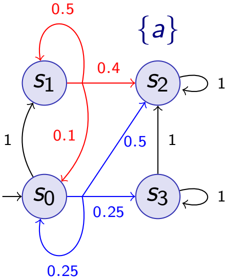
and

$$x_s^{(0)} = \begin{cases} 1 & \text{if } s \in \mathsf{Sat}(a) \\ 0 & \text{otherwise.} \end{cases}$$

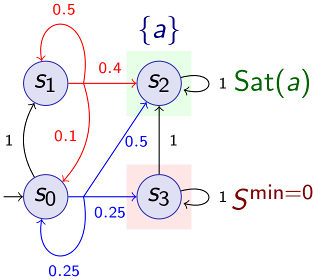
Analogue to the Jacobi method for linear equation systems.

Value iteration: Example

- Minimum/maximum probability of reaching an *a*-state



Value iteration: Example (min)



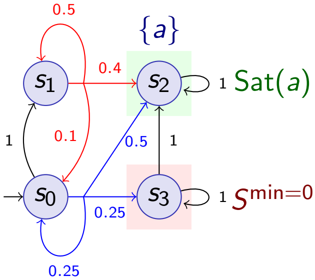
Compute: $p_{\min}(s_i, \mathbf{F}\ a)$

$\text{Sat}(a) = \{s_2\}$,
$S^{\min=0} = \{s_3\}$,
$S^? = \{s_0, s_1\}$

$$[x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}]$$

$n = 0 :\quad [0, 0, 1, 0]$

Value iteration: Example (min)

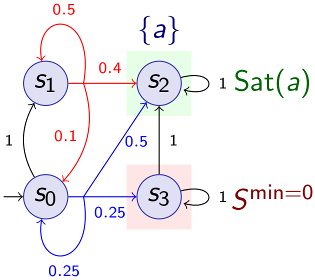


Compute: $p_{\min}(s_i, \mathbf{F}\, a)$

$\mathrm{Sat}(a) = \{s_2\}$,
$S^{\min=0} = \{s_3\}$,
$S^{?} = \{s_0, s_1\}$

$$
\begin{array}{ll}
 & [x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}] \\
n = 0: & [0, 0, 1, 0] \\
n = 1: & [\min(1 \cdot 0, 0.25 \cdot 0 + 0.25 \cdot 0 + 0.5 \cdot 1), \\
 & \quad 0.1 \cdot 0 + 0.5 \cdot 0 + 0.4 \cdot 1, 1, 0] \\
 & \quad = [0, 0.4, 1, 0]
\end{array}
$$

Value iteration: Example (min)

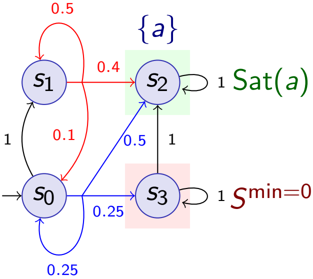


Compute: $p_{\min}(s_i, \mathbf{F}\, a)$

$\mathrm{Sat}(a) = \{s_2\}$,
$S^{\min=0} = \{s_3\}$,
$S^? = \{s_0, s_1\}$

$$
\begin{array}{ll}
 & [x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}] \\
n = 0: & [0, 0, 1, 0] \\
n = 1: & [\min(1 \cdot 0, 0.25 \cdot 0 + 0.25 \cdot 0 + 0.5 \cdot 1), \\
 & \quad 0.1 \cdot 0 + 0.5 \cdot 0 + 0.4 \cdot 1, 1, 0] \\
 & \quad = [0, 0.4, 1, 0] \\
n = 2: & [min(1 \cdot 0.4, 0.25 \cdot 0 + 0.25 \cdot 0 + 0.5 \cdot 1), \\
 & \quad 0.1 \cdot 0 + 0.5 \cdot 0.4 + 0.4 \cdot 1, 1, 0] \\
 & \quad = [0.4, 0.6, 1, 0]
\end{array}
$$

Value iteration: Example (min)

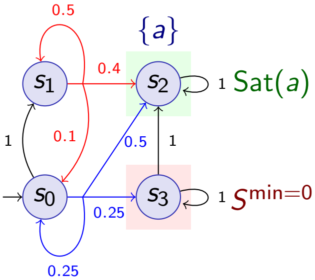


Compute: $p_{\min}(s_i, \mathbf{F}\, a)$

$\mathrm{Sat}(a) = \{s_2\}$,
$S^{\min=0} = \{s_3\}$,
$S^{?} = \{s_0, s_1\}$

$$
\begin{array}{ll}
 & [x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}] \\
n = 0: & [0, 0, 1, 0] \\
n = 1: & [\min(1 \cdot 0, 0.25 \cdot 0 + 0.25 \cdot 0 + 0.5 \cdot 1), \\
 & \quad 0.1 \cdot 0 + 0.5 \cdot 0 + 0.4 \cdot 1, 1, 0] \\
 & \quad = [0, 0.4, 1, 0] \\
n = 2: & [min(1 \cdot 0.4, 0.25 \cdot 0 + 0.25 \cdot 0 + 0.5 \cdot 1), \\
 & \quad 0.1 \cdot 0 + 0.5 \cdot 0.4 + 0.4 \cdot 1, 1, 0] \\
 & \quad = [0.4, 0.6, 1, 0] \\
n = 3: & \ldots
\end{array}
$$

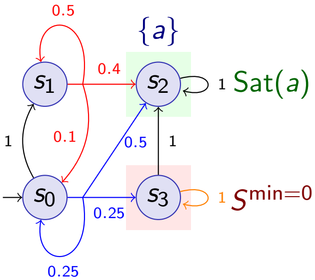
Value iteration: Example (min)



$$
\begin{array}{rl}
 & [x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}] \\
n = 0: & [0.000000, 0.000000, 1, 0] \\
n = 1: & [0.000000, 0.400000, 1, 0] \\
n = 2: & [0.400000, 0.600000, 1, 0] \\
n = 3: & [0.600000, 0.740000, 1, 0] \\
n = 4: & [0.650000, 0.830000, 1, 0] \\
n = 5: & [0.662500, 0.880000, 1, 0] \\
n = 6: & [0.665625, 0.906250, 1, 0] \\
n = 7: & [0.666406, 0.919688, 1, 0] \\
n = 8: & [0.666602, 0.926484, 1, 0] \\
 & \cdots \\
n = 20: & [0.666667, 0.933332, 1, 0] \\
n \to \infty: & \left[\frac{2}{3}, \frac{14}{15}, 1, 0\right]
\end{array}
$$

Generating an optimal scheduler

Min scheduler $\sigma^{\min}$



$$
\begin{array}{rl}
 & [x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}] \\
n = 20: & [0.666667, 0.933332, 1, 0] \\
n \to \infty: & \left[\frac{2}{3}, \frac{14}{15}, 1, 0\right]
\end{array}
$$

- In s_1 and s_2 only one choice is possible.
- In s_0 and s_3 we have two possibilities.
  - First determine $Act^{\min}(s_0)$ and $Act^{\min}(s_3)$:
    - $Act^{\min}(s_0) =$ "blue transition",
    - $Act^{\min}(s_3) =$ "orange transition".
  - For both states, the choice is unique; otherwise proceed (for max) as in the proof of the theorem on memoryless schedulers.

Linear programming

- Linear programming
  - optimization of a linear objective function
  - subject to a set of linear (in)equalities
- General form:
  - n real variables $x_1, x_2, \ldots, x_n$
  - Objective function: $\max c_1x_1 + c_2x_2 + \cdots + c_nx_n$
  - Constraints:

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &\leq b_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &\leq b_2 \\
&\cdots \\
a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &\leq b_m
\end{aligned}
$$

- In matrix/vector form:

$$
\max\ c^T x
$$

$$
\text{such that } Ax \leq b
$$

Solution of linear programs

Efficient algorithm for solving linear programs exist:

- Simplex algorithm (Danzig, 1947)
- Ellipsoid method (Khachiyan, 1979)
- Interior point method (Karmarkar, 1984)

Literature:

- Korte, Vygen – Combinatorial Optimization, Springer 2001
- Schrijver – Theory of Linear and Integer Programming, Wiley 1986

Method 2: Linear programming problem

Minimum probabilities $p_{\min}(s, \mathbf{F}\, a)$ can be computed as follows:

- $p_{\min}(s, \mathbf{F}\, a) = 1$ if $s \in \mathrm{Sat}(a)$
- $p_{\min}(s, \mathbf{F}\, a) = 0$ if $s \in S^{\min=0}$

Method 2: Linear programming problem

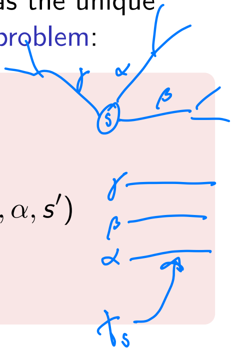
Minimum probabilities $p_{\min}(s, \mathbf{F}\, a)$ can be computed as follows:

- $p_{\min}(s, \mathbf{F}\, a) = 1$ if $s \in \mathrm{Sat}(a)$
- $p_{\min}(s, \mathbf{F}\, a) = 0$ if $s \in S^{\min=0}$
- values for the remaining states in the set $S^{?} = S \setminus (\mathrm{Sat}(a) \cup S^{\min=0})$ can be obtained as the unique solution of the following linear programming problem:

maximize $\sum_{s \in S^{?}} x_s$
such that

$$x_s \leq \sum_{s' \in S^{?}} P(s, \alpha, s') \cdot x_{s'} + \sum_{s' \in \mathrm{Sat}(a)} P(s, \alpha, s')$$

for all $s \in S^{?}$ and for all $\alpha \in Act(s)$.



Method 2: Linear programming problem

Maximum probabilities $p_{\max}(s, \mathbf{F}\, a)$ can be computed as follows:

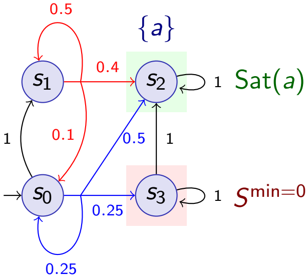
- $p_{\max}(s, \mathbf{F}\, a) = 1$ if $s \in \mathrm{Sat}(a)$
- $p_{\max}(s, \mathbf{F}\, a) = 0$ if $s \in S^{\max=0}$
- values for the remaining states in the set $S^? = S \setminus (\mathrm{Sat}(a) \cup S^{\max=0})$ can be obtained as the unique solution of the following linear programming problem:

minimize $\sum_{s \in S^?} x_s$
such that

$$x_s \geq \sum_{s' \in S^?} P(s, \alpha, s') \cdot x_{s'} + \sum_{s' \in \mathrm{Sat}(a)} P(s, \alpha, s')$$

for all $s \in S^?$ and for all $\alpha \in Act(s)$.

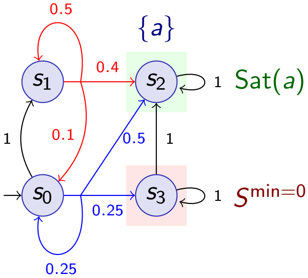
Linear programming: Example (min)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_s = 1$, $S^{\min=0}$: $x_3 = 0$
For $S^? = \{s_0, s_1\}$:
Maximize $x_0 + x_1$ subject to constraints:

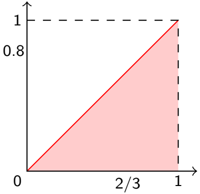
- $x_0 \leq x_1$
- $x_0 \leq 0.25x_0 + 0.5$
- $x_1 \leq 0.1x_0 + 0.5x_1 + 0.4$

Linear programming: Example (min)

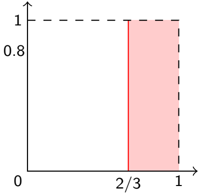


Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_s = 1$, $S^{\min=0}$: $x_3 = 0$
For $S^{?} = \{s_0, s_1\}$:
Maximize $x_0 + x_1$ subject to constraints:

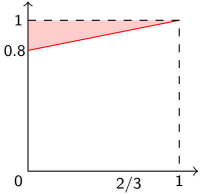
- $x_0 \leq x_1$
- $x_0 \leq 0.25x_0 + 0.5$
- $x_1 \leq 0.1x_0 + 0.5x_1 + 0.4$



$x_0 \leq x_1$

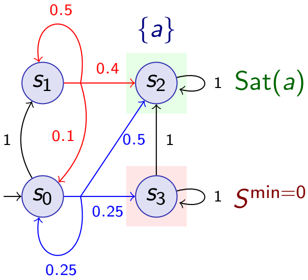


$x_0 \leq 2/3$



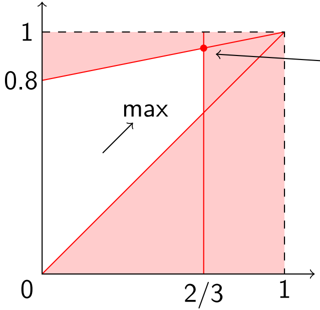
$x_1 \leq 0.2x_0 + 0.8$

Linear programming: Example (min)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_s = 1$, $S^{\min=0}$: $x_3 = 0$
For $S^? = \{s_0, s_1\}$:
Maximize $x_0 + x_1$ subject to constraints:

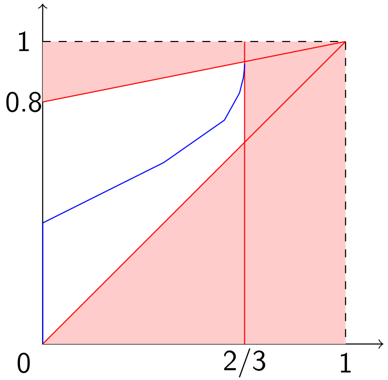
- $x_0 \leq x_1$
- $x_0 \leq 0.25x_0 + 0.5$
- $x_1 \leq 0.1x_0 + 0.5x_1 + 0.4$



Optimal solution:
$(x_0, x_1) = (2/3,\ 14/15)$

$p_{\min}(\mathbf{F}\, a) = (2/3, 14/15, 1, 0)$.

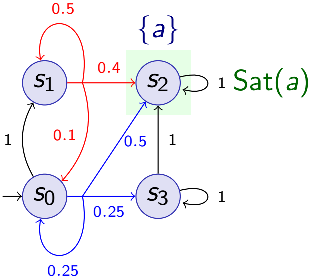
Value iteration + LP: Example



$$[x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}]$$

$$n = 0: \quad [0.000000, 0.000000, 1, 0]$$
$$n = 1: \quad [0.000000, 0.400000, 1, 0]$$
$$n = 2: \quad [0.400000, 0.600000, 1, 0]$$
$$n = 3: \quad [0.600000, 0.740000, 1, 0]$$
$$n = 4: \quad [0.650000, 0.830000, 1, 0]$$
$$n = 5: \quad [0.662500, 0.880000, 1, 0]$$
$$n = 6: \quad [0.665625, 0.906250, 1, 0]$$
$$n = 7: \quad [0.666406, 0.919688, 1, 0]$$
$$n = 8: \quad [0.666602, 0.926484, 1, 0]$$
$$\ldots$$
$$n = 20: \quad [0.666667, 0.933332, 1, 0]$$
$$n \to \infty: \quad \left[\frac{2}{3}, \frac{14}{15}, 1, 0\right]$$

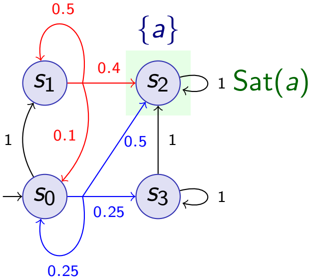
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 \geq 0.25x_0 + 0.25x_3 + 0.5$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 \geq x_2$
- $x_3 \geq x_3$

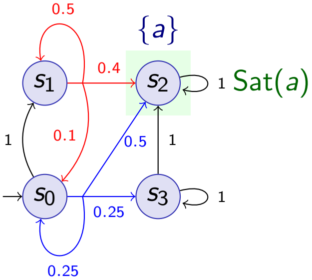
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\ a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 \geq 0.25x_0 + 0.25x_3 + 0.5$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 \geq x_2$
- $x_3 \geq x_3$ redundant!

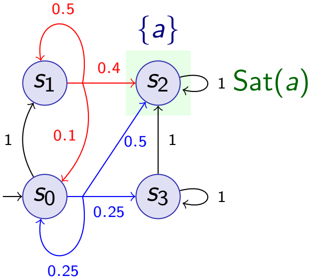
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\ a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 \geq 0.25x_0 + 0.25x_3 + 0.5$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 \geq x_2 \rightarrow x_3 = 1$
- $x_3 \geq x_3$

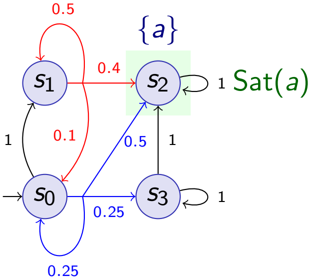
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 \geq 0.25x_0 + 0.25x_3 + 0.5$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 = 1$
- $x_3 \geq x_3$

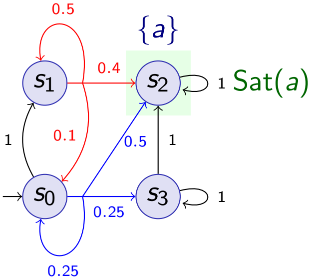
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\ a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 \geq 0.25x_0 + 0.75$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 = 1$
- $x_3 \geq x_3$

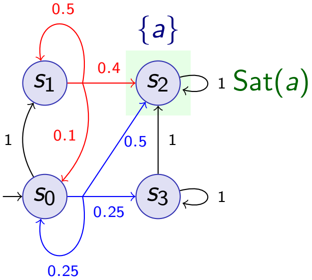
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $0.75x_0 \geq 0.75$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 = 1$
- $x_3 \geq x_3$

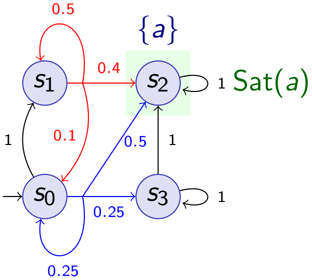
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 = 1$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 = 1$
- $x_3 \geq x_3$

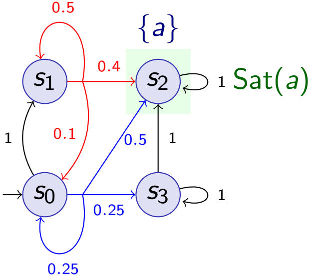
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 = 1$
- $x_1 \geq 0.2x_0 + 0.8$
- $x_3 = 1$
- $x_3 \geq x_3$

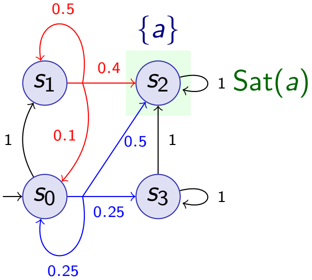
Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 = 1$
- $x_1 = 1$
- $x_3 = 1$
- $x_3 \geq x_3$

Linear programming: Example (max)



Let $x_i = p_{\min}(s_i, \mathbf{F}\, a)$
Sat(a) : $x_2 = 1$, $S^{\max=0} = \emptyset$
For $S^? = \{s_0, s_1, s_3\}$:
Minimize $x_0 + x_1 + x_3$ subject to constraints:

- $x_0 \geq x_1$
- $x_0 = 1$
- $x_1 = 1$
- $x_3 = 1$
- $x_3 \geq x_3$

Optimal solution: $p_{\max}(\mathbf{F}\, a) = (1, 1, 1, 1)$

Method 3: Policy iteration

- Value iteration:
  - iterates over (vectors of) probabilities
- Policy iteration:
  - iterates over schedulers ("policies")

Method 3: Policy iteration

- Value iteration:
  - iterates over (vectors of) probabilities
- Policy iteration:
  - iterates over schedulers ("policies")

1. Start with an arbitrary (memoryless) scheduler σ
2. Compute the reachability probabilities $\mathrm{Pr}^{\sigma}(\mathbf{F}\, a)$ for σ
3. Improve the scheduler in each state
4. Repeat steps 2+3 until no change in scheduler.

Method 3: Policy iteration

- Value iteration:
  - iterates over (vectors of) probabilities
- Policy iteration:
  - iterates over schedulers (“policies”)

1. Start with an arbitrary (memoryless) scheduler σ
2. Compute the reachability probabilities $\mathrm{Pr}^{\sigma}(\mathbf{F}\, a)$ for σ
3. Improve the scheduler in each state
4. Repeat steps 2+3 until no change in scheduler.

- Termination:
  - finite number of memoryless schedulers
  - improvement (in min/max probabilities) each time

Method 3: Policy iteration

1. Start with an arbitrary (memoryless) scheduler σ
   - Pick an element of $Act(s)$ for each state $s \in S$
2. Compute the reachability probabilities $\mathrm{Pr}^{\sigma}(\mathbf{F}\, a)$ for σ
   - probabilistic reachability on a DTMC
   - i. e., solve linear equation system
3. Improve the scheduler in each state:

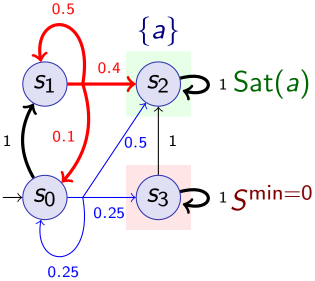
$$\sigma'(s) = \arg\min \Big\{ \sum_{s' \in S} P(s, \alpha, s') \cdot \mathrm{Pr}^{\sigma}(s', \mathbf{F}\, a) \Big| \alpha \in Act(s) \Big\}$$

$$\sigma'(s) = \arg\max \Big\{ \sum_{s' \in S} P(s, \alpha, s') \cdot \mathrm{Pr}^{\sigma}(s', \mathbf{F}\, a) \Big| \alpha \in Act(s) \Big\}.$$

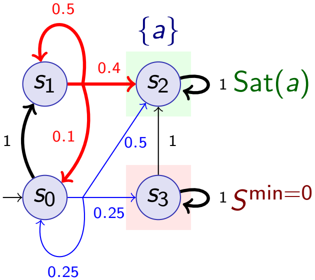
4. Repeat 2 and 3 until no change in scheduler.

Policy iteration: Example

Arbitrary scheduler σ



Policy iteration: Example



Arbitrary scheduler σ

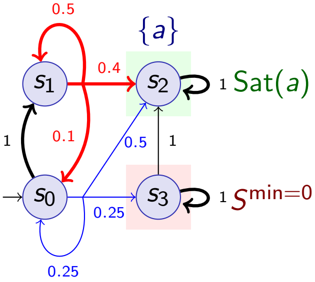
Compute $\Pr^{\sigma}(\mathbf{F}\, a)$:

- $x_2 = 1$
- $x_3 = 0$
- $x_0 = x_1$
- $x_1 = 0.1x_0 + 0.5x_1 + 0.4$

Solution:

$$\Pr^{\sigma}(\mathbf{F}\, a) = (1, 1, 1, 0)$$

Policy iteration: Example



Arbitrary scheduler σ

Compute $\Pr^{\sigma}(\mathbf{F}\, a)$:

- $x_2 = 1$
- $x_3 = 0$
- $x_0 = x_1$
- $x_1 = 0.1x_0 + 0.5x_1 + 0.4$

Solution:

$$\Pr^{\sigma}(\mathbf{F}\, a) = (1, 1, 1, 0)$$

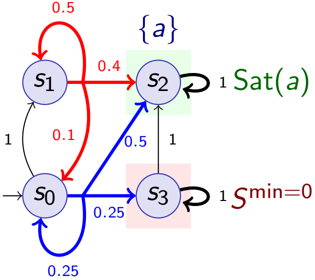
Refine σ in state s_0:

$\min\{1(1), 0.5(1) + 0.25(0) + 0.25(1)\}$

$= \min\{1, 0.75\} = 0.75$

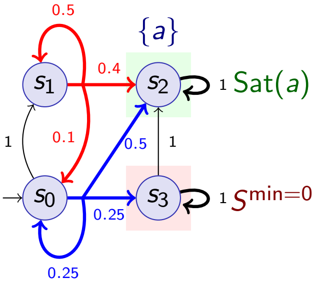
$\Rightarrow$ Take the blue transition instead of the black one.

Policy iteration: Example



Refined scheduler σ'

Policy iteration: Example



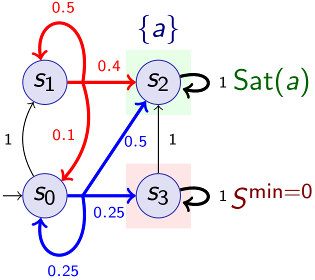
Refined scheduler σ'
Compute $\Pr^{\sigma'}(\mathbf{F}\, a)$:

- $x_2 = 1$
- $x_3 = 0$
- $x_0 = 0.25x_0 + 0.5$
- $x_1 = 0.1x_0 + 0.5x_1 + 0.4$

Solution:

$$\Pr^{\sigma}(\mathbf{F}\, a) = (2/3, 14/15, 1, 0)$$

Policy iteration: Example



Refined scheduler σ'

Compute $\Pr^{\sigma'}(\mathbf{F}\, a)$:

- $x_2 = 1$
- $x_3 = 0$
- $x_0 = 0.25x_0 + 0.5$
- $x_1 = 0.1x_0 + 0.5x_1 + 0.4$

Solution:

$$\Pr^{\sigma}(\mathbf{F}\, a) = (2/3, 14/15, 1, 0)$$

This is optimal.

Summary

- Probabilistic reachability in MDPs
- Qualitative case: min/max probability > 0
  - simple graph-based computation
  - need to do this first before other computation methods
- Memoryless schedulers suffice
  - Reduction to finite number of schedulers
- Computing reachability probabilities (and generation of optimal adversary)
  - Value iteration
    - approximate; iterative; fixed point computation
  - Reduction to linear programming problem
    - good for small examples; doesn't scale well
  - Policy iteration