Policy Evaluation → Value Iteration (Bellman Equations for MDP_1)

- $<\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{C}, \mathcal{G}, s_0>$
- Define V*(s) {optimal cost} as the minimum expected cost to reach a goal from this state.
- V* should satisfy the following equation:

$$V^*(s) = 0 \quad \text{if } s \in \mathcal{G}$$

$$= \min_a Q^*(s,a)$$

Q*(s,a)

V*(s) = $\min_a$ Q*(s,a)

Bellman Equations for MDP_2

- $<\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, s_0, \gamma>$
- Define **V*(s)** {optimal **value**} as the **maximum** expected **discounted reward** from this state.
- V* should satisfy the following equation:

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s') \left[\mathcal{R}(s, a, s') + \gamma V^*(s') \right]$$

Fixed Point Computation in VI

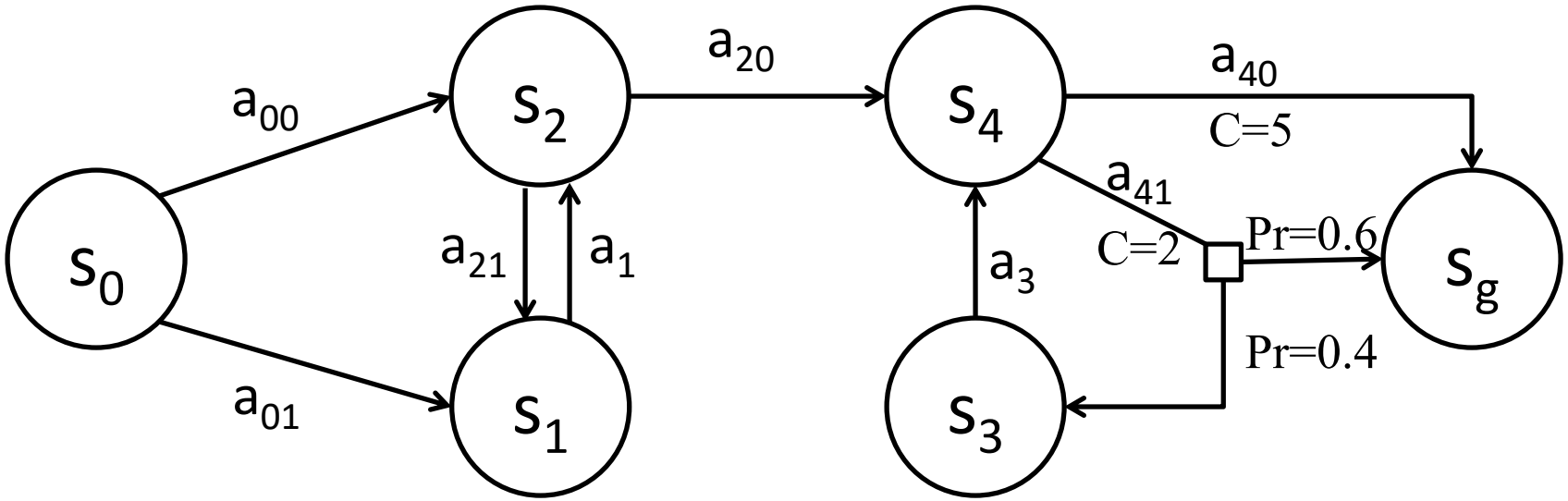
$$V^*(s) = \min_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s') \left[\mathcal{C}(s, a, s') + V^*(s') \right]$$

iterative refinement

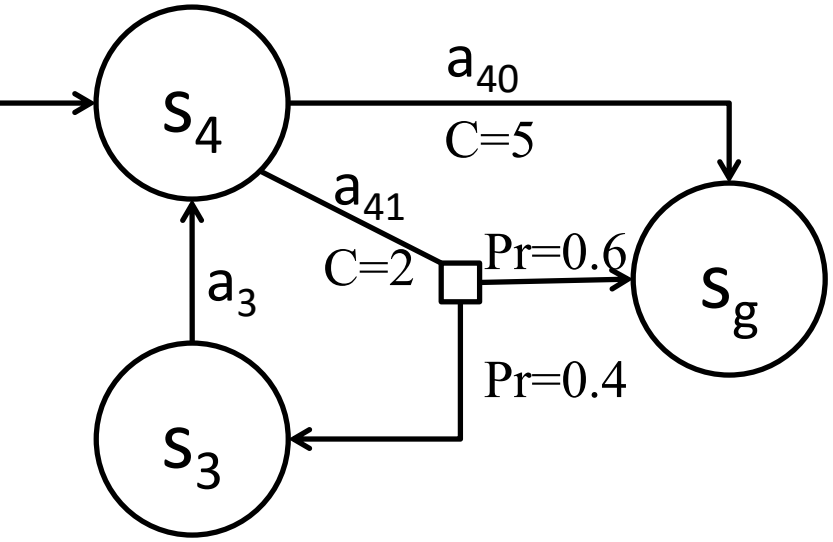
$$V_n(s) \leftarrow \min_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s') \left[\mathcal{C}(s, a, s') + V_{n-1}(s') \right]$$

non-linear

Example



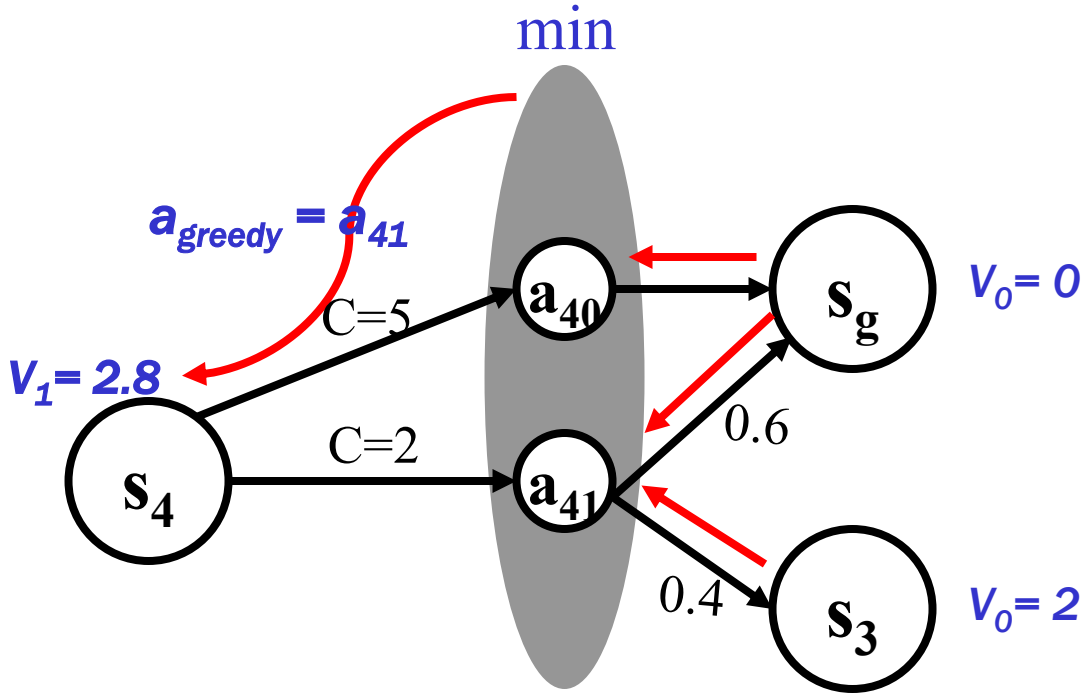
Bellman Backup



$Q_1(s_4,a_{40}) = 5 + 0$

$Q_1(s_4,a_{41}) = 2 + 0.6 \times 0 + 0.4 \times 2$

$= 2.8$



Value Iteration [Bellman 57]

No restriction on initial value function

iteration n

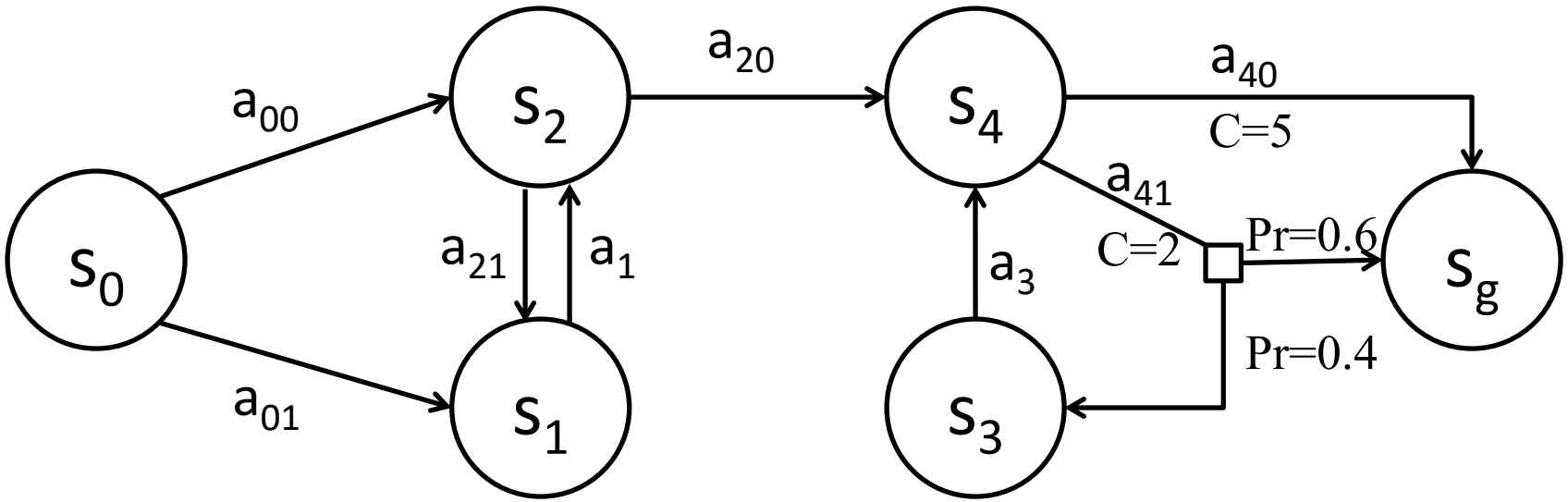
1 initialize V_0 arbitrarily for each state
2 $n \leftarrow 0$
3 **repeat**
4 $\quad n \leftarrow n+1$
5 $\quad$ **foreach** $s \in \mathcal{S}$ **do**
6 $\quad\quad$ compute $V_n(s)$ using Bellman backup at s
7 $\quad\quad$ compute $\text{residual}_n(s) = |V_n(s) - V_{n-1}(s)|$
8 $\quad$ **end**
9 **until** $\max_{s \in \mathcal{S}} \text{residual}_n(s) < \epsilon$;
10 return greedy policy: $\pi^{V_n}(s) = \text{argmin}_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{T}(s,a,s') \left[\mathcal{C}(s,a,s') + V_n(s')\right]$

ϵ-consistency

termination condition

Example

(all actions cost 1 unless otherwise stated)



n	$V_n(s_0)$	$V_n(s_1)$	$V_n(s_2)$	$V_n(s_3)$	$V_n(s_4)$
0	3	3	2	2	1
1	3	3	2	2	2.8
2	3	3	3.8	3.8	2.8
3	4	4.8	3.8	3.8	3.52
4	4.8	4.8	4.52	4.52	3.52
5	5.52	5.52	4.52	4.52	3.808
20	5.99921	5.99921	4.99969	4.99969	3.99969

Comments

- Decision-theoretic Algorithm
- Dynamic Programming
- Fixed Point Computation
- Probabilistic version of Bellman-Ford Algorithm
  - for shortest path computation
  - MDP_1 : Stochastic Shortest Path Problem

- Time Complexity
  - one iteration: $O(|\mathcal{S}|^2|\mathcal{A}|)$
  - number of iterations: $poly(|\mathcal{S}|, |\mathcal{A}|, 1/\epsilon, 1/(1-\gamma))$
- Space Complexity: $O(|\mathcal{S}|)$

Changing the Search Space

- Value Iteration
  - Search in value space
  - Compute the resulting policy
- Policy Iteration
  - Search in policy space
  - Compute the resulting value

Policy iteration [Howard'60]

- assign an arbitrary assignment of π_0 to each state.
- repeat
  - Policy Evaluation: compute V_{n+1}: the evaluation of π_n
  - Policy Improvement: for all states s
    - compute $\pi_{n+1}(s)$: $\text{argmin}_{a \in Ap(s)} Q_{n+1}(s,a)$
- until $\pi_{n+1} = \pi_n$

costly: $O(n^3)$

Modified Policy Iteration → **approximate by value iteration using fixed policy**

Advantage

- searching in a finite (policy) space as opposed to uncountably infinite (value) space ⇒ convergence in fewer number of iterations.
- all other properties follow!

Modified Policy iteration

- assign an arbitrary assignment of π_0 to each state.

- repeat
  - Policy Evaluation: compute V_{n+1} the *approx.* evaluation of π_n
  - Policy Improvement: for all states s
    - compute $\pi_{n+1}(s)$: $\text{argmin}_{a \in Ap(s)} Q_{n+1}(s,a)$
- until $\pi_{n+1} = \pi_n$

Advantage

- probably the most competitive synchronous dynamic programming algorithm.

VI → Asynchronous VI

- Is backing up *all* states in an iteration essential?
  - No!
- States may be backed up
  - as many times
  - in any order
- If no state gets starved
  - convergence properties still hold!!

Applications

- Stochastic Games
- Robotics: navigation, helicopter manuevers…
- Finance: options, investments
- Communication Networks
- Medicine: Radiation planning for cancer
- Controlling workflows
- Optimize bidding decisions in auctions
- Traffic flow optimization
- Aircraft queueing for landing; airline meal provisioning
- Optimizing software on mobiles
- Forest firefighting
- …

Extensions

- Heuristic Search + Dynamic Programming
  - AO*, LAO*, RTDP, …
- Hierarchical MDPs
  - hierarchy of sub-tasks, actions to scale better
- Reinforcement Learning
  - learning the probability and rewards
  - acting while learning – connections to psychology
- Partially Observable Markov Decision Processes
  - noisy sensors; partially observable environment
  - popular in robotics