Reinforcement Learning
Lecture 2: Bandits and MDPs

Lecturer: Prof. Dr. Mathias Niepert

Institute for Artificial Intelligence
Machine Learning and Simulation Lab

University of Stuttgart
Germany

imprs-is

April 18, 2024

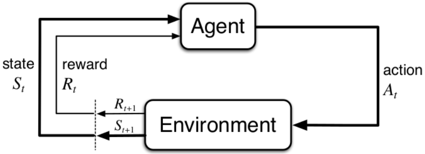
Outline

1. Exploration vs. Exploitation
2. Markov Decision Processes
3. Optimality in MPDs

Exploration vs. Exploitation

What is reinforcement learning?

- Agent observes the **state**
- Agent chooses an **action**
- Agent gets a **reward**
- Aim is to learn a **policy**: what action to choose in a given state in order to get maximum *long-term* reward
- Problems are reduced to three signals being passed back and forth



Many flavours of reinforcement learning

model-based	$S_t, A_t, R_{t+1}, S_{t+1} \ldots \rightarrow p(s' \mid s, a), r(s, a, s') \rightarrow v(s) \rightarrow \pi(s)$
model-free	
value-based	$S_t, A_t, R_{t+1}, S_{t+1} \ldots \rightarrow q(s, a) \rightarrow \pi(s)$
policy-based	$S_t, A_t, R_{t+1}, S_{t+1} \ldots \rightarrow \pi(s)$
actor-critic	$S_t, A_t, R_{t+1}, S_{t+1} \ldots \rightarrow q(s, a), \pi(s)$
imitation learning	$\{(S_{1:T}, A_{1:T}, R_{1:T})^i\}_{i=1}^n \rightarrow \pi(s)$

learning | dynamic programming

k-armed bandit

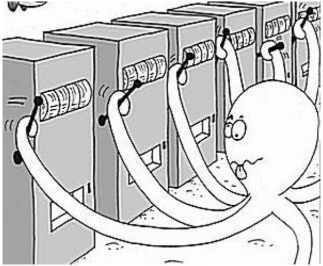


image credits: Microsoft Research

- There are k actions (machines)
- Each machine returns a reward from a (stationary) probability distribution
- Objective is to maximize the expected total reward, aggregated over the first T choices

Value

- Each action a has an expected or mean reward, the **value**:

$$q_*(a) = \mathbb{E}[R_t \mid A_t = a]$$

- If you would know the true action value q_* **for every** a, the next choice would be trivial
- **Estimate** of the action-value at time step t: $Q_t(a)$

Exploration vs. exploitation

- At each time step t there is (at least) one action that maximizes Q_t, called the *greedy* action
- **Greedy policy**:

$$A_t = \arg\max_a Q_t(a)$$

- Exploitation: selecting *greedy* action
- Exploration: selecting *nongreedy* action
  - improving estimate of the nongreedy action's value
  - reward lower in the short run
  - potentially much higher in the long run
- What is better? What does it depend on?
  - current action-value estimates
  - uncertainties
  - number of remaining steps

ϵ-greedy action selection

- Simple idea to force continued exploration
- With probability $1 - \epsilon$ take the *greedy* action
- With probability ϵ take a random action
- All actions are chosen with non-zero probability

Estimating action-values

- *Sample average* method:

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{n(a)}$$

- If $n(a) = 0$, set $Q_t(a) = 0$
- As $n(a) \to \infty, Q_t(a) = q_*(a)$
- We sometimes write $\hat{Q}(a)$ for the estimate
- What is the difference between $q_*(a)$ and $\max_a Q_t(a)$?
  - $q_*(a)$ is the true value of a
  - $\max_a Q_t(a)$ is the greedy action value at time t

ϵ-greedy vs greedy

Which would be better in each of these cases?

1. What if reward variance is very small, e.g. zero?
2. What if reward variance is larger?
3. What if task is non-stationary?

Softmax action selection

- ϵ-greedy: even if worst action is very bad, it will still be chosen with same probability as second-best
- Vary selection probability as a function of the value estimate
- Choose a at time t from among the k actions with probability:

$$\pi_t(a) = \Pr\{A_t = a\} = \frac{\exp(Q_t(a)/\tau)}{\sum_{a'=1}^{k} \exp(Q_t(a')/\tau)}$$

- Also known as the Gibbs or Boltzmann distribution

Softmax action selection

- What if our estimate of the best action $a_* = \max_a q_*(a)$ is initially very small?
- Effect of temperature τ:
  - as $\tau \to \infty$, ... choose action at random
  - as $\tau \to 0$, ... select greedy action

Incremental action-value estimates

- Pick an action
- R_i is now the reward received after ith selection of *this* action

$$Q_n = \frac{R_1 + R_2 + \ldots R_{n-1}}{n-1}$$

$$\begin{aligned} Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i = \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\left(R_n + (n-1)Q_n\right) = \frac{1}{n}\left(R_n + nQ_n - Q_n\right) \end{aligned}$$

$$Q_{n+1} = Q_n + \frac{1}{n}\left[R_n - Q_n\right]$$

$$NewEstimate \leftarrow OldEstimate + StepSize[Target - OldEstimate]$$

Incremental update

- General form is very important and will show up frequently:

$$NewEstimate \leftarrow OldEstimate + StepSize[Target - OldEstimate]$$

- Here, *StepSize* or α depends on n: $\alpha = 1/n$
- Often it is kept constant, e.g. $\alpha = 0.1$
- What is the implication of keeping α constant? Why would it make sense?
  - gives more weights to recent rewards
  - ...think of non-stationary environments

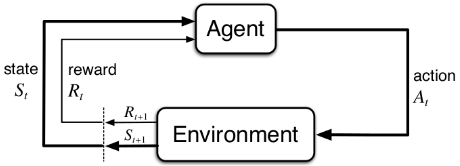
Markov Decision Processes

From bandits to Markov Decision Processes

- Bandits:
  - ✗ states
  - ✓ feedback
  - ✓ decision making
- Markov Chains:
  - ✓ states
  - ✗ feedback
  - ✗ decision making
- Markov Reward Process:
  - ✓ states
  - ✓ feedback
  - ✗ decision making
- Markov Decision Process:
  - ✓ states
  - ✓ feedback
  - ✓ decision making

Agent - Environment Interaction Loop

- Discrete time steps $t = 0, 1, 2, 3 \ldots$
- Agent receives (is in) state $S_t \in \mathcal{S}$
- Agent selects an action $A_t \in \mathcal{A}(S_t)$
- Agent receives reward $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$
  ...and finds itself in a new state S_{t+1}



- $S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots$
- We use R_{t+1} to denote the reward due to A_t (*next* reward)

Goals and rewards

- **Goal:** maximize cumulative reward
- **Immediate reward:** reward R_t at time step t
- Maximize expected cummulative reward, i.e. **return**:

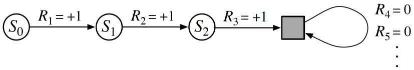
$$G = R_1 + R_2 + R_3 + \ldots + R_T$$

- Typically we seek to maximize **discounted return**:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

Unified view of episodic and non-episodic returns

$$G_t = \sum_{i=0}^{T} \gamma^i R_{t+i+1}$$



- If $T < \infty$: **episodic task**
  - T is the final time step
  - S_T is a *terminal state*
  - followed by a reset
- $\mathcal{S}^+$ denotes all states
- T can vary from episode to episode
- Unification: episode termination by transitioning to a special *absorbing state*

Returns of successive time steps

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

- Can we express G_t in terms of future returns?

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots \\ &= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} \ldots) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

Transition Function and Reward

Transition function:
Choosing action a in state s, what is the **probability of transitioning to state** s'?

$$p(s' \mid s, a) = \Pr\left\{S_{t+1} = s' \mid S_t = s, A_t = a\right\} = \sum_{r \in \mathcal{R}} p(s', r \mid s, a)$$

Reward function:
Choosing action a in state s and transitioning to s', what is the **immediate reward**?

$$r(s, a, s') = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s'] = \sum_{r \in \mathcal{R}} r \frac{p(s', r \mid s, a)}{p(s' \mid s, a)}$$

Important: $r(s, a, s')$ is a function but an expectation (average) over all possible rewards – typically and unless otherwise specified, we assume there is a single reward for each (s, a, s') and we can drop $\mathbb{E}$

Reward definitions

- $r(s, a, s')$: expected immediate reward on transition from s to s' under action a
- $r(s, a)$: expected immediate reward starting in s and choosing action a

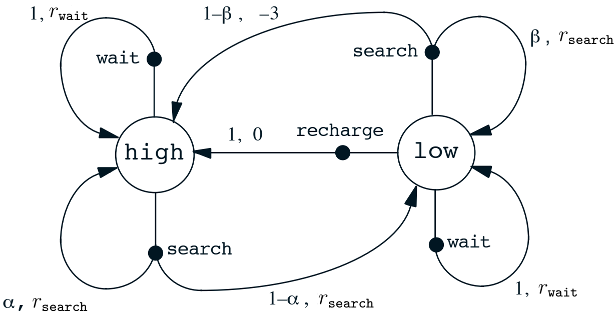
$$r(s,a) = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r \mid s, a)$$

- $r(s)$: expected immediate reward for *being* in state s
  - "bag of treasure" sitting on a grid-world square

Recycling robot MDP (Sutton & Barto)

- At each step, robot has a choice of three actions:
  - go out and search for a can
  - wait till a human brings it a can
  - go to charging station to recharge
- Searching is better (higher reward), but runs down battery.
  Running out of battery power is very bad and robot needs to be rescued
- Decision based on current state - is energy high or low
- Reward is number of cans (expected to be) collected, negative reward for needing rescue

Transition graph



$$\mathcal{S} = \{\text{high}, \text{low}\}$$
$$\mathcal{A}(\texttt{high}) = \{\text{search}, \text{wait}\}$$
$$\mathcal{A}(\texttt{low}) = \{\text{search}, \text{wait}, \text{recharge}\}$$
$$\mathcal{R} = \{r_{\text{search}}, r_{\text{wait}}, 0, -3\}$$

Tabular representation

s	a	s'	$p(s' \mid s, a)$	$r(s, a, s')$
high	search	high	α	$r_{\texttt{search}}$
high	search	low	$1-\alpha$	$r_{\texttt{search}}$
low	search	high	$1-\beta$	-3
low	search	low	β	$r_{\texttt{search}}$
high	wait	high	1	$r_{\texttt{wait}}$
high	wait	low	0	$r_{\texttt{wait}}$
low	wait	high	0	$r_{\texttt{wait}}$
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	$0.$

$$p(s' \mid s, a) = \Pr\left\{S_{t+1} = s' \mid S_t = s, A_t = a\right\}$$

$$r(s, a, s') = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a, S_{t+1} = s']$$

Optimality in MPDs

Policy

- *Policy* π maps states $s \in \mathcal{S}$ to probability distributions over actions $a \in \mathcal{A}$
- **Deterministic policy:** $a = \pi(s)$
- **Stochastic policy:** $\pi(a \mid s) = \Pr\{A_t = a \mid S_t = s\}$

Value under policy

Value of a state s under a policy π:

$$v_\pi(s) = \mathbb{E}_\pi\left[G_t \mid S_t = s\right]$$
$$= \mathbb{E}_\pi\left[\sum_{i=0}^{\infty} \gamma^i R_{t+i+1} \mid S_t = s\right] \text{ for all } s \in \mathcal{S}$$

$\mathbb{E}_\pi[\cdot]$ denotes the expectation of a random variable, given that the agent follows policy π

Expected Value and Mean

- **Summary statistics** are *deterministic* functions of random variables
- Examples are *mean* and *covariance*

Definition (Expected value)

Given a function $g : \mathbb{R} \to \mathbb{R}$ of a uni-variate continuous random variable $X \sim p(x)$ the *expected value* of g is defined as

$$\mathbb{E}_X[g(x)] = \int_{\mathcal{X}} g(x)p(x)dx$$

- If X is discrete then

$$\mathbb{E}_X[g(x)] = \sum_{\mathcal{X}} g(x)p(x)$$

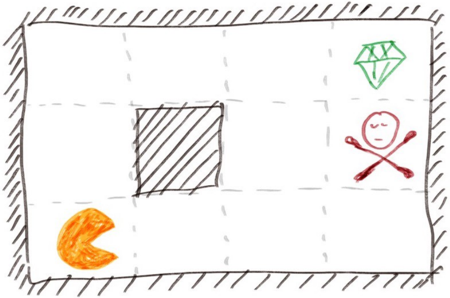
- If X is multivariate then

$$\mathbb{E}_X[g(\boldsymbol{x})] = \begin{bmatrix} \mathbb{E}_{X_1}[g(x_1)] \\ \vdots \\ \mathbb{E}_{X_D}[g(x_D)] \end{bmatrix} \in \mathbb{R}^D$$

- The **mean** is defined as

$$g(x) = x \implies \mathbb{E}_X[x] = \int_{\mathcal{X}} xp(x)dx$$

Example: grid world



- Rewards: -0.01, $+1$, -1
- Actions: N, E, S, W
- States: agent's location

Example: grid world

	1	2	3	4
3	0.812	0.868	0.918	1.000
2	0.762		0.660	-1.000
1	0.705	0.655	0.611	0.388

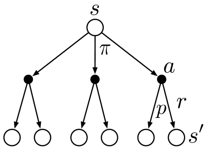
- Rewards: -0.01, $+1$, -1
- Actions: N, E, S, W
- States: agent's location

Action values

Value of taking action a in state s under a policy π:

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}_\pi\left[G_t \mid S_t = s, A_t = a\right] \\ &= \mathbb{E}_\pi\left[\sum_{i=0}^{\infty} \gamma^i R_{t+i+1} \mid S_t = s, A_t = a\right] \end{aligned}$$

Recursive relationship for v_π

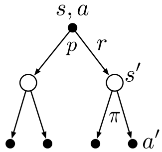


$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi \left[G_t \mid S_t = s \right] \\
&= \mathbb{E}_\pi \left[\sum_{i=0}^{\infty} \gamma^i R_{t+i+1} \mid S_t = s \right] \\
&= \mathbb{E}_\pi \left[R_{t+1} + \gamma G_{t+1} \mid S_t = s \right] \\
&= \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a) \Big[r + \gamma \mathbb{E}_\pi [G_{t+1} \mid S_{t+1} = s'] \Big] \\
&= \sum_a \pi(a \mid s) \sum_{s', r} p(s', r \mid s, a) \Big[r + \gamma v_\pi(s') \Big] \quad \text{for all } s \in \mathcal{S}
\end{aligned}
$$

This is the *Bellman equation for* v_π!

Recursive relationship for q_π

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[G_t \mid S_t = s, A_t = a\right] \\
&= \mathbb{E}_\pi\left[\sum_{i=0}^{\infty} \gamma^i R_{t+i+1} \mid S_t = s, A_t = a\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a\right] \\
&= \sum_{s',r} p(s',r \mid s,a)\Big[r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s', A_{t+1} = a'\right]\Big] \\
&= \sum_{s',r} p(s',r \mid s,a)\Big[r + \gamma \sum_{a'} \pi(a'|s') q_\pi(s',a')\Big]
\end{aligned}
$$



Relating $q_\pi \leftrightarrow v_\pi$

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[G_t \mid S_t = s, A_t = a\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a\right] \\
&= \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s', A_{t+1} = a'\right]\Big] \\
&= \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]\Big] \\
&= \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma v_\pi(s')\Big]
\end{aligned}
$$

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[G_t \mid S_t = s\right] \\
&= \sum_a \pi(a|s) q_\pi(s,a)
\end{aligned}
$$

Tansition Matrix

For a Markov state s and successor state s', the state transition probability is defined by $p : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$:

$$p(s' \mid s, a) = \Pr\left\{S_{t+1} = s', \mid S_t = s, A_t = a\right\}$$

State transition matrix $\mathcal{P}$ defines transition probabilities **from** all states s **to** all successor states s',

$$\mathcal{P} = \text{from} \begin{array}{c} \text{to} \\ \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ldots & \vdots \\ p_{n1} & \cdots & p_{nn} \end{bmatrix} \end{array}$$

where each row of the matrix sums to 1.

Bellman Equation in Matrix Form

The Bellman equation can be expressed concisely using matrices,

$$v = \mathcal{R} + \gamma \mathcal{P} v$$

where v is a column vector of values.

- The Bellman equation is a linear equation
- It can be solved directly:

$$v = (I - \gamma \mathcal{P})^{-1} \mathcal{R}$$

- Computational complexity is $O(n^3)$ for n states

Optimal policies and optimal value functions

- An **optimal policy** π_* has the highest/**optimal value** function $v_*(s)$
- Always choosing the action which yields highest **return**

$$v_*(s) = \max_{\pi} v_\pi(s) \text{ for all } s \in \mathcal{S}$$

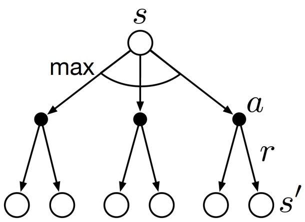
- **Optimal action-value function**:

$$\begin{aligned} q_*(s,a) &= \max_{\pi} q_\pi(s,a) \text{ for all } s \in \mathcal{S}, a \in \mathcal{A} \\ &= \max_{\pi} \mathbb{E}_\pi \Big[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = t \Big] \end{aligned}$$

Bellman optimality equation for v_*

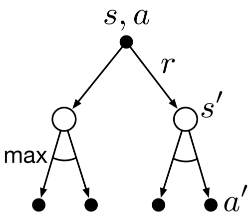
Value under optimal policy = expected return for best action from that state.

$$
\begin{aligned}
v_*(s) &= \max_a q_*(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}\left[G_t \mid S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a\right] \\
&= \max_a \mathbb{E}\left[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a\right] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma v_*(s')\Big]
\end{aligned}
$$



Bellman optimality equation for q_*

$$q_*(s,a) = \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a\right]$$
$$= \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma \max_{a'} q_*(s', a')\right]$$



Summary: reinforcement learning problem

- Agent, environment
- States, actions, rewards
- Policy $\pi(a \mid s)$: probability of choosing a in s
- Value $V(s)$: value of a state
- Action value $Q(s, a)$: value of a state-action pair
- Model/dynamics $p(s, a, s')$: probability of going from $s \rightarrow s'$ when choosing a
- Reward function $r(s, a, s') \rightarrow \mathbb{R}$: reward from choosing a in s and reaching s'
- Return G: sum of discounted future rewards
- Total future discounted reward $R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$
- What do want to learn?
  - value V or Q
  - policy
  - model
- Aim: Learn to maximize discounted sum of future rewards