Planning and Optimization

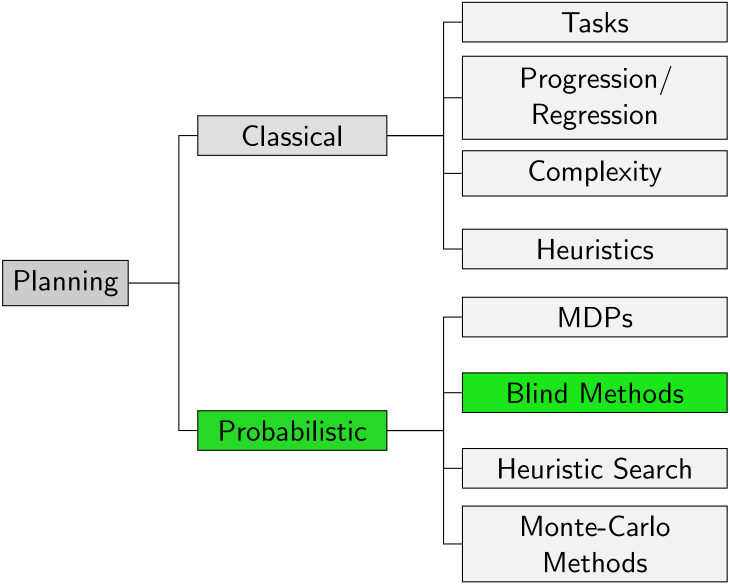
F3. Blind Methods: Policy Iteration

Gabriele Röger and Thomas Keller

Universität Basel

November 26, 2018

Content of this Course



Policy Evaluation

Expected Values under Uncertainty

Definition (Expected Value of a Random Variable)

Let V be a random variable with $n \in \mathbb{N}$ outcomes $d_1, \ldots, d_n \in \mathbb{R}$, and let d_i for $i = 1, \ldots, n$ occur with probability $p_i \in [0, 1]$ s.t. $\sum_{i=1}^{n} p_i = 1$.

The expected value of X is $\mathbb{E}[X] = \sum_{i=1}^{n}(p_i \cdot d_i)$.

Example: Expected Values under Uncertainty

Example

The expected payoff of placing one bet in Swiss Lotto for a cost of 2.50 with (simplified) payout structure

- $d_1 = 30.000.000$ with $p_1 = \frac{1}{31474716}$ (6+1)
- $d_2 = 1.000.000$ with $p_2 = \frac{1}{5245786}$ (6)
- $d_4 = 5.000$ with $p_4 = \frac{1}{850668}$ (5)
- $d_4 = 50$ with $p_4 = \frac{1}{111930}$ (4)
- $d_5 = 10$ with $p_5 = \frac{1}{11480}$ is (3)

$$\mathbb{E}[X] = (\frac{30000000}{31474716} + \frac{1000000}{5245786} + \frac{5000}{850668} + \frac{50}{111930} + \frac{10}{11480}) - 2.5 \approx -1.35.$$

Proper SSP Policy

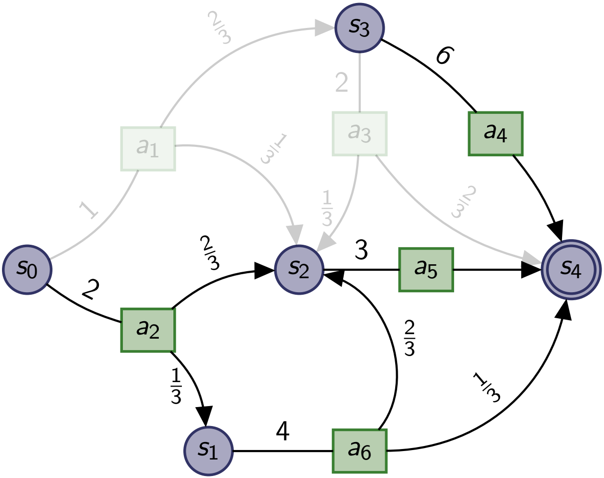
Definition (Proper SSP Policy)

Let $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ be an SSP and π be a policy for $\mathcal{T}$. π is proper if it reaches a goal state from each state with probability 1, i.e. if

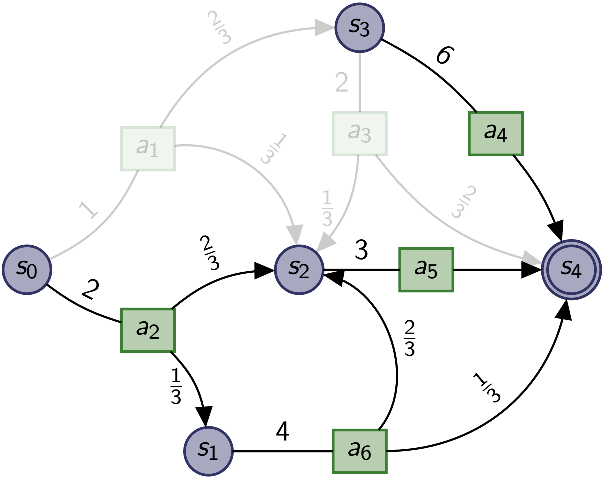
$$\sum_{s \xrightarrow{p_1:\ell_1} s', \ldots, s'' \xrightarrow{p_n:\ell_n} s_\star} \prod_{i=1}^{n} p_i = 1$$

for all states $s \in S$.

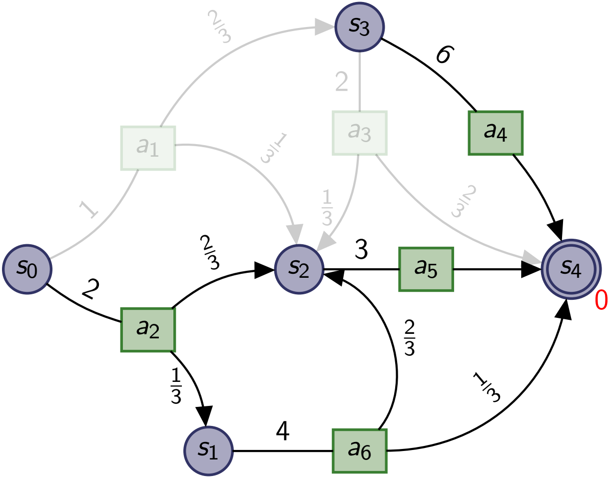
Example: Policy Evaluation for Proper SSP Policy



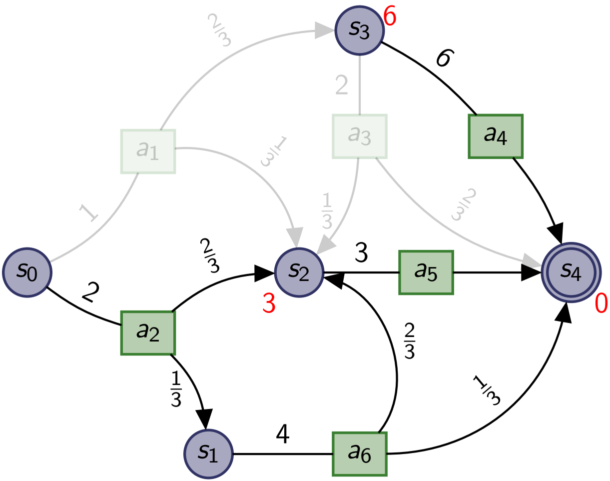
Example: Policy Evaluation for Acyclic Proper SSP Policy



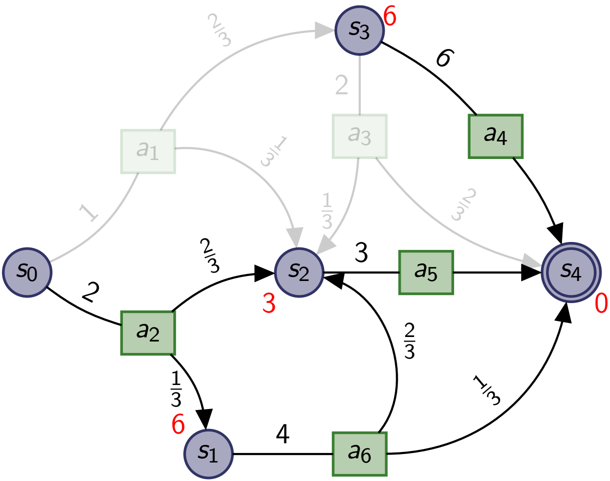
Example: Policy Evaluation for Acyclic Proper SSP Policy



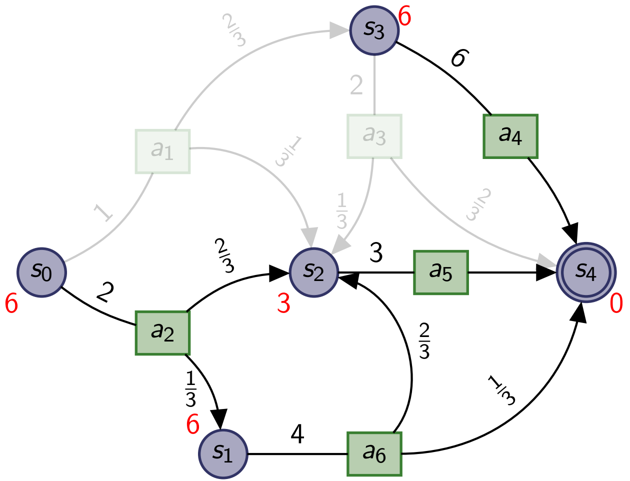
Example: Policy Evaluation for Acyclic Proper SSP Policy



Example: Policy Evaluation for Acyclic Proper SSP Policy



Example: Policy Evaluation for Acyclic Proper SSP Policy



Policy Evaluation for Acyclic Proper SSP Policy

Acyclic Policy Evaluation for SSP $\mathcal{T}$ and complete policy π

initialize $V_\pi(s) := \perp$ for all $s \in S$
while there is a $s \in S$ with $V_\pi(s) = \perp$:
  pick $s \in S$ with $V_\pi(s) = \perp$ and
    $V_\pi(s') \neq \perp$ for all $s' \in \mathrm{succ}(s, \pi(s))$
  set $V_\pi(s) := c(\pi(s)) + \sum_{s' \in \mathrm{succ}(s,\pi(s))} T(s, \pi(s), s') \cdot V_\pi(s')$
return V_π

Note: can be generalized to executable policies

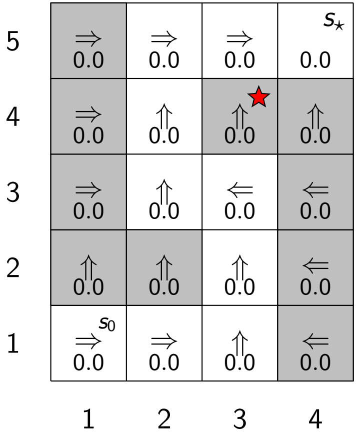
Iterative Policy Evaluation for SSPs

- impossible to compute state-values in one sweep over the state space in presence of cycles
- iterative refinment of $\hat{V}^{i-1}$ to $\hat{V}^{i}$ possible:

$$\hat{V}^{i}_{\pi}(s) = c(\pi(s)) + \sum_{s' \in \mathrm{succ}(s,\pi(s))} T(s,\pi(s),s') \cdot \hat{V}^{i-1}_{\pi}(s')$$

- iterative policy evaluation converges to the true state-values of proper π, i.e., $\lim_{i\to\infty} \hat{V}^{i}_{\pi} = V_{\pi}$
- converges regardless of $\hat{V}^{0}_{\pi}$

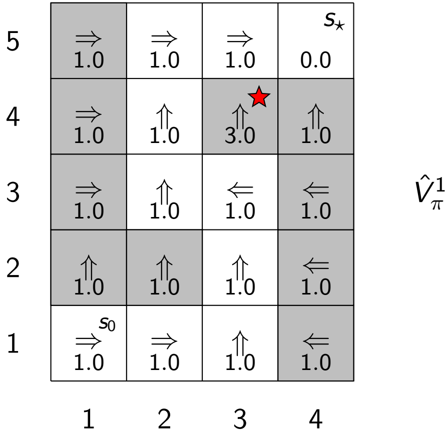
Example: Iterative Policy Evaluation for SSPs



$\hat{V}_\pi^0$

- cost of 1 for all actions except for moving away from (3,4) where cost is 3
- get stuck when moving away from gray cells with prob. 0.6

Example: Iterative Policy Evaluation for SSPs



- cost of 1 for all actions except for moving away from (3,4) where cost is 3
- get stuck when moving away from gray cells with prob. 0.6

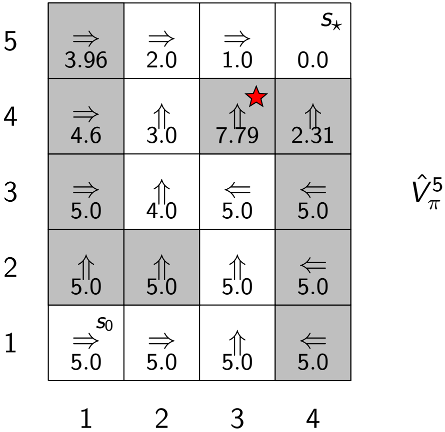
Example: Iterative Policy Evaluation for SSPs

5	⇒ 2.0	⇒ 2.0	⇒ 1.0	$s_\star$ 0.0
4	⇒ 2.0	⇑ 2.0	⇑ 5.2	⇑ 1.6
3	⇒ 2.0	⇑ 2.0	⇐ 2.0	⇐ 2.0
2	⇑ 2.0	⇑ 2.0	⇑ 2.0	⇐ 2.0
1	s_0 ⇒ 2.0	⇒ 2.0	⇑ 2.0	⇐ 2.0
	1	2	3	4

$\hat{V}_\pi^2$

- cost of 1 for all actions except for moving away from (3,4) where cost is 3
- get stuck when moving away from gray cells with prob. 0.6

Example: Iterative Policy Evaluation for SSPs



- cost of 1 for all actions except for moving away from (3,4) where cost is 3
- get stuck when moving away from gray cells with prob. 0.6

Example: Iterative Policy Evaluation for SSPs

5	$\Rightarrow$ 4.46	$\Rightarrow$ 2.0	$\Rightarrow$ 1.0	$s_\star$ 0.0
4	$\Rightarrow$ 5.43	$\Uparrow$ 3.0	$\Uparrow$ 8.44	$\Uparrow$ 2.49
3	$\Rightarrow$ 6.39	$\Uparrow$ 4.0	$\Leftarrow$ 5.0	$\Leftarrow$ 7.31
2	$\Uparrow$ 8.31	$\Uparrow$ 6.39	$\Uparrow$ 6.0	$\Leftarrow$ 8.18
1	s_0 $\Rightarrow$ 9.0	$\Rightarrow$ 8.0	$\Uparrow$ 7.0	$\Leftarrow$ 8.96
	1	2	3	4

$\hat{V}_\pi^{10}$

- cost of 1 for all actions except for moving away from (3,4) where cost is 3
- get stuck when moving away from gray cells with prob. 0.6

Example: Iterative Policy Evaluation for SSPs

	1	2	3	4
5	$\Rightarrow$ 4.49	$\Rightarrow$ 2.0	$\Rightarrow$ 1.0	$s_\star$ 0.0
4	$\Rightarrow$ 5.49	$\Uparrow$ 3.0	$\Uparrow$ 8.49 ★	$\Uparrow$ 2.49
3	$\Rightarrow$ 6.49	$\Uparrow$ 4.0	$\Leftarrow$ 5.0	$\Leftarrow$ 7.49
2	$\Uparrow$ 8.98	$\Uparrow$ 6.49	$\Uparrow$ 6.0	$\Leftarrow$ 8.49
1	s_0 $\Rightarrow$ 9.0	$\Rightarrow$ 8.0	$\Uparrow$ 7.0	$\Leftarrow$ 9.49

$\hat{V}_\pi^{18}$

- cost of 1 for all actions except for moving away from (3,4) where cost is 3
- get stuck when moving away from gray cells with prob. 0.6

Iterative Policy Evaluation

Iterative Policy Evaluation for SSP $\mathcal{T}$, policy π and $\epsilon > 0$

initialize $\hat{V}^0$ arbitarily
for $i = 1, 2, \ldots$:
  for all states $s \in S$:
    $\hat{V}^i_\pi(s) := c(\pi(s)) + \sum_{s' \in S} T(s, \pi(s), s') \cdot \hat{V}^{i-1}_\pi(s')$
  if $\max_{s \in S} |\hat{V}^i_\pi(s) - \hat{V}^{i-1}_\pi(s)| < \epsilon$:
    return $\hat{V}^i_\pi$

Note: can be generalized to executable policies

Policy Evaluation: DR-MDPs

What about policy evaluation for DR-MDPs?

- DR-MDPs (with finite state set) are always cyclic
  $\Rightarrow$ acyclic policy evaluation not applicable
- But: existence of goal state not required for iterative policy evaluation
- albeit traces are infinite, iterative policy evaluation converges due to discount factor in DR-MDPs

$\Rightarrow$ use iterative policy evaluation

Policy Evaluation: FH-MDPs

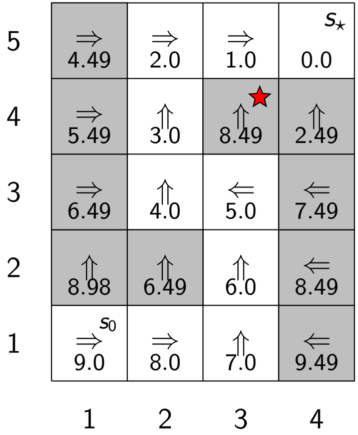
What about policy evaluation for FH-MDPs?

- The relevant state space for FH-MDPs consists of pairs of states and steps-to-go
- as each transition includes a decrease of the steps-to-go, the state space is always acyclic

$\Rightarrow$ use acyclic policy evaluation

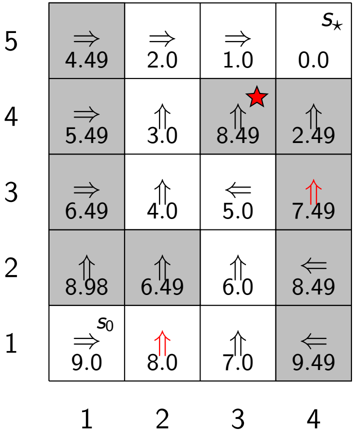
Policy Iteration

Example: Greedy Action



- Can we learn more from this than the state-values of a policy?

Example: Greedy Action



- Can we learn more from this than the state-values of a policy?
- Yes! By evaluating all state-action pairs
  we can derive a better policy

Greedy actions and policies

Definition (Greedy Action)

Let s be a state of an SSP or DR-MDP $\mathcal{T}$ and V be a state-value function for $\mathcal{T}$. The greedy action in s with respect to V is

$$a_V(s) := \arg \min_{\ell \in L(s)} c(\ell) + \sum_{s' \in \mathrm{succ}(s,\ell)} T(s, \ell, s') \cdot V(s').$$

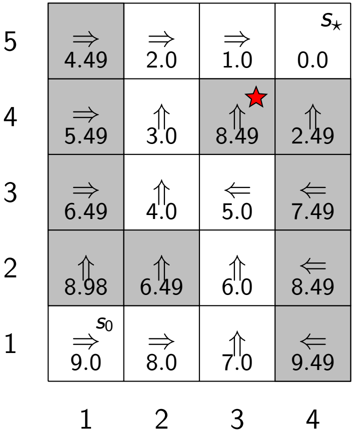
The greedy policy is the policy π_V with $\pi_V(s) = a_V(s)$.

Note: V is often derived as $V_{\pi'}$ from a policy π', but we allow for arbitrary state-value functions that map each state to a real value.

Policy Iteration

- Policy Iteration (PI) was first proposed by Howard in 1960
- exploits observation that greedy actions in result of policy evaluation describe better policy
- starts with arbitrary policy π_0
- alternates policy evaluation and policy improvement
- until convergence to an optimal policy (when policy doesn't change between two steps)

Example: Policy Iteration



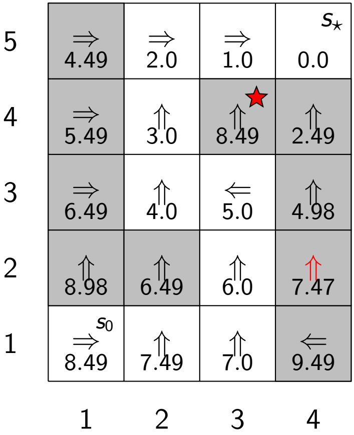
π_0

Example: Policy Iteration

5	$\Rightarrow$ 4.49	$\Rightarrow$ 2.0	$\Rightarrow$ 1.0	$s_\star$ 0.0
4	$\Rightarrow$ 5.49	$\Uparrow$ 3.0	$\Uparrow$ 8.49	$\Uparrow$ 2.49
3	$\Rightarrow$ 6.49	$\Uparrow$ 4.0	$\Leftarrow$ 5.0	$\Uparrow$ 4.98
2	$\Uparrow$ 8.98	$\Uparrow$ 6.49	$\Uparrow$ 6.0	$\Leftarrow$ 8.49
1	s_0 $\Rightarrow$ 8.49	$\Uparrow$ 7.49	$\Uparrow$ 7.0	$\Leftarrow$ 9.49
	1	2	3	4

π_1

Example: Policy Iteration



$\pi_2 = \pi_3$

Policy Iteration

Policy Iteration for SSP, FH-MDP or DR-MDP $\mathcal{T}$

initialize π_0 to any policy (for SSP: proper)
for $i = 1, 2, \ldots$:
    compute V_{π_i}
    let π_{i+1} be the greedy policy w.r.t V_{π_i}
    if $\pi_i = \pi_{i+1}$:
        return π_i

Summary

Summary

- Policy evaluation for acyclic policy is possible in one sweep over the state space.
- Iterative policy evaluation converges over multiple sweeps to true state-values.
- Greedy actions in evaluated policy allow to improve policy.
- Policy iteration alternates policy evaluation and policy improvement.
- Policy iteration results in optimal policy.