Lecture 2: Making Good Sequences of Decisions Given a Model of World

CS234: RL

Emma Brunskill

Winter 2018

Human in the loop exoskeleton work from Steve Collins' lab

Class Structure

- Last Time:
  - Introduction
  - The components of an agent: model, value, policy
- This time:
  - Making good decisions given a Markov decision process
- Next time:
  - Policy evaluation when don’t have a model of how the world works

1 minute Quick Check

- Turn to the person next to you: what is a model, value and policy?

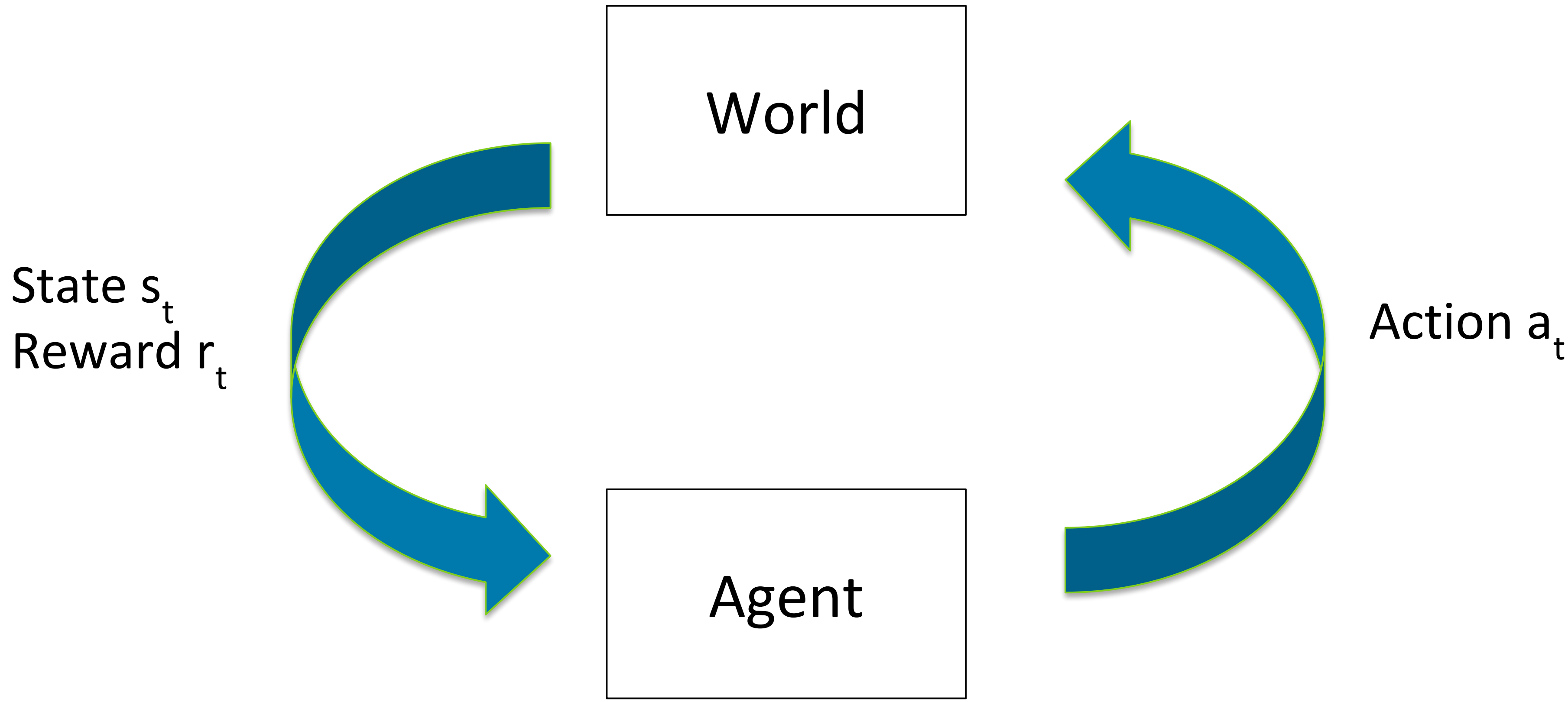
Models, Policies, Values

- **Model:** Mathematical models of dynamics and reward
- **Policy:** function mapping agent's states to action
- **Value function:** future rewards from being in a state and/or action when following a particular policy

Today: Given a Model of the World

1. Markov Processes
2. Markov Reward Processes (MRPs)
3. Markov Decision Processes (MDPs)
4. Evaluation and Control in MDPs

Full Observability: Markov Decision Process (MDP)



- MDPs can model a huge number of interesting problems and settings
  - Bandits: single state MDP
  - Optimal control mostly about continuous-state MDPs
  - Partially observable MDPs = MDP where state is history

Recall: Markov Property

- Information state: sufficient statistic of history
- **Definition:**
  - State s_t is Markov if and only if (iff):
    - $p(s_{t+1} | s_t, a_t) = p(s_{t+1} | h_t, a_t)$
- Future is independent of past given present

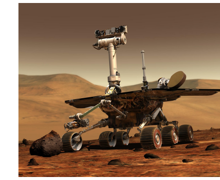
Markov Process or Markov Chain

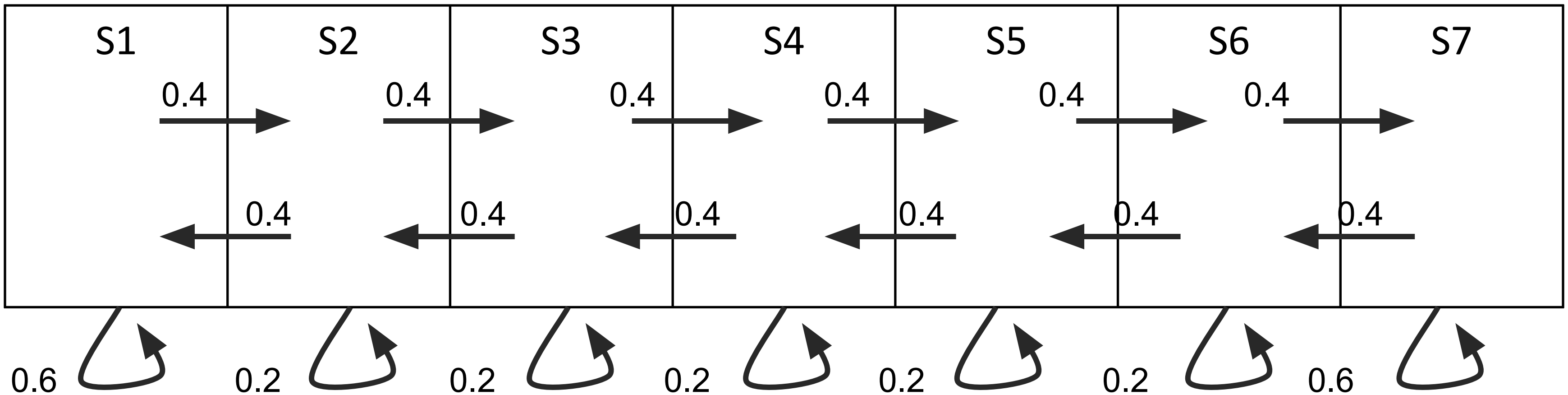
- Memoryless random process:
  - Sequence of random states with Markov property
- **Definition of MP**:
  - *S* is a (finite) set of states
  - *P* is dynamics / transition model, that specifies $P(s_{t+1} = s' | s_t = s)$
- Note: no rewards, no actions
- If finite number (N) of states, can express *P* as a matrix

$$P = \quad \textit{from} \quad \begin{pmatrix} P(s1|s1) & P(s2|s1) & \dots & P(sN|s1) \\ P(s1|s2) & P(s2|s2) & \dots & \\ \dots. & & & \\ P(s1|sN) & P(s2|sN) & \dots & P(sN|sN) \end{pmatrix}$$

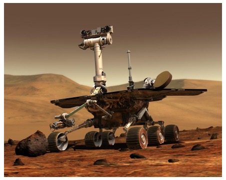
to

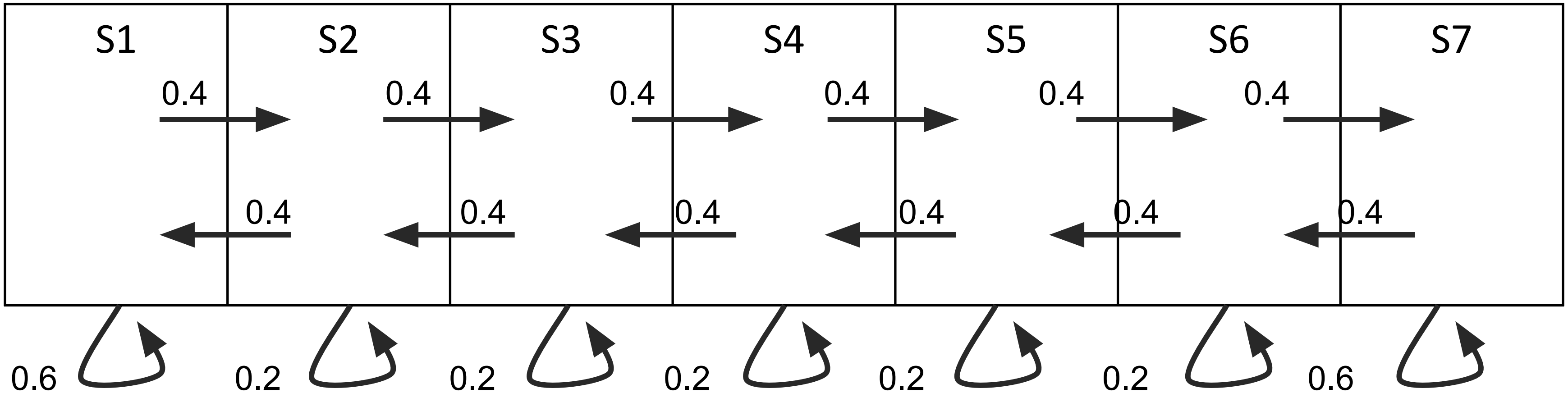
Ex. Mars Rover Markov Chain





Ex. Mars Rover MC Episodes

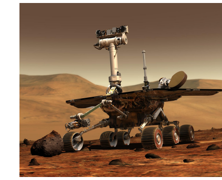


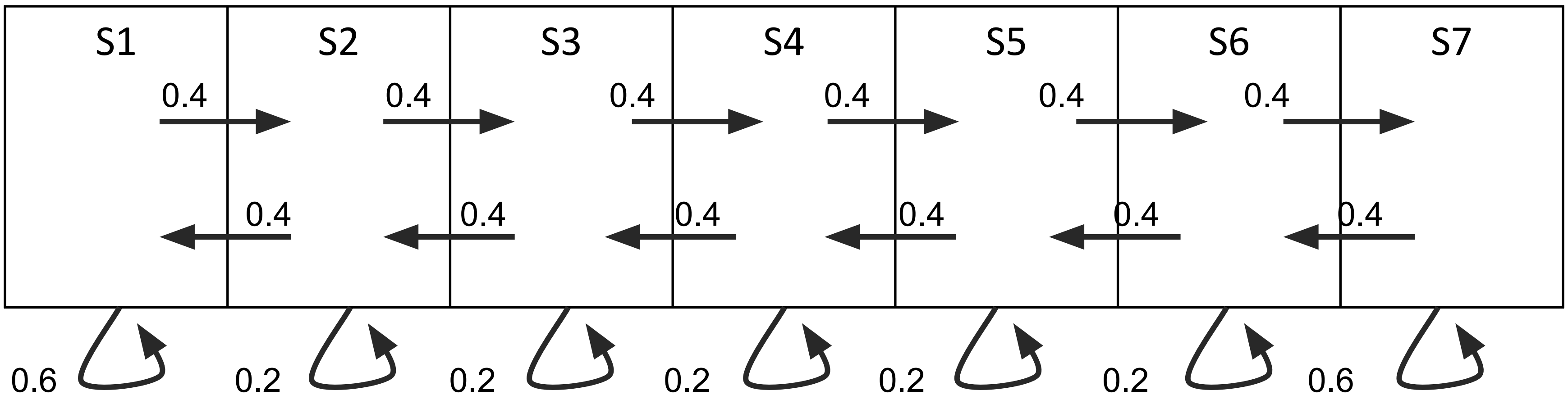


Example: sample episodes starting from S4

- S4, S5, S6, S7, S7, S7…
- S4, S4, S5, S4, S5, S6, ...
- S4, S3, S2, S1, ...

Ex. Mars Rover Transition P





$$
P = \textit{from} \begin{pmatrix}
0.6 & 0.4 & 0 & 0 & 0 & 0 & 0 \\
0.4 & 0.2 & 0.4 & 0 & 0 & 0 & 0 \\
0 & 0.4 & 0.2 & 0.4 & 0 & 0 & 0 \\
0 & 0 & 0.4 & 0.2 & 0.4 & 0 & 0 \\
0 & 0 & 0 & 0.4 & 0.2 & 0.4 & 0 \\
0 & 0 & 0 & 0 & 0.4 & 0.2 & 0.4 \\
0 & 0 & 0 & 0 & 0 & 0.4 & 0.6
\end{pmatrix}
$$

to

Markov Reward Process (MRP)

- A Markov Reward Process is a Markov Chain + rewards
- **Definition of MRP**:
  - *S* is a (finite) set of states
  - *P* is dynamics / transition model, that specifies $P(s_{t+1} = s' | s_t = s)$
  - *R* is a reward function $R(s_t = s) = \mathbb{E}[r_t \mid s_t = s]$
  - Discount factor $\gamma \in [0,1]$
- Note: no actions
- If finite number (N) of states, can express *R* as a vector

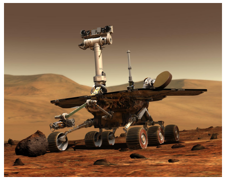
Return & Value Function

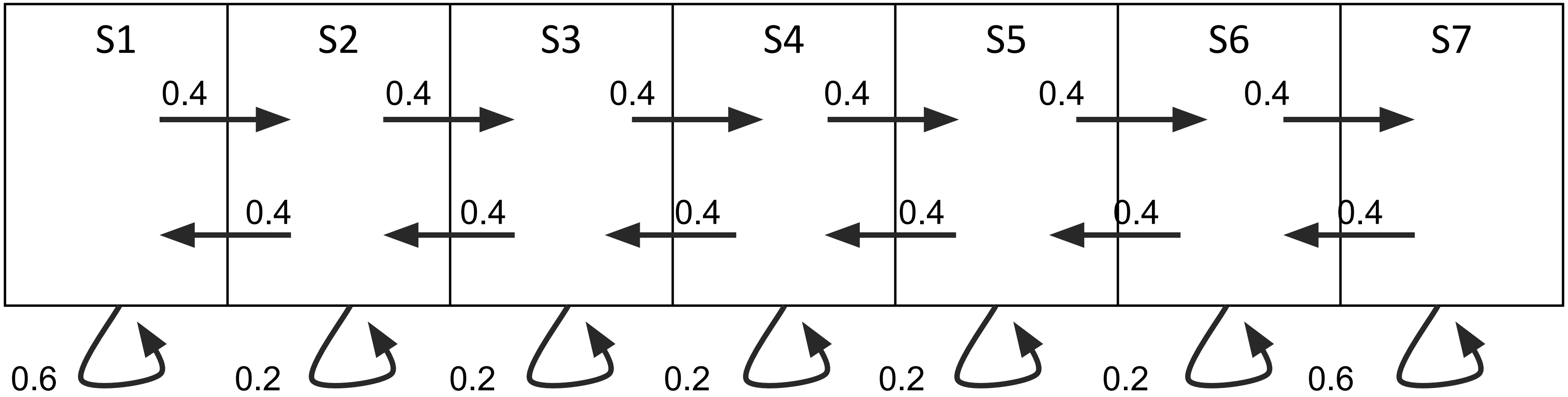
- **Definition of Horizon:**
  - Number of time steps in each episode in a process
  - Can be infinite
  - Otherwise called **finite** Markov reward process
- **Definition of Return G_t** (for a Markov reward process):
  - Discounted sum of rewards from time step t to horizon
  - $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots$
- **Definition of State value function V(s)** (for a MRP):
  - Expected return from starting in state s
  - $V(s) = \mathbb{E}\,[G_t | s_t = s] = \mathbb{E}\,[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots | s_t = s]$

Discount Factor

- Mathematically convenient (avoid infinite returns and values)
- Humans often act as if there's a discount factor < 1
- γ=0: Only care about immediate reward
- γ=1: Future reward is as beneficial as immediate reward
- If episode lengths are always finite, can use γ=1

Ex. Mars Rover MRP

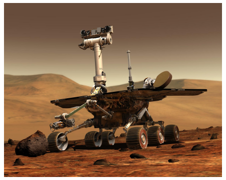


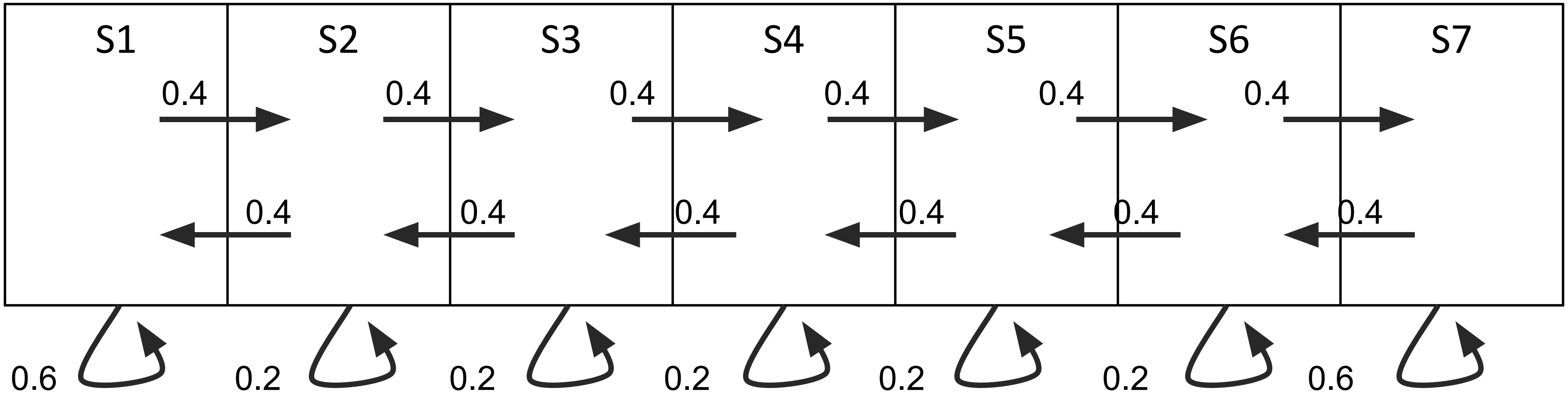


Reward: +1 in S1, +10 in S7, 0 in all other states

Sample returns for sample 4-step episodes, γ=½

Ex. Mars Rover MRP



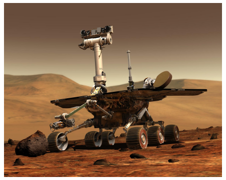


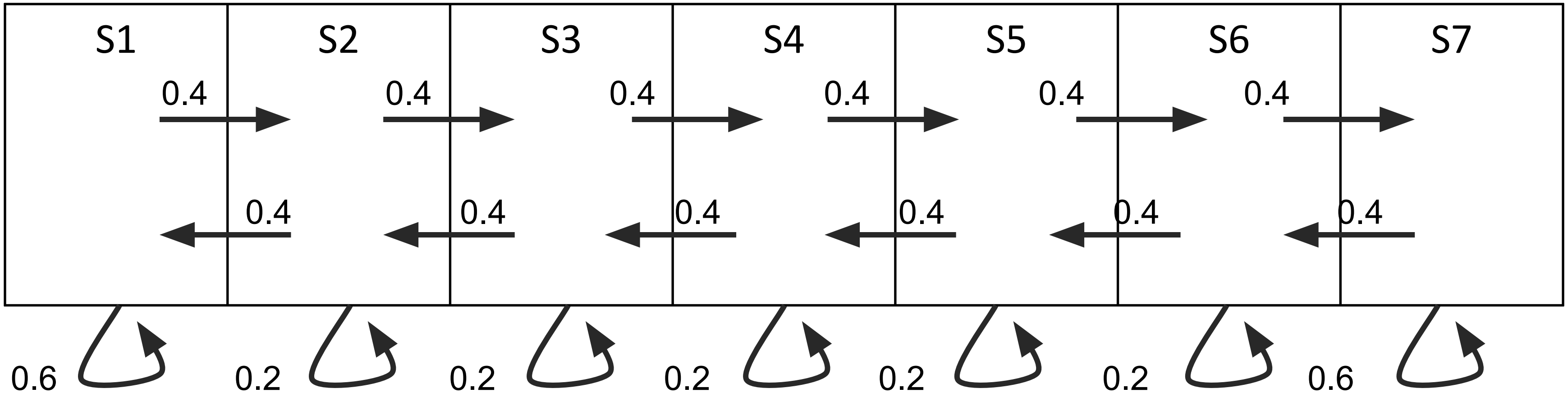
Reward: +1 in S1, +10 in S7, 0 in all other states

Sample returns for sample 4-step episodes, γ=½

- S4, S5, S6, S7: 0 + ½ *0 + ¼ * 0 + ⅛ * 10 = 1.25

Ex. Mars Rover MRP



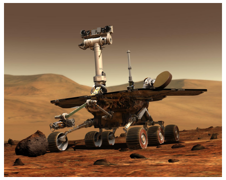


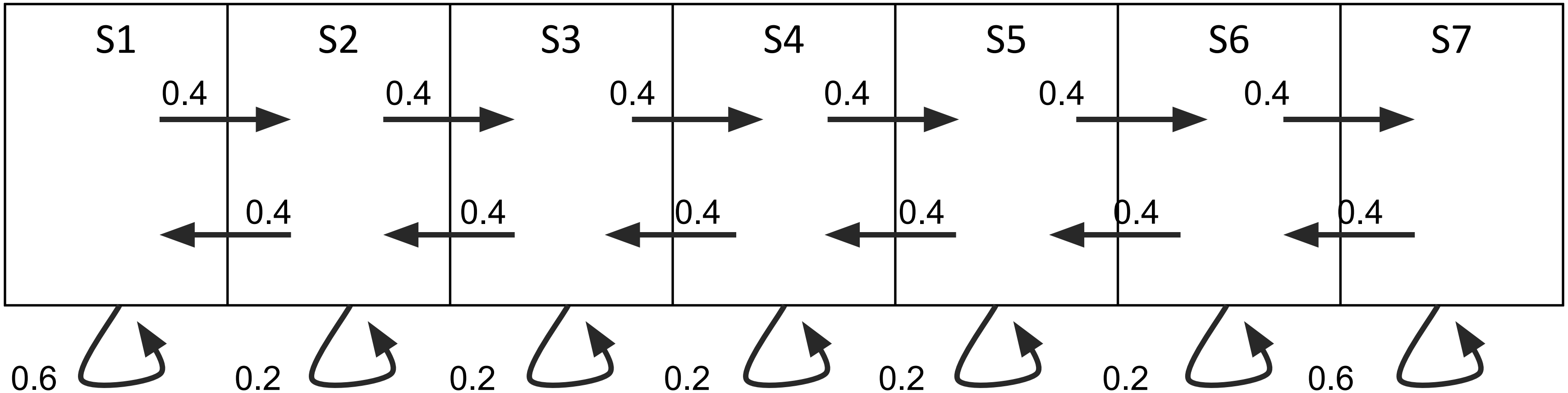
Reward: +1 in S1, +10 in S7, 0 in all other states

Sample returns for sample 4-step episodes, γ=½

- S4, S5, S6, S7: 0 + ½ *0 + ¼ * 0 + ⅛ * 10 = 1.25
- S4, S4, S5, S4: 0 + ½ *0 + ¼ * 0 + ⅛ * 0 = 0
- S4, S3, S2, S1: 0 + ½ *0 + ¼ * 0 + ⅛ * 1 = 0.125

Ex. Mars Rover MRP V(S4)





Reward: +1 in S1, +10 in S7, 0 in all other states

Value for infinite-step **horizon process**, γ=½

- V(S1) = 1.53
- V(S2) = 0.37
- V(S3) = 0.13
- V(S4) = 0.22
- V(S5) = 0.85
- V(S6) = 3.59
- V(S7) = 15.31

Computing the Value of a Markov Reward Process

- Could estimate by simulation
  - Generate a large number of episodes
  - Average returns
  - Concentration inequalities bound how quickly average concentrates to expected value

Computing the Value of a Markov Reward Process

- Could estimate by simulation
- Markov property yields additional structure
- MRP value function satisfies:

$$V(s) = \underbrace{R(s)}_{\text{Immediate reward}} + \underbrace{\gamma \sum_{s' \in S} P(s'|s)V(s')}_{\text{Discounted sum of future rewards}}$$

Matrix Form of Bellman Eqn for Markov Reward Processes

- For finite state MRP can express using matrices

$$\begin{bmatrix} V(s1) \\ \dots \\ V(sN) \end{bmatrix} = \begin{bmatrix} R(s1) \\ \dots \\ R(sN) \end{bmatrix} + \gamma \begin{bmatrix} P(s1|s1) & \dots & P(sN|s1) \\ & \dots & \\ P(s1|sN) & \dots & P(sN|sN) \end{bmatrix} \begin{bmatrix} V(s1) \\ \dots \\ V(sN) \end{bmatrix}$$

$$V = R + \gamma PV$$

Analytic Solution for Value of MRP

- For finite state MRP can express using matrices

$$\begin{bmatrix} V(s1) \\ \dots \\ V(sN) \end{bmatrix} = \begin{bmatrix} R(s1) \\ \dots \\ R(sN) \end{bmatrix} + \gamma \begin{bmatrix} P(s1|s1) & \dots & P(sN|s1) \\ & \dots & \\ P(s1|sN) & \dots & P(sN|sN) \end{bmatrix} \begin{bmatrix} V(s1) \\ \dots \\ V(sN) \end{bmatrix}$$

$$V = R + \gamma P V$$

Matrix inverse, ~O(N^3)

$$\begin{aligned} V - \gamma P V &= R \\ (I - \gamma P) V &= R \\ V &= (I - \gamma P)^{-1} R \end{aligned}$$

Iterative Algorithm for Computing Value of a MRP

- Dynamic programming
- Initialize $V_0(s) = 0$ for all s
- For k=1 until convergence
  - For all s in S:

$$V_k(s) = R(s) + \gamma \sum_{s' \in S} P(s'|s) V_{k-1}(s')$$

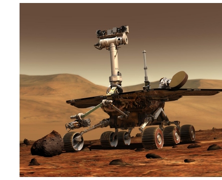
- Computational complexity: $O(S^2)$ for each t

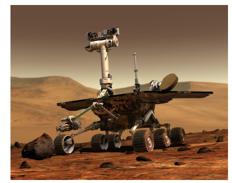
Markov Decision Process (MDP)

- A Markov Decision Process is Markov Reward Process + actions
- **Definition of MDP**:
  - *S* is a (finite) set of Markov states
  - *A* is a (finite) set of actions
  - *P* is dynamics / transition model for **each action**, that specifies $P(s_{t+1} = s' | s_t = s, a_t = a)$
  - R is a reward function $R(s_t = s, a_t = a) = \mathbb{E}[r_t \mid s_t = s, a_t = a]$*
  - Discount factor $\gamma \in [0,1]$
- MDP is a tuple: (S, A, P, R, γ)

*Reward sometimes defined as a function of the current state, or as a function of the state-action-next state. Most frequently in this class we will assume reward is a function of state and action

Ex. Mars Rover MDP



S1	S2	S3	S4	S5	S6	S7
Okay Field Site R=+1	R=0	R=0	 R=0	R=0	R=0	Fantastic Field Site R=+10

$$P(s'|s,TL)= \begin{matrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{matrix} \qquad P(s'|s,TR)= \begin{matrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

- 2 actions: TryLeft or TryRight
  - Deterministic: Succeeds unless hit edge, then stay

MDP Policies

- Policy specifies what action to take in each state
  - Can be deterministic or stochastic
- For generality consider as a conditional distribution: given a state specifies a distribution over actions
- Policy $\pi(a|s) = P(a_t=a|s_t=s)$

MDP + Policy

- MDP + $\pi(a|s)$ = a Markov reward process
- Precisely it is a a MRP $(S, R^{\pi}, P^{\pi}, \gamma)$ where

$$
\begin{aligned}
R^{\pi}(s) &= \sum_{a \in A} \pi(a|s) R(s,a) \\
P^{\pi}(s'|s) &= \sum_{a \in A} \pi(a|s) P(s'|s,a)
\end{aligned}
$$

Policy Evaluation for MDP

- MDP + $\pi(a|s)$ = a Markov reward process
- Precisely it is a a MRP $(S, R^{\pi}, P^{\pi}, \gamma)$ where

$$\begin{aligned} R^{\pi}(s) &= \sum_{a \in A} \pi(a|s) R(s,a) \\ P^{\pi}(s'|s) &= \sum_{a \in A} \pi(a|s) P(s'|s,a) \end{aligned}$$

- Implies we can use same techniques to evaluate the value of a policy for a MDP as we could to compute the value of a MRP

Slight Modification to Iterative Algorithm for Computing Value of a MRP

- Initialize $V_0(s) = 0$ for all s
- For k=1 until convergence
  - For all s in S:

$$V_k^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V_{k-1}^\pi(s')$$

- Just replaced dynamics and reward model

Slight Modification to Iterative Algorithm for Computing Value of a MRP

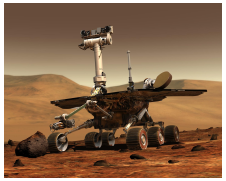
- Initialize $V_0(s) = 0$ for all s
- For k=1 until convergence
  - For all s in S:

Bellman backup for a particular policy

$$V_k^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V_{k-1}^\pi(s')$$

- Just replaced dynamics and reward model

Policy Evaluation: Example

S1	S2	S3	S4	S5	S6	S7
Okay Field Site +1						Fantastic Field Site +10

- Deterministic actions of TryLeft or TryRight
- Reward: +1 in state S1, +10 in state S7, 0 otherwise
- Let $\pi_0(s)$=TryLeft for all states (e.g. always go left)
- Set discount factor to 0. What is the value of this policy? = *R*

$$V_k^\pi(s) = R^\pi(s) + \gamma \sum_{s' \in S} P^\pi(s'|s) V_{k-1}^\pi(s')$$

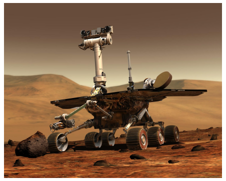
MDP Control

- Compute the optimal policy

$$\pi^*(s) = \arg\max_{\pi} V^{\pi}(s)$$

- There exists a unique optimal value function
- Optimal policy for a MDP in an infinite horizon problem (agent acts forever) is:
  - Deterministic

Short Exercise: How Many Deterministic Policies?

S1	S2	S3	S4	S5	S6	S7
Okay Field Site +1						Fantastic Field Site +10

- 7 discrete states (location of rover)
- 2 actions: TryLeft or TryRight

Is the optimal policy unique?

MDP Control

- Compute the optimal policy

$$\pi^*(s) = \arg\max_{\pi} V^{\pi}(s)$$

- There exists a unique optimal value function
- Optimal policy for a MDP in an infinite horizon problem (agent acts forever) is:
  - Deterministic
  - Stationary (does not depend on time step)
  - Unique? Not necessarily, may be ties

Policy Search

- One option is searching to compute best policy
- Number of deterministic policies is $|A|^{|S|}$
- Policy iteration is generally more efficient than enumeration

New Definition: State-Action Value Q

- State-action value of a policy

$$Q^{\pi}(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^{\pi}(s')$$

- Take action a, then follow policy

Policy Iteration (PI)

1. i=0; Initialize $\pi_0(s)$ randomly for all states s
2. While i == 0 or $|\pi_i - \pi_{i-1}| > 0$
   - Policy evaluation of π_i
   - i=i+1
   - Policy improvement

Use a L1 norm: measures if the policy changed for any state

Policy Improvement

Compute state-action value of a policy π_i

$$Q^{\pi_i}(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^{\pi_i}(s')$$

Note

$$\begin{aligned}
\max_a Q^{\pi_i}(s,a) &= \max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^{\pi_i}(s') \\
&\geq R(s,\pi_i(s)) + \gamma \sum_{s' \in S} P(s'|s,\pi_i(s)) V^{\pi_i}(s') \\
&= V^{\pi_i}(s)
\end{aligned}$$

Define new policy

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a) \;\forall s \in S$$

Policy Iteration (PI)

1. i=0; Initialize $\pi_0(s)$ randomly for all states s
2. While i == 0 or $|\pi_i - \pi_{i-1}| > 0$
   - Policy evaluation: Compute value of π_i
   - i=i+1
   - Policy improvement:

$$Q^{\pi_i}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi_i}(s')$$

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$$

Use a L1 norm: measures if the policy changed for any state

Delving Deeper Into Improvement

$$Q^{\pi_i}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi_i}(s')$$

$$\max_a Q^{\pi_i}(s,a) \geq V^{\pi_i}(s)$$

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$$

- So if take $\pi_{i+1}(s)$ then followed π_i forever,
  - expected sum of rewards would be at least as good as if we had always followed π_i
- But new proposed policy is to always follow π_{i+1} …

Monotonic Improvement in Policy

- Definition

$$V^{\pi_1} \geq V^{\pi_2} \quad \rightarrow \quad V^{\pi_1}(s) \geq V^{\pi_2}(s) \;\; \forall s \in S$$

- Proposition: $V^{\pi'} >= V^{\pi}$ with strict inequality if π is suboptimal (where π' is the new policy we get from doing policy improvement)

Proof

$$
\begin{aligned}
V^{\pi_i}(s) \quad & \leq \quad \max_a Q^{\pi_i}(s,a) \\
& = \quad \max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^{\pi_i}(s') \\
& = \quad R(s, \pi_{i+1}(s)) + \gamma \sum_{s' \in S} P(s'|s, \pi_{i+1}(s)) V^{\pi_i}(s') \\
& \leq \quad R(s, \pi_{i+1}(s)) + \gamma \sum_{s' \in S} P(s'|s, \pi_{i+1}(s)) \left(\max_{a'} Q^{\pi_i}(s', a') \right) \\
& = \quad R(s, \pi_{i+1}(s)) + \gamma \sum_{s' \in S} P(s'|s, \pi_{i+1}(s)) \left(R(s', \pi_{i+1}(s') + \gamma \sum_{s'' \in S} P(s''|s', \pi_{i+1}(s')) V^{\pi_i}(s'') \right) \\
& \ldots \\
& = V^{\pi_{i+1}}(s)
\end{aligned}
$$

// uses definition of π_{i+1}

If Policy Doesn't Change ($\pi_{i+1}(s) = \pi_i(s)$ for all s) Can It Ever Change Again in More Iterations?

- Recall policy improvement step

$$Q^{\pi_i}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi_i}(s')$$

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$$

- Assuming can do Q computation and policy update exactly, no change to Q and policy

Policy Iteration (PI)

1. i=0; Initialize $\pi_0(s)$ randomly for all states s
2. While i == 0 or $|\pi_i - \pi_{i-1}| > 0$
   - Policy evaluation: Compute value of π_i
   - i=i+1
   - Policy improvement:

$$Q^{\pi_i}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi_i}(s')$$

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$$

Use a L1 norm: measures if the policy changed for any state

Policy Iteration Can Take At Most |A|^|S| Iterations* (Size of # Policies)

1. i=0; Initialize $\pi_0(s)$ randomly for all states s
2. Converged = 0;
3. While i == 0 or $|\pi_i - \pi_{i-1}| > 0$
   - i=i+1
   - Policy evaluation: Compute V^π
   - Policy improvement:

$$Q^{\pi_i}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi_i}(s')$$

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$$

* For finite state and action spaces

MDP: Computing Optimal Policy and Optimal Value

- Policy iteration computes optimal value and policy
- Value iteration is another technique
  - Idea: Maintain optimal value of starting in a state s if have a finite number of steps k left in the episode
  - Iterate to consider longer and longer episodes

Bellman Equation and Bellman Backup Operators

- Bellman equation
  - The value function for a policy must satisfy

$$V^{\pi}(s) = R^{\pi}(s) + \gamma \sum_{s' \in S} P^{\pi}(s'|s) V^{\pi}(s')$$

- Bellman backup operator
  - Applied to a value function
  - Returns a new value function
  - Improves the value if possible

$$BV(s) = \max_{a} R(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s')$$

  - *BV* yields a value function over all s

Value Iteration (VI)

1. Initialize $V_0(s)=0$ for all states s
2. Set k=1
3. Loop until [finite horizon, convergence]
   - For each state s

$$V_{k+1}(s) = \max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s')$$

   - View as Bellman backup on value function

$$V_{k+1} = BV_k$$

$$\pi_{k+1}(s) = \arg\max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s')$$

Looking at Policy Iteration As Bellman Operations: Policy Evaluation: Compute Fixed Point of B^{π}

- Bellman backup operator for a particular policy

$$B^{\pi}V(s) = R^{\pi}(s) + \gamma \sum_{s' \in S} P^{\pi}(s'|s)V(s)$$

- To do policy evaluation, repeatedly apply operator until V stops changing

$$V^{\pi} = B^{\pi}B^{\pi} \dots B^{\pi}V$$

Looking at Policy Iteration As Bellman Operations: Policy Improvement, Slight Variant of Bellman

- Bellman backup operator for a particular policy

$$B^{\pi}V(s) = R^{\pi}(s) + \gamma \sum_{s' \in S} P^{\pi}(s'|s)V(s)$$

- To do policy improvement

$$\pi_{k+1}(s) = \arg\max_{a} R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a)V^{\pi_k}(s')$$

Going Back to Value Iteration (VI)

1. Initialize $V_0(s)=0$ for all states s
2. Set k=1
3. Loop until [finite horizon, convergence]
   - For each state s

$$V_{k+1}(s) \quad = \quad \max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s')$$

   - Doing a Bellman backup on value function

$$\begin{aligned} V_{k+1} &= BV_k \\ \pi_{k+1}(s) &= \arg\max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s') \end{aligned}$$

Contraction Operator

- Let O be an operator
- If $|OV - OV'| \leq |V - V|$
- Then O is a contraction operator

Will Value Iteration Converge?

- Yes, if discount factor $\gamma < 1$ or end up in a terminal state with probability 1
- Bellman backup is a contraction if discount factor, $\gamma < 1$
- If apply it to two different value functions, distance between value functions shrinks after apply Bellman equation to each

Bellman Backup is a Contraction on V (γ<1)

|| V-V’|| = Infinity norm (find max difference over all states, e.g. max(s) |V(s) – V’(s) |

$$
\begin{aligned}
\|BV_k - BV_j\| &= \left\| \left(\max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_k(s') \right) - \left(\max_{a'} R(s,a') + \gamma \sum_{s'} P(s'|s,a') V_j(s') \right) \right\| \\
&\leq \left\| \max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a) V_k(s') - R(s,a) - \gamma \sum_{s'} P(s'|s,a) V_j(s') \right\| \\
&= \left\| \max_a \gamma \sum_{s'} P(s'|s,a)(V_k(s') - V_j(s')) \right\| \\
&\leq \left\| \max_a \gamma \sum_{s'} P(s'|s,a) \|V_k - V_j\| \right\| \\
&\leq \left\| \gamma \|V_k - V_j\| \max_a \sum_{s'} P(s'|s,a) \right\| \\
&= \gamma \|V_k - V_j\|
\end{aligned}
$$

Note: even if all inequalities are equalities, this still is a contraction as long as the discount factor is < 1

Check Understanding

- Prove value iteration converges to a unique solution for discrete state and action space and $\gamma<1$
- Does the initialization of values in value iteration impact anything?

Consider Value Iteration for Finite Horizon:
V_k = optimal value if making k more decisions
π_k = optimal policy if making k more decisions

1. Initialize $V_0(s)=0$ for all states s
2. Set k=1
3. Loop until [finite horizon, convergence]
   - For each state s

$$V_{k+1}(s) = \max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s')$$

   - Doing a Bellman backup on value function

$$\begin{aligned} V_{k+1} &= BV_k \\ \pi_{k+1}(s) &= \arg\max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s') \end{aligned}$$

Consider Value Iteration for Finite Horizon: Is optimal policy stationary (independent of time step)? In general, no

1. Initialize $V_0(s)=0$ for all states s
2. Set k=1
3. Loop until [finite horizon, convergence]
   - For each state s

$$V_{k+1}(s) = \max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s')$$

   - Doing a Bellman backup on value function

$$V_{k+1} = BV_k$$

$$\pi_{k+1}(s) = \arg\max_a R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V_k(s')$$

Value vs Policy Iteration

- Value iteration:
  - Compute optimal value if horizon=k
    - Note this can be used to compute optimal policy if horizon = k
  - Increment k
- Policy iteration:
  - Compute infinite horizon value of a policy
  - Use to select another (better) policy
  - Closely related to a very popular method in RL: policy gradient

What You Should Know

- Define MP, MRP, MDP, Bellman operator, contraction, model, Q-value, policy
- Be able to implement
  - Value iteration & policy iteration
- Contrast benefits and weaknesses of policy evaluation approaches
- Be able to prove contraction properties
- Limitations of presented approaches and Markov assumptions
  - Which policy evaluation methods require Markov assumption?