Modeling and Verification of Probabilistic Systems

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://moves.rwth-aachen.de/teaching/ws-1516/movep15/

November 25, 2015

Overview

Markov decision process (MDP)

Markov decision process (MDP)

Markov decision processes

- In MDPs, both nondeterministic and probabilistic choices coexist.

Markov decision process (MDP)

Markov decision processes

- In MDPs, both nondeterministic and probabilistic choices coexist.
- MDPs are transition systems in which in any state a nondeterministic choice between probability distributions exists.

Markov decision process (MDP)

Markov decision processes

- In MDPs, both nondeterministic and probabilistic choices coexist.
- MDPs are transition systems in which in any state a nondeterministic choice between probability distributions exists.
- Once a probability distribution has been chosen nondeterministically, the next state is selected probabilistically—as in DTMCs.

Markov decision process (MDP)

Markov decision processes

- In MDPs, both nondeterministic and probabilistic choices coexist.
- MDPs are transition systems in which in any state a nondeterministic choice between probability distributions exists.
- Once a probability distribution has been chosen nondeterministically, the next state is selected probabilistically—as in DTMCs.
- Any MC is thus an MDP in which in any state the probability distribution is uniquely determined.

Markov decision process (MDP)

Markov decision processes

- In MDPs, both nondeterministic and probabilistic choices coexist.
- MDPs are transition systems in which in any state a nondeterministic choice between probability distributions exists.
- Once a probability distribution has been chosen nondeterministically, the next state is selected probabilistically—as in DTMCs.
- Any MC is thus an MDP in which in any state the probability distribution is uniquely determined.

Markov decision process (MDP)

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S is a countable set of states with initial distribution $\iota_{\text{init}} : S \rightarrow [0, 1]$

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S is a countable set of states with initial distribution $\iota_{\text{init}} : S \to [0, 1]$
- Act is a finite set of actions

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S is a countable set of states with initial distribution $\iota_{\text{init}} : S \to [0, 1]$
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0, 1]$, transition probability function

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S is a countable set of states with initial distribution $\iota_{\text{init}} : S \to [0, 1]$
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0, 1]$, transition probability function such that:

$$\text{for all } s \in S \text{ and } \alpha \in Act : \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{ 0, 1 \}$$

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S is a countable set of states with initial distribution $\iota_{\text{init}} : S \to [0, 1]$
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0, 1]$, transition probability function such that:

$$\text{for all } s \in S \text{ and } \alpha \in Act : \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{ 0, 1 \}$$

- AP is a set of atomic propositions and labeling $L : S \to 2^{AP}$.

Markov decision process (MDP)

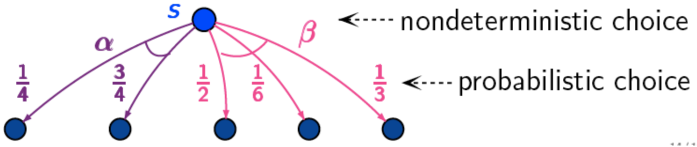
Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S is a countable set of states with initial distribution $\iota_{\text{init}} : S \to [0,1]$
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0,1]$, transition probability function such that:

$$\text{for all } s \in S \text{ and } \alpha \in Act : \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{0, 1\}$$

- AP is a set of atomic propositions and labeling $L : S \to 2^{AP}$.



Markov decision process (MDP)

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S, $\iota_{\text{init}} : S \to [0, 1]$, AP and L are as before, i.e., as for DTMCs, and
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0, 1]$, transition probability function such that:

$$\text{for all } s \in S \text{ and } \alpha \in Act : \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{ 0, 1 \}$$

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S, $\iota_{\text{init}} : S \to [0, 1]$, AP and L are as before, i.e., as for DTMCs, and
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0, 1]$, transition probability function such that:

$$\text{for all } s \in S \text{ and } \alpha \in Act : \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{ 0, 1 \}$$

Enabled actions

Let $Act(s) \; = \; \{ \alpha \in Act \mid \exists s' \in S. \, \mathbf{P}(s, \alpha, s') > 0 \}$ be the set of enabled actions in state s.

Markov decision process (MDP)

Markov decision process

An MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ where

- S, $\iota_{\text{init}} : S \to [0, 1]$, AP and L are as before, i.e., as for DTMCs, and
- Act is a finite set of actions
- $\mathbf{P} : S \times Act \times S \to [0, 1]$, transition probability function such that:

$$\text{for all } s \in S \text{ and } \alpha \in Act : \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{ 0, 1 \}$$

Enabled actions

Let $Act(s) \; = \; \{ \alpha \in Act \mid \exists s' \in S. \mathbf{P}(s, \alpha, s') > 0 \}$ be the set of enabled actions in state s. We require $Act(s) \neq \varnothing$ for any state s.

Overview

Policies

Policy

Policies

Policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP. A policy for $\mathcal{M}$ is a function $\mathfrak{S} : S^+ \rightarrow Act$

Policies

Policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\mathrm{init}}, AP, L)$ be an MDP. A policy for $\mathcal{M}$ is a function $\mathfrak{S} : S^+ \to Act$ such that $\mathfrak{S}(s_0\, s_1 \ldots s_n) \in Act(s_n)$ for all $s_0\, s_1 \ldots s_n \in S^+$.

Policies

Policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP. A policy for $\mathcal{M}$ is a function $\mathfrak{S} : S^+ \rightarrow Act$ such that $\mathfrak{S}(s_0\, s_1 \ldots s_n) \in Act(s_n)$ for all $s_0\, s_1 \ldots s_n \in S^+$.

The path

$$\pi \;=\; s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \xrightarrow{\alpha_3} \ldots$$

is called a $\mathfrak{S}$-path if $\alpha_i = \mathfrak{S}(s_0 \ldots s_{i-1})$ for all $i > 0$.

Induced DTMC of an MDP by a policy

Induced DTMC of an MDP by a policy

DTMC of an MDP induced by a policy

Induced DTMC of an MDP by a policy

DTMC of an MDP induced by a policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP and $\mathfrak{S}$ a policy on $\mathcal{M}$. The DTMC induced by $\mathfrak{S}$, denoted $\mathcal{M}_{\mathfrak{S}}$, is given by

$$\mathcal{M}_{\mathfrak{S}} \;=\; (S^+, \mathbf{P}_{\mathfrak{S}}, \iota_{\text{init}}, AP, L')$$

Induced DTMC of an MDP by a policy

DTMC of an MDP induced by a policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP and $\mathfrak{S}$ a policy on $\mathcal{M}$. The DTMC induced by $\mathfrak{S}$, denoted $\mathcal{M}_{\mathfrak{S}}$, is given by

$$\mathcal{M}_{\mathfrak{S}} \;=\; (S^+, \mathbf{P}_{\mathfrak{S}}, \iota_{\text{init}}, AP, L')$$

where for $\sigma = s_0 s_1 \ldots s_n$: $\mathbf{P}_{\mathfrak{S}}\big(\sigma,\ \sigma\, s_{n+1}\big) \;=\; \mathbf{P}\big(s_n,\ \mathfrak{S}(\sigma),\ s_{n+1}\big)$

Induced DTMC of an MDP by a policy

DTMC of an MDP induced by a policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP and $\mathfrak{S}$ a policy on $\mathcal{M}$. The DTMC induced by $\mathfrak{S}$, denoted $\mathcal{M}_{\mathfrak{S}}$, is given by

$$\mathcal{M}_{\mathfrak{S}} \; = \; (S^+, \mathbf{P}_{\mathfrak{S}}, \iota_{\text{init}}, AP, L')$$

where for $\sigma = s_0 s_1 \ldots s_n$: $\mathbf{P}_{\mathfrak{S}}\big(\sigma, \; \sigma\, s_{n+1}\big) \; = \; \mathbf{P}\big(s_n, \, \mathfrak{S}(\sigma), \, s_{n+1}\big)$ and $L'(\sigma) \, = \, L(s_n)$.

Induced DTMC of an MDP by a policy

DTMC of an MDP induced by a policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP and $\mathfrak{S}$ a policy on $\mathcal{M}$. The DTMC induced by $\mathfrak{S}$, denoted $\mathcal{M}_{\mathfrak{S}}$, is given by

$$\mathcal{M}_{\mathfrak{S}} \; = \; (S^+, \mathbf{P}_{\mathfrak{S}}, \iota_{\text{init}}, AP, L')$$

where for $\sigma = s_0 s_1 \ldots s_n$: $\mathbf{P}_{\mathfrak{S}}(\sigma, \; \sigma\, s_{n+1}) \; = \; \mathbf{P}(s_n, \, \mathfrak{S}(\sigma), \, s_{n+1})$ and $L'(\sigma) \, = \, L(s_n)$.

$\mathcal{M}_{\mathfrak{S}}$ is infinite, even if the MDP $\mathcal{M}$ is finite.

Induced DTMC of an MDP by a policy

DTMC of an MDP induced by a policy

Let $\mathcal{M} = (S, Act, \mathbf{P}, \iota_{\text{init}}, AP, L)$ be an MDP and $\mathfrak{S}$ a policy on $\mathcal{M}$. The DTMC induced by $\mathfrak{S}$, denoted $\mathcal{M}_{\mathfrak{S}}$, is given by

$$\mathcal{M}_{\mathfrak{S}} \ = \ (S^+, \mathbf{P}_{\mathfrak{S}}, \iota_{\text{init}}, AP, L')$$

where for $\sigma = s_0 s_1 \ldots s_n$: $\mathbf{P}_{\mathfrak{S}}\big(\sigma,\ \sigma\, s_{n+1}\big) \ = \ \mathbf{P}\big(s_n, \mathfrak{S}(\sigma), s_{n+1}\big)$ and $L'(\sigma) = L(s_n)$.

$\mathcal{M}_{\mathfrak{S}}$ is infinite, even if the MDP $\mathcal{M}$ is finite. Since policy $\mathfrak{S}$ might select different actions for finite paths that end in the same state s, a policy as defined above is also referred to as *history-dependent*.

Probability measure on MDP

Probability measure on MDP

Probability measure on MDP

Let $Pr^{\mathcal{M}}_{\mathfrak{S}}$, or simply $Pr^{\mathfrak{S}}$, denote the probability measure $Pr^{\mathcal{M}_{\mathfrak{S}}}$ associated with the DTMC $\mathcal{M}_{\mathfrak{S}}$.

Probability measure on MDP

Probability measure on MDP

Let $Pr_{\mathfrak{S}}^{\mathcal{M}}$, or simply $Pr^{\mathfrak{S}}$, denote the probability measure $Pr^{\mathcal{M}_{\mathfrak{S}}}$ associated with the DTMC $\mathcal{M}_{\mathfrak{S}}$.

This measure is the basis for associating probabilities with events in the MDP $\mathcal{M}$.

Probability measure on MDP

Probability measure on MDP

Let $Pr^{\mathcal{M}}_{\mathfrak{S}}$, or simply $Pr^{\mathfrak{S}}$, denote the probability measure $Pr^{\mathcal{M}_{\mathfrak{S}}}$ associated with the DTMC $\mathcal{M}_{\mathfrak{S}}$.

This measure is the basis for associating probabilities with events in the MDP $\mathcal{M}$. Let, e.g., $P \subseteq \left(2^{AP}\right)^{\omega}$ be an ω-regular property.

Probability measure on MDP

Probability measure on MDP

Let $Pr^{\mathcal{M}}_{\mathfrak{S}}$, or simply $Pr^{\mathfrak{S}}$, denote the probability measure $Pr^{\mathcal{M}_{\mathfrak{S}}}$ associated with the DTMC $\mathcal{M}_{\mathfrak{S}}$.

This measure is the basis for associating probabilities with events in the MDP $\mathcal{M}$. Let, e.g., $P \subseteq (2^{AP})^{\omega}$ be an ω-regular property. Then $Pr^{\mathfrak{S}}(P)$ is defined as:

$$Pr^{\mathfrak{S}}(P) \;=\; Pr^{\mathcal{M}_{\mathfrak{S}}}(P) \;=\; Pr_{\mathcal{M}_{\mathfrak{S}}}\{\, \pi \in Paths(\mathcal{M}_{\mathfrak{S}}) \mid trace(\pi) \in P \,\}.$$

Positional policy

Positional policy

Positional policy

Let $\mathcal{M}$ be an MDP with state space S. Policy $\mathfrak{S}$ on $\mathcal{M}$ is *positional* (or: *memoryless*) iff for each sequence $s_0\, s_1\, \ldots s_n$ and $t_0\, t_1 \ldots t_m \in S^+$ with $s_n = t_m$:

$$\mathfrak{S}(s_0\, s_1\, \ldots s_n) \;=\; \mathfrak{S}(t_0\, t_1 \ldots t_m).$$

Positional policy

Positional policy

Let $\mathcal{M}$ be an MDP with state space S. Policy $\mathfrak{S}$ on $\mathcal{M}$ is *positional* (or: *memoryless*) iff for each sequence $s_0\,s_1\,\ldots s_n$ and $t_0\,t_1\ldots t_m \in S^+$ with $s_n = t_m$:

$$\mathfrak{S}(s_0\,s_1\,\ldots s_n) \;=\; \mathfrak{S}(t_0\,t_1\ldots t_m).$$

In this case, $\mathfrak{S}$ can be viewed as a function $\mathfrak{S} : S \to Act$.

Positional policy

Positional policy

Let $\mathcal{M}$ be an MDP with state space S. Policy $\mathfrak{S}$ on $\mathcal{M}$ is *positional* (or: *memoryless*) iff for each sequence $s_0\, s_1\, \ldots s_n$ and $t_0\, t_1 \ldots t_m \in S^+$ with $s_n = t_m$:

$$\mathfrak{S}(s_0\, s_1\, \ldots s_n) \ = \ \mathfrak{S}(t_0\, t_1 \ldots t_m).$$

In this case, $\mathfrak{S}$ can be viewed as a function $\mathfrak{S} : S \to Act$.

Policy $\mathfrak{S}$ is positional if it always selects the same action in a given state.

Positional policy

Positional policy

Let $\mathcal{M}$ be an MDP with state space S. Policy $\mathfrak{S}$ on $\mathcal{M}$ is *positional* (or: *memoryless*) iff for each sequence $s_0\, s_1 \ldots s_n$ and $t_0\, t_1 \ldots t_m \in S^+$ with $s_n = t_m$:

$$\mathfrak{S}(s_0\, s_1 \ldots s_n) \;=\; \mathfrak{S}(t_0\, t_1 \ldots t_m).$$

In this case, $\mathfrak{S}$ can be viewed as a function $\mathfrak{S} : S \to Act$.

Policy $\mathfrak{S}$ is positional if it always selects the same action in a given state. This choice is independent of what has happened in the history, i.e., which path led to the current state.

Finite-memory policies

- *Finite-memory policies* (shortly: fm-policies) are a generalisation of positional policies.
- The behavior of an fm-policy is described by a deterministic finite automaton (DFA).
- The selection of the action to be performed in the MDP $\mathcal{M}$ depends on the current state of $\mathcal{M}$ (as before) and the current state (called *mode*) of the policy, i.e., the DFA.

Finite-memory policy

Finite-memory policy

Finite-memory policy

Let $\mathcal{M}$ be an MDP with state space S and action set Act.
A *finite-memory policy* $\mathfrak{S}$ for $\mathcal{M}$ is a tuple $\mathfrak{S} = (Q, act, \Delta, start)$ with:

- Q is a finite set of modes,
- $\Delta : Q \times S \rightarrow Q$ is the transition function,

Finite-memory policy

Finite-memory policy

Let $\mathcal{M}$ be an MDP with state space S and action set Act.
A *finite-memory policy* $\mathfrak{S}$ for $\mathcal{M}$ is a tuple $\mathfrak{S} = (Q, act, \Delta, start)$ with:

- Q is a finite set of modes,
- $\Delta : Q \times S \rightarrow Q$ is the transition function,
- $act : Q \times S \rightarrow Act$ is a function that selects an action $act(q, s) \in Act(s)$ for any mode $q \in Q$ and state $s \in S$ of $\mathcal{M}$,

Finite-memory policy

Finite-memory policy

Let $\mathcal{M}$ be an MDP with state space S and action set Act.
A *finite-memory policy* $\mathfrak{S}$ for $\mathcal{M}$ is a tuple $\mathfrak{S} = (Q, act, \Delta, start)$ with:

- Q is a finite set of modes,
- $\Delta : Q \times S \rightarrow Q$ is the transition function,
- $act : Q \times S \rightarrow Act$ is a function that selects an action $act(q, s) \in Act(s)$ for any mode $q \in Q$ and state $s \in S$ of $\mathcal{M}$,
- $start : S \rightarrow Q$ is a function that selects a starting mode for state $s \in S$.

An MDP under a finite-memory policy

An MDP under a finite-memory policy

The behavior of an MDP $\mathcal{M}$ under fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is:

An MDP under a finite-memory policy

The behavior of an MDP $\mathcal{M}$ under fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is:

- Initially, a starting state s_0 is randomly determined according to the initial distribution ι_{init}, i.e., $\iota_{\text{init}}(s_0) > 0$.

An MDP under a finite-memory policy

The behavior of an MDP $\mathcal{M}$ under fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is:

- Initially, a starting state s_0 is randomly determined according to the initial distribution ι_{init}, i.e., $\iota_{\text{init}}(s_0) > 0$.

- The fm-policy $\mathfrak{S}$ initializes its DFA to the mode $q_0 = start(s_0) \in Q$.

An MDP under a finite-memory policy

The behavior of an MDP $\mathcal{M}$ under fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is:

- Initially, a starting state s_0 is randomly determined according to the initial distribution ι_{init}, i.e., $\iota_{\text{init}}(s_0) > 0$.
- The fm-policy $\mathfrak{S}$ initializes its DFA to the mode $q_0 = start(s_0) \in Q$.
- If $\mathcal{M}$ is in state s and the current mode of $\mathfrak{S}$ is q, then the decision of $\mathfrak{S}$, i.e., the selected action, is $\alpha = act(q, s) \in Act(s)$.

An MDP under a finite-memory policy

The behavior of an MDP $\mathcal{M}$ under fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is:

- Initially, a starting state s_0 is randomly determined according to the initial distribution ι_{init}, i.e., $\iota_{\text{init}}(s_0) > 0$.
- The fm-policy $\mathfrak{S}$ initializes its DFA to the mode $q_0 = start(s_0) \in Q$.
- If $\mathcal{M}$ is in state s and the current mode of $\mathfrak{S}$ is q, then the decision of $\mathfrak{S}$, i.e., the selected action, is $\alpha = act(q, s) \in Act(s)$.
- The policy changes to mode $\Delta(q, s)$, while $\mathcal{M}$ performs the selected action α and randomly moves to the next state according to the distribution $\mathbf{P}(s, \alpha, \cdot)$.

Finite-memory policies

Relation fm-policy to definition policy

Finite-memory policies

Relation fm-policy to definition policy

An fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is identified with policy, $\mathfrak{S}' : Paths^* \rightarrow Act$ which is defined as follows.

Finite-memory policies

Relation fm-policy to definition policy

An fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is identified with policy, $\mathfrak{S}' : Paths^* \to Act$ which is defined as follows.

1. For the starting state s_0, let $\mathfrak{S}'(s_0) = act(start(s_0), s_0)$.

Finite-memory policies

Relation fm-policy to definition policy

An fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is identified with policy, $\mathfrak{S}' : Paths^* \to Act$ which is defined as follows.

1. For the starting state s_0, let $\mathfrak{S}'(s_0) = act(start(s_0), s_0)$.
2. For path fragment $\widehat{\pi} = s_0\, s_1 \ldots s_n$ let

$$\mathfrak{S}'(\widehat{\pi}) \;=\; act(q_n, s_n)$$

   where $q_0 = start(s_0)$ and $q_{i+1} = \Delta(q_i, s_i)$ for $0 \leqslant i \leqslant n$.

Finite-memory policies

Relation fm-policy to definition policy

An fm-policy $\mathfrak{S} = (Q, act, \Delta, start)$ is identified with policy, $\mathfrak{S}' : Paths^* \to Act$ which is defined as follows.

1. For the starting state s_0, let $\mathfrak{S}'(s_0) = act(start(s_0), s_0)$.
2. For path fragment $\widehat{\pi} = s_0 \, s_1 \ldots s_n$ let

$$\mathfrak{S}'(\widehat{\pi}) \; = \; act(q_n, s_n)$$

where $q_0 = start(s_0)$ and $q_{i+1} = \Delta(q_i, s_i)$ for $0 \leqslant i \leqslant n$.

Positional policies can be considered as fm-policies with just a single mode.

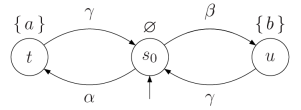
Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

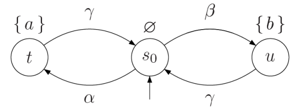
Consider the MDP:



Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Consider the MDP:

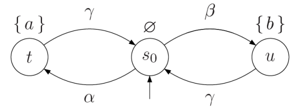


Positional policy $\mathfrak{S}_\alpha$ always chooses α in state s_0

Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Consider the MDP:

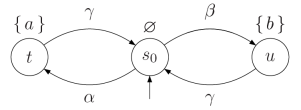


Positional policy $\mathfrak{S}_\alpha$ always chooses α in state s_0
Positional policy $\mathfrak{S}_\beta$ always chooses β in state s_0.

Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Consider the MDP:



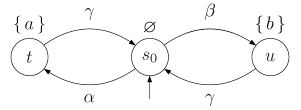
Positional policy $\mathfrak{S}_\alpha$ always chooses α in state s_0
Positional policy $\mathfrak{S}_\beta$ always chooses β in state s_0. Then:

$$Pr_{\mathfrak{S}_\alpha}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; Pr_{\mathfrak{S}_\beta}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; 0.$$

Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Consider the MDP:



Positional policy $\mathfrak{S}_\alpha$ always chooses α in state s_0
Positional policy $\mathfrak{S}_\beta$ always chooses β in state s_0. Then:

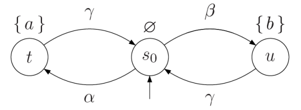
$$Pr_{\mathfrak{S}_\alpha}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; Pr_{\mathfrak{S}_\beta}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; 0.$$

Now consider fm-policy $\mathfrak{S}_{\alpha\beta}$ which alternates between selecting α and β.

Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Consider the MDP:



Positional policy $\mathfrak{S}_\alpha$ always chooses α in state s_0
Positional policy $\mathfrak{S}_\beta$ always chooses β in state s_0. Then:

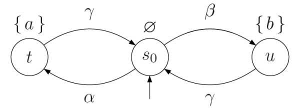
$$Pr_{\mathfrak{S}_\alpha}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; Pr_{\mathfrak{S}_\beta}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; 0.$$

Now consider fm-policy $\mathfrak{S}_{\alpha\beta}$ which alternates between selecting α and β.
Then: $Pr_{\mathfrak{S}_{\alpha\beta}}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; 1$.

Positional versus fm-policies

Positional policies are insufficient for ω-regular properties

Consider the MDP:



Positional policy $\mathfrak{S}_\alpha$ always chooses α in state s_0
Positional policy $\mathfrak{S}_\beta$ always chooses β in state s_0. Then:

$$Pr_{\mathfrak{S}_\alpha}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; Pr_{\mathfrak{S}_\beta}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; 0.$$

Now consider fm-policy $\mathfrak{S}_{\alpha\beta}$ which alternates between selecting α and β.
Then: $Pr_{\mathfrak{S}_{\alpha\beta}}(s_0 \models \Diamond a \wedge \Diamond b) \;=\; 1.$

Thus, the class of positional policies is insufficiently powerful to characterise minimal (or maximal) probabilities for ω-regular properties.

Other kinds of policies

- Counting policies that base their decision on the number of visits to a state, or the length of the history (i.e., number of visits to all states)

- Partial-observation policies that base their decision on the trace $L(s_0) \ldots L(s_n)$ of the history $s_0 \ldots s_n$.

- Randomised policies. This is applicable to all (deterministic) policies.

  For instance, a randomised positional policy $\mathfrak{S} : S \rightarrow Dist(Act)$, where $Dist(X)$ is the set of probability distributions on X, such that $\mathfrak{S}(s)(\alpha) > 0$ iff $\alpha \in Act(s)$. Similar can be done for fm-policies and history-dependent policies etc..

- There is a strict hierarchy of policies, showing their expressiveness (black board).

Overview

1. Markov Decision Processes
2. Policies
   - Positional policies
   - Finite-memory policies
3. Reachability probabilities
   - Mathematical characterisation
   - Value iteration
   - Linear programming
   - Policy iteration
4. Summary

Reachability probabilities

Reachability probabilities

Reachability probabilities

Reachability probabilities

Let $\mathcal{M}$ be an MDP with state space S and $\mathfrak{S}$ be a policy on $\mathcal{M}$. The reachability probability of $G \subseteq S$ from state $s \in S$ under policy $\mathfrak{S}$ is:

$$Pr^{\mathfrak{S}}(s \models \Diamond G) \; = \; Pr_s^{\mathcal{M}_{\mathfrak{S}}}\{\, \pi \in Paths(s) \mid \pi \models \Diamond G \,\}$$

Reachability probabilities

Reachability probabilities

Let $\mathcal{M}$ be an MDP with state space S and $\mathfrak{S}$ be a policy on $\mathcal{M}$. The reachability probability of $G \subseteq S$ from state $s \in S$ under policy $\mathfrak{S}$ is:

$$Pr^{\mathfrak{S}}(s \models \Diamond G) \;=\; Pr_s^{\mathcal{M}_{\mathfrak{S}}}\{\, \pi \in Paths(s) \mid \pi \models \Diamond G \,\}$$

Maximal and minimal reachability probabilities

The minimal reachability probability of $G \subseteq S$ from $s \in S$ is:

$$Pr^{\min}(s \models \Diamond G) = \inf_{\mathfrak{S}} Pr^{\mathfrak{S}}(s \models \Diamond G)$$

Reachability probabilities

Reachability probabilities

Let $\mathcal{M}$ be an MDP with state space S and $\mathfrak{S}$ be a policy on $\mathcal{M}$. The reachability probability of $G \subseteq S$ from state $s \in S$ under policy $\mathfrak{S}$ is:

$$Pr^{\mathfrak{S}}(s \models \lozenge G) \;=\; Pr_s^{\mathcal{M}_{\mathfrak{S}}}\{\, \pi \in Paths(s) \mid \pi \models \lozenge G \,\}$$

Maximal and minimal reachability probabilities

The minimal reachability probability of $G \subseteq S$ from $s \in S$ is:

$$Pr^{\min}(s \models \lozenge G) = \inf_{\mathfrak{S}} Pr^{\mathfrak{S}}(s \models \lozenge G)$$

In a similar way, the maximal reachability probability of $G \subseteq S$ is:

$$Pr^{\max}(s \models \lozenge G) \;=\; \sup_{\mathfrak{S}} Pr^{\mathfrak{S}}(s \models \lozenge G).$$

where policy $\mathfrak{S}$ ranges over all, infinitely (countably) many, policies.

Examples

Maximal reachability probabilities

MInimal guarantees for safety properties

Reasoning about the maximal probabilities for $\Diamond G$ is needed, e.g., for showing that $Pr^{\mathfrak{S}}(s \models \Diamond G) \leqslant \varepsilon$ for all policies $\mathfrak{S}$ and some small upper bound $0 < \varepsilon \leqslant 1$.

Maximal reachability probabilities

MInimal guarantees for safety properties

Reasoning about the maximal probabilities for $\Diamond G$ is needed, e.g., for showing that $Pr^{\mathfrak{S}}(s \models \Diamond G) \leqslant \varepsilon$ for all policies $\mathfrak{S}$ and some small upper bound $0 < \varepsilon \leqslant 1$. Then:

$$Pr^{\mathfrak{S}}(s \models \Box \neg G) \;\geqslant\; 1 - \varepsilon \quad \text{for all policies } \mathfrak{S}.$$

Maximal reachability probabilities

MInimal guarantees for safety properties

Reasoning about the maximal probabilities for $\Diamond G$ is needed, e.g., for showing that $Pr^{\mathfrak{S}}(s \models \Diamond G) \leqslant \varepsilon$ for all policies $\mathfrak{S}$ and some small upper bound $0 < \varepsilon \leqslant 1$. Then:

$$Pr^{\mathfrak{S}}(s \models \Box \neg G) \;\geqslant\; 1 - \varepsilon \quad \text{for all policies } \mathfrak{S}.$$

The task to compute $Pr^{\max}(s \models \Diamond G)$ can thus be understood as showing that a safety property (namely $\Box \neg G$) holds with sufficiently large probability, viz. $1 - \varepsilon$, regardless of the resolution of nondeterminism.

Equation system for max-reach probabilities

[1]Richard Bellman, an american mathematician (1920–1984), also known from the Bellman-Form shortest path algorithm.

Equation system for max-reach probabilities

Equation system for max-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$.

[1]Richard Bellman, an american mathematician (1920–1984), also known from the Bellman-Form shortest path algorithm.

Equation system for max-reach probabilities

Equation system for max-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\max}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.

[1] Richard Bellman, an american mathematician (1920–1984), also known from the Bellman-Form shortest path algorithm.

Equation system for max-reach probabilities

Equation system for max-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\max}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.
- If $s \not\models \exists \Diamond G$, then $x_s = 0$.

[1]Richard Bellman, an american mathematician (1920–1984), also known from the Bellman-Form shortest path algorithm.

Equation system for max-reach probabilities

Equation system for max-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\max}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.
- If $s \not\models \exists \Diamond G$, then $x_s = 0$.
- If $s \models \exists \Diamond G$ and $s \notin G$, then

$$x_s = \max \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t \;\middle|\; \alpha \in Act(s) \right\}$$

[1] Richard Bellman, an american mathematician (1920–1984), also known from the Bellman-Form shortest path algorithm.

Equation system for max-reach probabilities

Equation system for max-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\max}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.
- If $s \not\models \exists \Diamond G$, then $x_s = 0$.
- If $s \models \exists \Diamond G$ and $s \notin G$, then

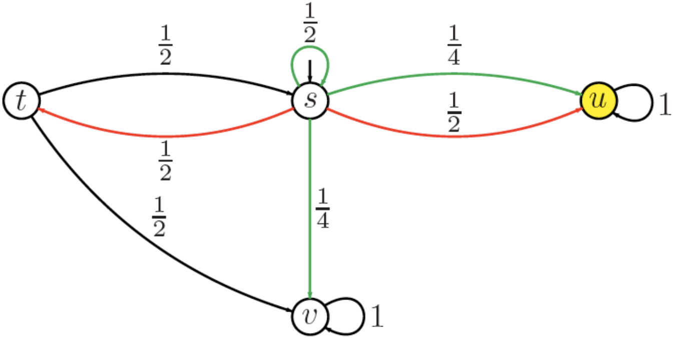
$$x_s \;=\; \max \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t \;\middle|\; \alpha \in Act(s) \right\}$$

This is a Bellman [1] equation as used in dynamic programming.

[1]Richard Bellman, an american mathematician (1920–1984), also known from the Bellman-Form shortest path algorithm.

Example

Example



equation system for reachability objective $\Diamond\{ u \}$ is:

$x_u = 1$ and $x_v = 0$

$$x_s \; = \; \max\{ \tfrac{1}{2}x_s + \tfrac{1}{4}x_u + \tfrac{1}{4}x_v, \tfrac{1}{2}x_u + \tfrac{1}{2}x_t \} \quad \text{and} \quad x_t = \tfrac{1}{2}x_s + \tfrac{1}{2}x_v$$

Value iteration

Value iteration

The previous theorem suggests to calculate the values

$$x_s = Pr^{\max}(s \models \Diamond G)$$

by successive approximation.

Value iteration

The previous theorem suggests to calculate the values

$$x_s = Pr^{\max}(s \models \Diamond G)$$

by successive approximation.

For the states $s \models \exists \Diamond G$ and $s \notin G$, we have $x_s = \lim_{n \to \infty} x_s^{(n)}$

Value iteration

The previous theorem suggests to calculate the values

$$x_s = Pr^{\max}(s \models \Diamond G)$$

by successive approximation.

For the states $s \models \exists \Diamond G$ and $s \not\in G$, we have $x_s = \lim_{n\to\infty} x_s^{(n)}$ where

$$x_s^{(0)} = 0 \quad \text{and} \quad x_s^{(n+1)} = \max \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t^{(n)} \mid \alpha \in Act(s) \right\}.$$

Value iteration

The previous theorem suggests to calculate the values

$$x_s = Pr^{\max}(s \models \Diamond G)$$

by successive approximation.

For the states $s \models \exists \Diamond G$ and $s \not\in G$, we have $x_s = \lim_{n \to \infty} x_s^{(n)}$ where

$$x_s^{(0)} = 0 \quad \text{and} \quad x_s^{(n+1)} = \max \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t^{(n)} \;\middle|\; \alpha \in Act(s) \right\}.$$

Note that $x_s^{(0)} \leqslant x_s^{(1)} \leqslant x_s^{(2)} \leqslant \ldots$.

Value iteration

The previous theorem suggests to calculate the values

$$x_s = Pr^{\max}(s \models \lozenge G)$$

by successive approximation.

For the states $s \models \exists \lozenge G$ and $s \notin G$, we have $x_s = \lim_{n \to \infty} x_s^{(n)}$ where

$$x_s^{(0)} = 0 \quad \text{and} \quad x_s^{(n+1)} = \max \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t^{(n)} \mid \alpha \in Act(s) \right\}.$$

Note that $x_s^{(0)} \leqslant x_s^{(1)} \leqslant x_s^{(2)} \leqslant \ldots$. Thus, the values $Pr^{\max}(s \models \lozenge G)$ can be approximated by successively computing the vectors

$$(x_s^{(0)}), (x_s^{(1)}), (x_s^{(2)}), \ldots,$$

until $\max_{s \in S} |x_s^{(n+1)} - x_s^{(n)}|$ is below a certain (typically very small) threshold.

Positional policies suffice for reach probabilities

Positional policies suffice for reach probabilities

Existence of optimal positional policies

Let $\mathcal{M}$ be a finite MDP with state space S, and $G \subseteq S$. There exists a positional policy $\mathfrak{S}$ such that for any $s \in S$ it holds:

$$Pr^{\mathfrak{S}}(s \models \Diamond G) \;=\; Pr^{\max}(s \models \Diamond G).$$

Positional policies suffice for reach probabilities

Existence of optimal positional policies

Let $\mathcal{M}$ be a finite MDP with state space S, and $G \subseteq S$. There exists a positional policy $\mathfrak{S}$ such that for any $s \in S$ it holds:

$$Pr^{\mathfrak{S}}(s \models \Diamond G) \; = \; Pr^{\max}(s \models \Diamond G).$$

Proof:

On the blackboard.

Equation system for min-reach probabilities

Equation system for min-reach probabilities

Equation system for min-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$.

Equation system for min-reach probabilities

Equation system for min-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\min}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.

Equation system for min-reach probabilities

Equation system for min-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\min}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.
- If $Pr^{\min}(s \models G) = 0$, then $x_s = 0$.

Equation system for min-reach probabilities

Equation system for min-reach probabilities

Let $\mathcal{M}$ be a finite MDP with state space S, $s \in S$ and $G \subseteq S$. The vector $(x_s)_{s \in S}$ with $x_s = Pr^{\min}(s \models \Diamond G)$ yields the unique solution of the following equation system:

- If $s \in G$, then $x_s = 1$.
- If $Pr^{\min}(s \models G) = 0$, then $x_s = 0$.
- If $Pr^{\min}(s \models G) > 0$ and $s \notin G$, then

$$x_s = \min \left\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t \mid \alpha \in Act(s) \right\}$$

Preprocessing

Preprocessing

The preprocessing required to compute the set

$$S^{\min}_{=0} \ = \ \{ \, s \in S \mid Pr^{\min}(s \models \Diamond G) \, \} = 0$$

can be performed by graph algorithms.

Preprocessing

The preprocessing required to compute the set

$$S^{\min}_{=0} \;=\; \{\, s \in S \mid Pr^{\min}(s \models \Diamond G) \,\} = 0$$

can be performed by graph algorithms. The set $S^{\min}_{=0}$ is given by $S \setminus T$ where

$$T = \bigcup_{n \geqslant 0} T_n$$

and $T_0 = G$ and, for $n \geqslant 0$:

$$T_{n+1} \;=\; T_n \cup \{\, s \in S \mid \forall \alpha \in Act(s)\, \exists t \in T_n.\, \mathbf{P}(s, \alpha, t) > 0 \,\}.$$

Preprocessing

The preprocessing required to compute the set

$$S^{\min}_{=0} \;=\; \{\, s \in S \mid Pr^{\min}(s \models \Diamond G) \,\} = 0$$

can be performed by graph algorithms. The set $S^{\min}_{=0}$ is given by $S \setminus T$ where

$$T = \bigcup_{n \geqslant 0} T_n$$

and $T_0 = G$ and, for $n \geqslant 0$:

$$T_{n+1} \;=\; T_n \cup \{\, s \in S \mid \forall \alpha \in Act(s)\, \exists t \in T_n.\, \mathbf{P}(s, \alpha, t) > 0 \,\}.$$

As $T_0 \subseteq T_1 \subseteq T_2 \subseteq \ldots \subseteq S$ and S is finite, the sequence $(T_n)_{n \geqslant 0}$ eventually stabilizes, i.e., for some $n \geqslant 0$, $T_n = T_{n+1} = \ldots = T$.

Preprocessing

The preprocessing required to compute the set

$$S^{\min}_{=0} \;=\; \{\, s \in S \mid Pr^{\min}(s \models \Diamond G) \,\} = 0$$

can be performed by graph algorithms. The set $S^{\min}_{=0}$ is given by $S \setminus T$ where

$$T = \bigcup_{n \geqslant 0} T_n$$

and $T_0 = G$ and, for $n \geqslant 0$:

$$T_{n+1} \;=\; T_n \cup \{\, s \in S \mid \forall \alpha \in Act(s)\, \exists t \in T_n.\, \mathbf{P}(s, \alpha, t) > 0 \,\}.$$

As $T_0 \subseteq T_1 \subseteq T_2 \subseteq \ldots \subseteq S$ and S is finite, the sequence $(T_n)_{n \geqslant 0}$ eventually stabilizes, i.e., for some $n \geqslant 0$, $T_n = T_{n+1} = \ldots = T$.

It follows: $Pr^{\min}(s \models \Diamond G) > 0$ if and only if $s \in T$.

Positional policies for min-reach probabilities

Positional policies for min-reach probabilities

Existence of optimal positional policies

Let $\mathcal{M}$ be a finite MDP with state space S, and $G \subseteq S$. There exists a positional policy $\mathfrak{S}$ such that for any $s \in S$ it holds:

$$Pr^{\mathfrak{S}}(s \models \Diamond G) \;=\; Pr^{\min}(s \models \Diamond G).$$

Positional policies for min-reach probabilities

Existence of optimal positional policies

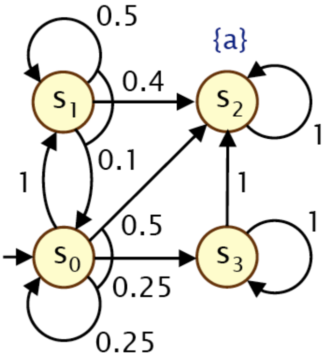
Let $\mathcal{M}$ be a finite MDP with state space S, and $G \subseteq S$. There exists a positional policy $\mathfrak{S}$ such that for any $s \in S$ it holds:

$$Pr^{\mathfrak{S}}(s \models \Diamond G) \; = \; Pr^{\min}(s \models \Diamond G).$$

Proof:

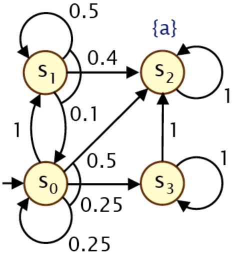
Similar to the case for maximal reachability probabilities.

Example value iteration



Determine $Pr^{\min}(s_i \models \Diamond\{ s_2 \})$.

Example value iteration

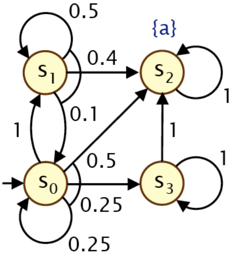


Determine

$Pr^{\min}(s_i \models \Diamond\{\, s_2 \,\})$

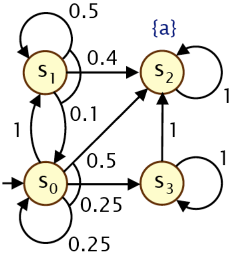
Example value iteration

1. $G = \{ s_2 \}, S^{\min}_{=0} = \{ s_3 \}, S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.



Determine
$Pr^{\min}(s_i \models \Diamond \{ s_2 \})$

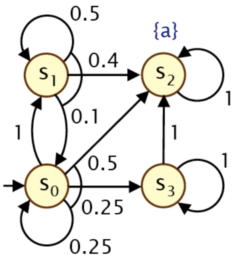
Example value iteration



Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

1. $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
2. $(x_s^{(0)}) = (0, 0, 1, 0)$

Example value iteration

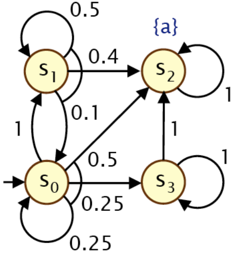


Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

1. $G = \{ s_2 \}, S^{\min}_{=0} = \{ s_3 \}, S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
2. $(x_s^{(0)}) = (0, 0, 1, 0)$
3. $(x_s^{(1)}) = (\min(1 \cdot 0, 0.25 \cdot 0 + 0.25 \cdot 0 + 0.5 \cdot 1),$

   $0.1 \cdot 0 + 0.5 \cdot 0 + 0.4 \cdot 1, 1, 0)$

Example value iteration



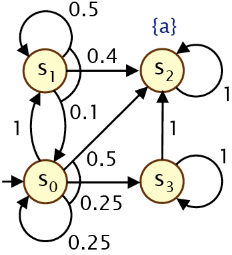
Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

1. $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
2. $(x_s^{(0)}) = (0, 0, 1, 0)$
3. $(x_s^{(1)}) = (\min(1{\cdot}0, 0.25{\cdot}0{+}0.25{\cdot}0{+}0.5{\cdot}1),$

$$0.1{\cdot}0{+}0.5{\cdot}0{+}0.4{\cdot}1, 1, 0)$$

$$= (0, 0.4, 1, 0)$$

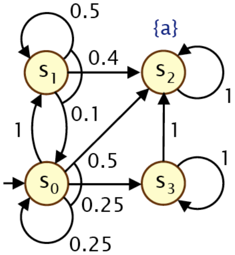
Example value iteration



Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

1. $G = \{ s_2 \}, S^{\min}_{=0} = \{ s_3 \}, S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
2. $(x_s^{(0)}) = (0, 0, 1, 0)$
3. $(x_s^{(1)}) = (\min(1{\cdot}0, 0.25{\cdot}0{+}0.25{\cdot}0{+}0.5{\cdot}1),$
   $0.1{\cdot}0{+}0.5{\cdot}0{+}0.4{\cdot}1, 1, 0)$
   $= (0, 0.4, 1, 0)$
4. $(x_s^{(2)}) = (\min(1{\cdot}0.4, 0.25{\cdot}0{+}0.25{\cdot}0{+}0.5{\cdot}1),$
   $0.1{\cdot}0{+}0.5{\cdot}0.4{+}0.4{\cdot}1, 1, 0)$

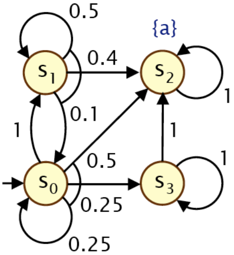
Example value iteration



Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

1. $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
2. $(x_s^{(0)}) = (0, 0, 1, 0)$
3. $(x_s^{(1)}) = (\min(1{\cdot}0, 0.25{\cdot}0{+}0.25{\cdot}0{+}0.5{\cdot}1),$
$0.1{\cdot}0{+}0.5{\cdot}0{+}0.4{\cdot}1, 1, 0)$
$= (0, 0.4, 1, 0)$
4. $(x_s^{(2)}) = (\min(1{\cdot}0.4, 0.25{\cdot}0{+}0.25{\cdot}0{+}0.5{\cdot}1),$
$0.1{\cdot}0{+}0.5{\cdot}0.4{+}0.4{\cdot}1, 1, 0)$
$= (0.4, 0.6, 1, 0)$

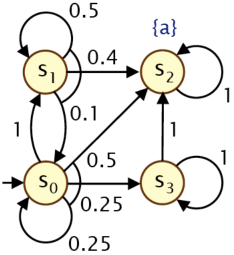
Example value iteration



Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

1. $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
2. $(x_s^{(0)}) = (0, 0, 1, 0)$
3. $(x_s^{(1)}) = (\min(1{\cdot}0, 0.25{\cdot}0 + 0.25{\cdot}0 + 0.5{\cdot}1), 0.1{\cdot}0 + 0.5{\cdot}0 + 0.4{\cdot}1, 1, 0)$
   $= (0, 0.4, 1, 0)$
4. $(x_s^{(2)}) = (\min(1{\cdot}0.4, 0.25{\cdot}0 + 0.25{\cdot}0 + 0.5{\cdot}1), 0.1{\cdot}0 + 0.5{\cdot}0.4 + 0.4{\cdot}1, 1, 0)$
   $= (0.4, 0.6, 1, 0)$
5. $(x_s^{(3)}) = \ldots\ldots$

Example value iteration



Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

	$[x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)}]$
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]
...	
n=20:	[0.666667, 0.933332, 1, 0]
n=21:	[0.666667, 0.933332, 1, 0]
	$\approx$ [2/3, 14/15, 1, 0]

Optimal positional policy

Positional policies $\mathfrak{S}_{\text{min}}$ and $\mathfrak{S}_{\text{max}}$ thus yield:

$$
\begin{aligned}
Pr^{\mathfrak{S}_{\text{min}}}(s \models \Diamond G) &= Pr^{\text{min}}(s \models \Diamond G) \quad \text{for all states } s \in S \\
Pr^{\mathfrak{S}_{\text{max}}}(s \models \Diamond G) &= Pr^{\text{max}}(s \models \Diamond G) \quad \text{for all states } s \in S
\end{aligned}
$$

Optimal positional policy

Positional policies $\mathfrak{S}_{\min}$ and $\mathfrak{S}_{\max}$ thus yield:

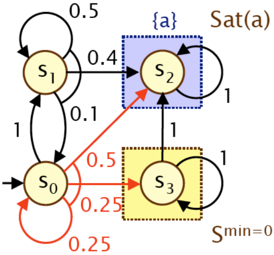
$$Pr^{\mathfrak{S}_{\min}}(s \models \lozenge G) = Pr^{\min}(s \models \lozenge G) \quad \text{for all states } s \in S$$
$$Pr^{\mathfrak{S}_{\max}}(s \models \lozenge G) = Pr^{\max}(s \models \lozenge G) \quad \text{for all states } s \in S$$

These policies are obtained as follows:

$$\mathfrak{S}_{\min}(s) = \arg\min\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot Pr^{\min}(t \models \lozenge G) \mid \alpha \in Act \}$$

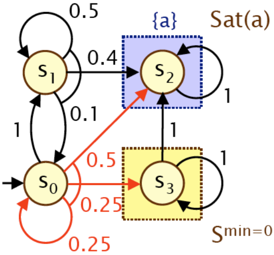
$$\mathfrak{S}_{\max}(s) = \arg\max\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot Pr^{\max}(t \models \lozenge G) \mid \alpha \in Act \}$$

Optimal positional policy

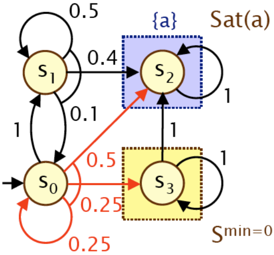


Optimal positional policy

- Outcome of the value iteration $(\, x_s \,) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$

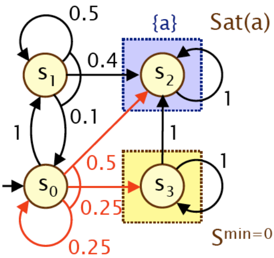


Optimal positional policy



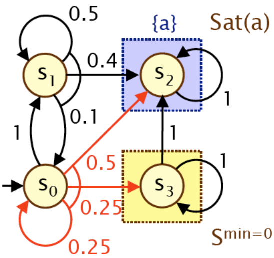
- Outcome of the value iteration $(\, x_s \,) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$
- How to obtain the optimal policy from this result?

Optimal positional policy



- Outcome of the value iteration $(\, x_s \,) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$
- How to obtain the optimal policy from this result?
- $x_{s_0} = \min(1 \cdot \frac{14}{15}, 0.5 \cdot 1 + 0.25 \cdot 0 + 0.25 \cdot \frac{2}{3})$

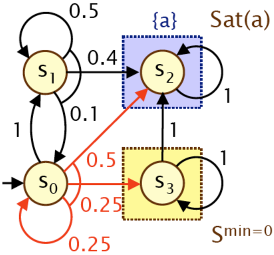
Optimal positional policy



- Outcome of the value iteration $(x_s) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$
- How to obtain the optimal policy from this result?
- $x_{s_0} = \min(1 \cdot \frac{14}{15}, 0.5 \cdot 1 + 0.25 \cdot 0 + 0.25 \cdot \frac{2}{3})$

  $\min(\frac{14}{15}, \frac{2}{3})$

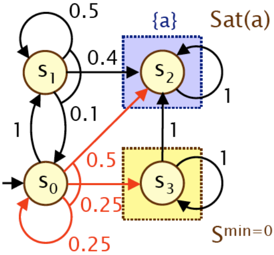
Optimal positional policy



- Outcome of the value iteration $(\, x_s \,) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$
- How to obtain the optimal policy from this result?
- $x_{s_0} = \min(1\cdot\frac{14}{15}, 0.5\cdot 1 + 0.25\cdot 0 + 0.25\cdot\frac{2}{3})$

  $\min(\frac{14}{15}, \frac{2}{3})$
- Thus the optimal policy always selects red in s_0

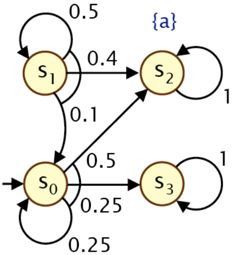
Optimal positional policy



- Outcome of the value iteration $(\, x_s \,) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$
- How to obtain the optimal policy from this result?
- $x_{s_0} = \min(1 \cdot \frac{14}{15}, 0.5 \cdot 1 + 0.25 \cdot 0 + 0.25 \cdot \frac{2}{3})$

  $\min(\frac{14}{15}, \frac{2}{3})$
- Thus the optimal policy always selects red in s_0
- Note that the minimal reach-probability is unique; the optimal policy need not to be unique.

Induced DTMC



- Outcome of the value iteration $(x_s) = (\frac{2}{3}, \frac{14}{15}, 1, 0)$
- How to obtain the optimal policy from this results?
- $x_{s_0} = \min(1 \cdot \frac{14}{15}, 0.5 \cdot 1 + 0.5 \cdot 0 + 0.25 \cdot \frac{2}{3})$

  $\min(\frac{14}{15}, \frac{2}{3})$
- Thus the optimal policy always selects red.

An alternative approach

A viable alternative to value iteration is linear programming.

Linear programming

Linear programming

Linear programming

Optimisation of a linear objective function subject to linear (in)equalities.

Linear programming

Linear programming

Optimisation of a linear objective function subject to linear (in)equalities.

Let $x_1, \ldots, x_n$ be non-negative real-valued variables. Maximise (or minimise) the objective function:

$$c_1 \cdot x_1 + c_2 \cdot x_2 + \ldots + c_n \cdot x_n \quad \text{for constants } c_1, \ldots, c_n \in \mathbb{R}$$

Linear programming

Linear programming

Optimisation of a linear objective function subject to linear (in)equalities.

Let $x_1, \ldots, x_n$ be non-negative real-valued variables. Maximise (or minimise) the objective function:

$$c_1 \cdot x_1 + c_2 \cdot x_2 + \ldots + c_n \cdot x_n \quad \text{for constants } c_1, \ldots, c_n \in \mathbb{R}$$

subject to the constraints

$$\begin{aligned} a_{11} \cdot x_1 + a_{12} \cdot x_2 + \ldots + a_{1n} \cdot x_n &\leqslant b_1 \\ \ldots \ldots & \\ a_{m1} \cdot x_1 + a_{m2} \cdot x_2 + \ldots + a_{mn} \cdot x_n &\leqslant b_m. \end{aligned}$$

Solution techniques: e.g., Simplex, ellipsoid method, interior point method.

Maximal reach probabilities as a linear program

Maximal reach probabilities as a linear program

Linear program for max-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\max}(s \models \lozenge G)$ are the unique solution of the *linear program*:

Maximal reach probabilities as a linear program

Linear program for max-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\max}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.

Maximal reach probabilities as a linear program

Linear program for max-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\max}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.
- If $s \not\models \exists \Diamond G$, then $x_s = 0$.

Maximal reach probabilities as a linear program

Linear program for max-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\max}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.
- If $s \not\models \exists \Diamond G$, then $x_s = 0$.
- If $s \models \exists \Diamond G$ and $s \notin G$, then $0 \leqslant x_s \leqslant 1$ and for all $\alpha \in Act(s)$:

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

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

Maximal reach probabilities as a linear program

Linear program for max-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\max}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.
- If $s \not\models \exists \Diamond G$, then $x_s = 0$.
- If $s \models \exists \Diamond G$ and $s \notin G$, then $0 \leqslant x_s \leqslant 1$ and for all $\alpha \in Act(s)$:

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

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

Proof:

See lecture notes.

Minimal reach probabilities as a linear program

Minimal reach probabilities as a linear program

Linear program for min-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\min}(s \models \Diamond G)$ are the unique solution of the *linear program*:

Minimal reach probabilities as a linear program

Linear program for min-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\min}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.

Minimal reach probabilities as a linear program

Linear program for min-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\min}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.
- If $Pr^{\min}(s \models \Diamond G) = 0$, then $x_s = 0$.

Minimal reach probabilities as a linear program

Linear program for min-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\min}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.
- If $Pr^{\min}(s \models \Diamond G) = 0$, then $x_s = 0$.
- If $Pr^{\min}(s \models \Diamond G) > 0$ and $s \notin G$ then $0 \leqslant x_s \leqslant 1$ and for all $\alpha \in Act(s)$:

$$x_s \leqslant \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t$$

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

Minimal reach probabilities as a linear program

Linear program for min-reach probabilities

Consider a finite MDP with state space S, and $G \subseteq S$. The values $x_s = Pr^{\min}(s \models \Diamond G)$ are the unique solution of the *linear program*:

- If $s \in G$, then $x_s = 1$.
- If $Pr^{\min}(s \models \Diamond G) = 0$, then $x_s = 0$.
- If $Pr^{\min}(s \models \Diamond G) > 0$ and $s \notin G$ then $0 \leqslant x_s \leqslant 1$ and for all $\alpha \in Act(s)$:

$$x_s \leqslant \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t$$

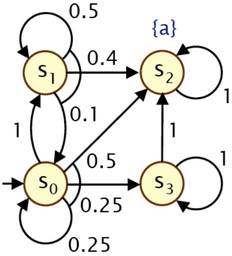
where $\sum_{s \in S} x_s$ is maximal.

Proof:

See lecture notes.

Example linear programming

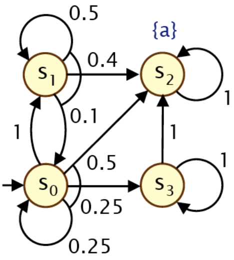
Example linear programming



Determine

$Pr^{\min}(s_i \models \Diamond \{ s_2 \})$

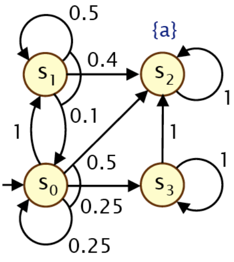
Example linear programming



Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

- $G = \{ s_2 \}, S^{\min}_{=0} = \{ s_3 \}, S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.

Example linear programming

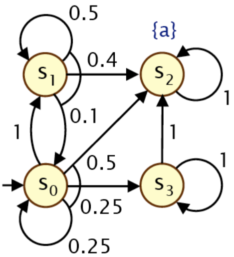


Determine
$Pr^{\min}(s_i \models \Diamond\{ s_2 \})$

- $G = \{ s_2 \}, S_{=0}^{\min} = \{ s_3 \}, S \setminus (G \cup S_{=0}^{\min}) = \{ s_0, s_1 \}$.
- Maximise $x_0 + x_1$ subject to the constraints:

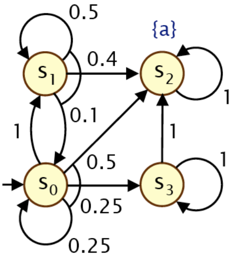
$$\begin{aligned} x_0 &\leqslant x_1 \\ x_0 &\leqslant \tfrac{1}{4} \cdot x_0 + \tfrac{1}{2} \\ x_1 &\leqslant \tfrac{1}{10} \cdot x_0 + \tfrac{1}{2} \cdot x_1 + \tfrac{2}{5} \end{aligned}$$

Example linear programming



- $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.

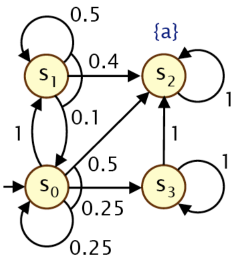
Example linear programming



- $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
- Maximise $x_0 + x_1$ subject to the constraints:

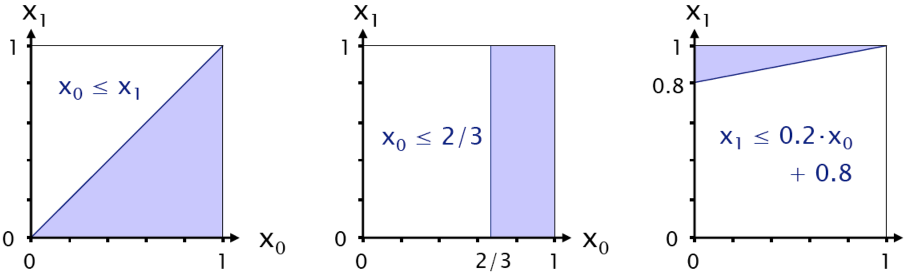
$$
\begin{aligned}
x_0 &\leqslant x_1 \\
x_0 &\leqslant \tfrac{2}{3} \\
x_1 &\leqslant \tfrac{1}{5} \cdot x_0 + \tfrac{4}{5}
\end{aligned}
$$

Example linear programming

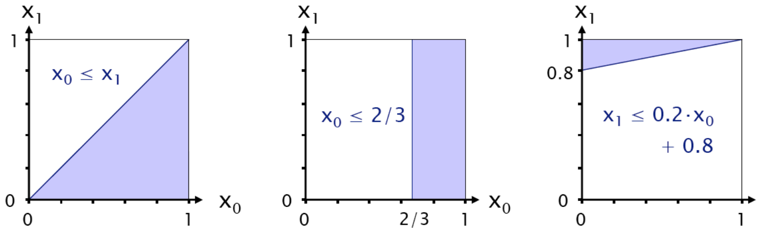


- $G = \{ s_2 \}$, $S^{\min}_{=0} = \{ s_3 \}$, $S \setminus (G \cup S^{\min}_{=0}) = \{ s_0, s_1 \}$.
- Maximise $x_0 + x_1$ subject to the constraints:

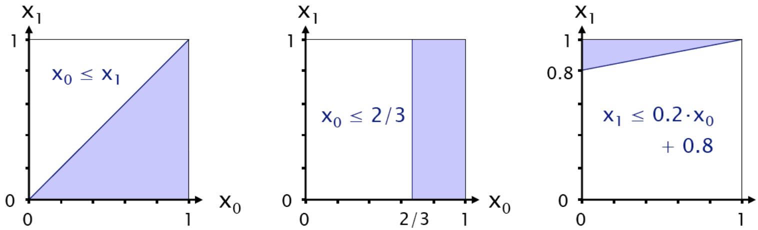
$$\begin{aligned} x_0 &\leqslant x_1 \\ x_0 &\leqslant \tfrac{2}{3} \\ x_1 &\leqslant \tfrac{1}{5} \cdot x_0 + \tfrac{4}{5} \end{aligned}$$

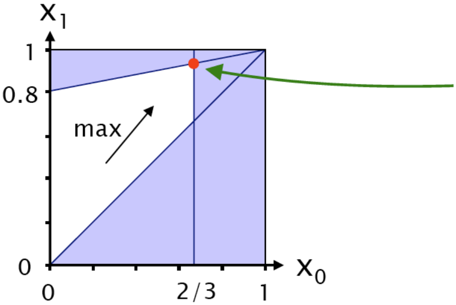


Example linear programming



Example linear programming





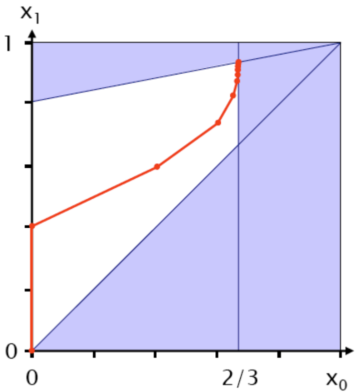
Solution:

(x_0, x_1)

$=$

$(2/3, 14/15)$

Value iteration vs. linear programming



	$[\, x_0^{(n)}, x_1^{(n)}, x_2^{(n)}, x_3^{(n)} \,]$
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]
...	
n=20:	[0.666667, 0.933332, 1, 0]
n=21:	[0.666667, 0.933332, 1, 0]
	≈ [2/3, 14/15, 1, 0]

This curve shows how the value iteration approach approximates the solution.

Time complexity

Time complexity

For finite MDP $\mathcal{M}$ with state space S, $G \subseteq S$ and $s \in S$, the values $Pr^{\max}(s \models \Diamond G)$ can be computed in time polynomial in the size of $\mathcal{M}$. The same holds for $Pr^{\min}(s \models \Diamond G)$.

Time complexity

Time complexity

For finite MDP $\mathcal{M}$ with state space S, $G \subseteq S$ and $s \in S$, the values $Pr^{\max}(s \models \Diamond G)$ can be computed in time polynomial in the size of $\mathcal{M}$. The same holds for $Pr^{\min}(s \models \Diamond G)$.

Proof:

Thanks to the characterisation as a linear program and polynomial time techniques to solve such linear programs such as ellipsoid methods.

Time complexity

Time complexity

For finite MDP $\mathcal{M}$ with state space S, $G \subseteq S$ and $s \in S$, the values $Pr^{\max}(s \models \Diamond G)$ can be computed in time polynomial in the size of $\mathcal{M}$. The same holds for $Pr^{\min}(s \models \Diamond G)$.

Proof:

Thanks to the characterisation as a linear program and polynomial time techniques to solve such linear programs such as ellipsoid methods.

Corollary

For finite MDPs, the question whether $Pr^{\mathfrak{S}}(s \models \Diamond G) \leqslant p$ for some rational $p \in [0, 1[$ is decidable in polynomial time.

Yet another alternative approach

A viable alternative to value iteration and linear programming is policy iteration.

Policy iteration

Value iteration

In value iteration, we iteratively attempt to improve the minimal (or maximal) reachability probabilities by starting with an underestimation, viz. zero for all states.

Policy iteration

Value iteration

In value iteration, we iteratively attempt to improve the minimal (or maximal) reachability probabilities by starting with an underestimation, viz. zero for all states.

Policy iteration

In policy iteration, the idea is to start with an arbitrary positional policy and improve it for each state in a step-by-step fashion, so as to determine the optimal one.

Policy iteration

Policy iteration

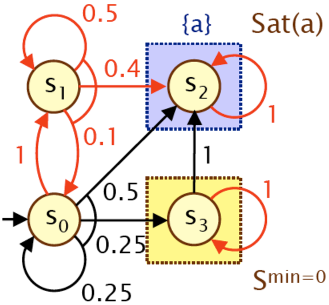
1. Start with an arbitrary positional policy $\mathfrak{S}$ that selects some $\alpha \in Act(s)$ for each state $s \in S \setminus G \cup S^{\min}_{=0}$.
2. Compute the reachability probabilities $Pr^{\mathfrak{S}}(s \models \lozenge G)$. This amounts to solving a linear equation system on DTMC $\mathcal{M}_{\mathfrak{S}}$.
3. Improve the policy in every state according to the following rules:

$$\begin{aligned}\mathfrak{S}^{(i+1)}(s) &= \arg\min\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot Pr^{\mathfrak{S}^{(i)}}(t \models \lozenge G) \mid \alpha \in Act \} \text{ or} \\ \mathfrak{S}^{(i+1)}(s) &= \arg\max\{ \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot Pr^{\mathfrak{S}^{(i)}}(t \models \lozenge G) \mid \alpha \in Act \}\end{aligned}$$

4. Repeat steps 2. and 3. until the policy does not change.
5. Termination[2]: finite number of states and improvement of min/max probabilities each time.

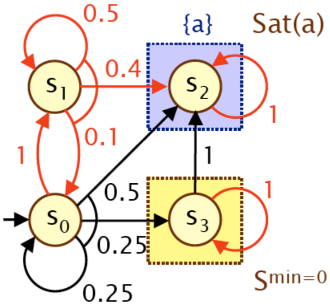
[2] For a proof, see Section 6.7 of the book by Tijms on A First Course in Stochastic

Policy iteration: example



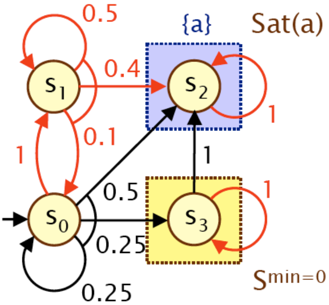
- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.

Policy iteration: example



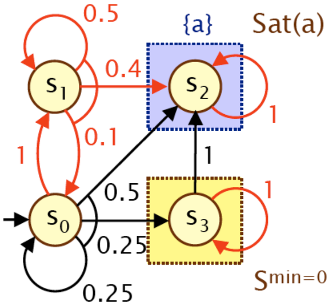
- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \Diamond G)$ for all i.

Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \lozenge G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

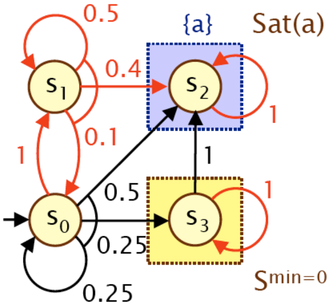
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = x_1$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.

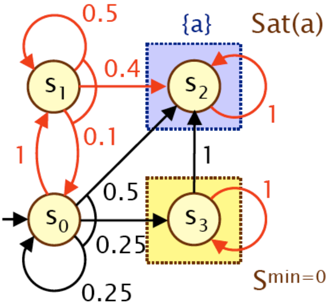
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = x_1$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.
- This yields $x_0 = x_1 = x_2 = 1$ and $x_3 = 0$.

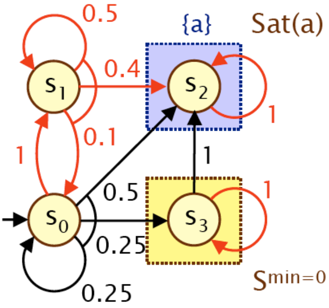
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = x_1$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.
- This yields $x_0 = x_1 = x_2 = 1$ and $x_3 = 0$.
- Change policy $\mathfrak{S}$ in s_0, yielding policy $\mathfrak{S}'$.

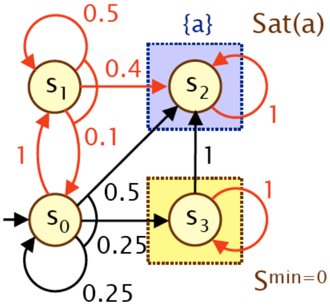
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = x_1$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.
- This yields $x_0 = x_1 = x_2 = 1$ and $x_3 = 0$.
- Change policy $\mathfrak{S}$ in s_0, yielding policy $\mathfrak{S}'$.
- This yields $\min(1 \cdot 1, \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 0 + \frac{1}{4} \cdot 1)$

Policy iteration: example

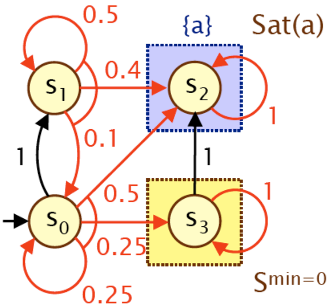


- Let $G = \{ s_2 \}$.
- Consider an arbitrary policy $\mathfrak{S}$.
- Compute $x_i = Pr^{\mathfrak{S}}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = x_1$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.
- This yields $x_0 = x_1 = x_2 = 1$ and $x_3 = 0$.
- Change policy $\mathfrak{S}$ in s_0, yielding policy $\mathfrak{S}'$.
- This yields $\min(1 \cdot 1, \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 0 + \frac{1}{4} \cdot 1)$

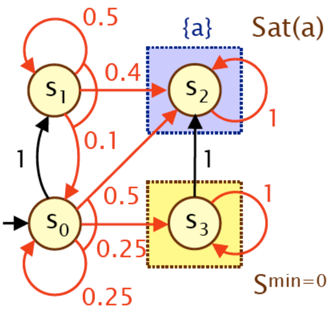
  that is, $\min(1, \frac{3}{4}) \; = \; \frac{3}{4}$.

Policy iteration: example



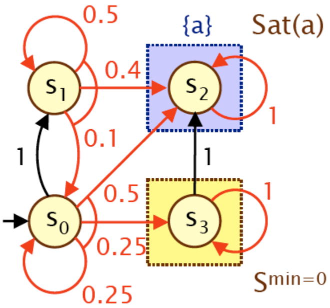
- Let $G = \{ s_2 \}$.
- Consider the adapted policy $\mathfrak{S}'$.

Policy iteration: example



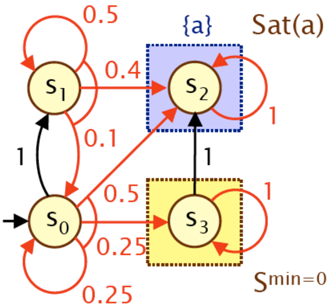
- Let $G = \{ s_2 \}$.
- Consider the adapted policy $\mathfrak{S}'$.
- Compute $x_i = Pr^{\mathfrak{S}'}(s_i \models \Diamond G)$ for all i.

Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider the adapted policy $\mathfrak{S}'$.
- Compute $x_i = Pr^{\mathfrak{S}'}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

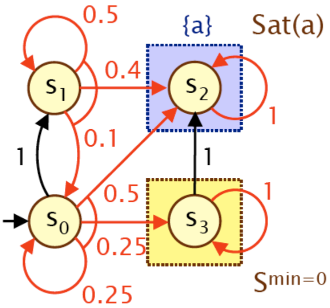
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider the adapted policy $\mathfrak{S}'$.
- Compute $x_i = Pr^{\mathfrak{S}'}(s_i \models \lozenge G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = \frac{1}{4} \cdot x_0 + \frac{1}{2}$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.

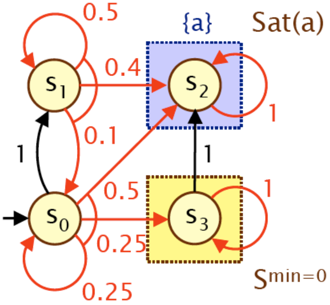
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider the adapted policy $\mathfrak{S}'$.
- Compute $x_i = Pr^{\mathfrak{S}'}(s_i \models \lozenge G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = \frac{1}{4} \cdot x_0 + \frac{1}{2}$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.
- This yields $x_0 = \frac{2}{3}$, $x_1 = \frac{14}{15}$, $x_2 = 1$ and $x_3 = 0$.

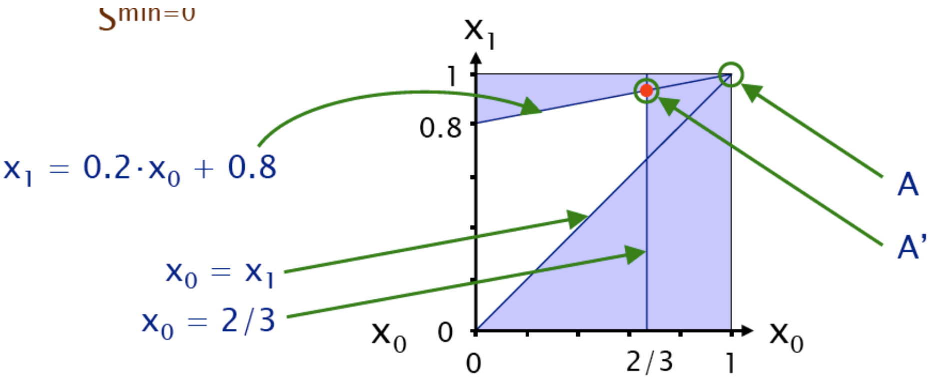
Policy iteration: example



- Let $G = \{ s_2 \}$.
- Consider the adapted policy $\mathfrak{S}'$.
- Compute $x_i = Pr^{\mathfrak{S}'}(s_i \models \Diamond G)$ for all i.
- Then: $x_2 = 1$, $x_3 = 0$,

  and $x_0 = \frac{1}{4} \cdot x_0 + \frac{1}{2}$, $x_1 = \frac{1}{10} \cdot x_0 + \frac{1}{2} \cdot x_1 + \frac{2}{5}$.
- This yields $x_0 = \frac{2}{3}$, $x_1 = \frac{14}{15}$, $x_2 = 1$ and $x_3 = 0$.
- This policy is optimal.

Graphical representation of policy iteration



where A denotes policy $\mathfrak{S}$ and A' policy $\mathfrak{S}'$.

Overview

Summary

Summary

Important points

Summary

Important points

1. Maximal reachability probabilities are suprema over reachability probabilities for all, potentially infinitely many, policies.

Summary

Important points

1. Maximal reachability probabilities are suprema over reachability probabilities for all, potentially infinitely many, policies.
2. They are characterised by equation systems with maximal operators.

Summary

Important points

1. Maximal reachability probabilities are suprema over reachability probabilities for all, potentially infinitely many, policies.
2. They are characterised by equation systems with maximal operators.
3. There exists a positional policy that yields the maximal reachability probability.

Summary

Important points

1. Maximal reachability probabilities are suprema over reachability probabilities for all, potentially infinitely many, policies.
2. They are characterised by equation systems with maximal operators.
3. There exists a positional policy that yields the maximal reachability probability.
4. Such policies can be determined using value or policy iteration.

Summary

Important points

1. Maximal reachability probabilities are suprema over reachability probabilities for all, potentially infinitely many, policies.
2. They are characterised by equation systems with maximal operators.
3. There exists a positional policy that yields the maximal reachability probability.
4. Such policies can be determined using value or policy iteration.
5. Or, alternatively, in polynomial time using linear programming.

Summary

Important points

1. Maximal reachability probabilities are suprema over reachability probabilities for all, potentially infinitely many, policies.
2. They are characterised by equation systems with maximal operators.
3. There exists a positional policy that yields the maximal reachability probability.
4. Such policies can be determined using value or policy iteration.
5. Or, alternatively, in polynomial time using linear programming.
6. Positional policies are not powerful enough for arbitrary ω-regular properties.