Sequential Decision

Session 7: Markov decision processes and Q-learning

Igor Colin

October 2026

ESIEE Paris

Today

Part	Content	Time
Lecture	MDPs, Bellman equations, value iteration, Q-learning	50 min
Tutorial 7	Bellman equations and value iteration (proofs), Q-learning	30 min
Lab 7	value iteration, Q-learning on cliff walking	1 h
Projects	the five topics, rules, forming pairs	40 min

Part I: bandits

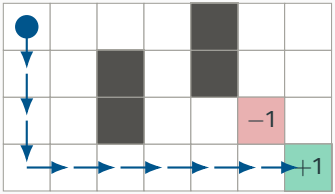
- One decision, one reward, and the situation does not change

Part II: reinforcement learning

- Each decision changes the state, hence the future rewards

Markov decision processes

When actions change the state



A robot moves in a maze: up, right, down or left.

- In a bandit, an arm does not change the situation. Here each move changes the **state**, and the reward comes at the end: a move has no value in itself
- The best move depends on where the robot is: a **policy** maps states to actions
- Other examples: games, inventory management, a dialogue, the generation of a text token by token

Markov decision processes

MDP

A set of states $\mathcal{S}$ and of actions $\mathcal{A}$, transition probabilities $P(s' \mid s, a)$, rewards $r(s, a)$, and a discount factor $\gamma \in [0, 1)$. At each step, in state S_t the agent plays A_t, receives $r(S_t, A_t)$ and moves to $S_{t+1} \sim P(\cdot \mid S_t, A_t)$.

Markov property

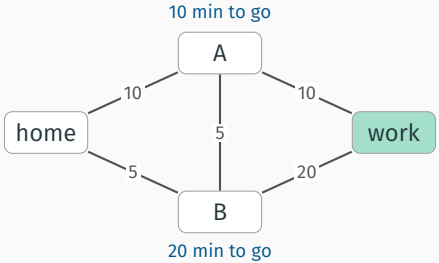
- The next state depends on the past only through (S_t, A_t)
- The state must contain all that matters (the position, not the path)

Objective: the discounted return

$$G = \sum_{t \geq 0} \gamma^t \, r(S_t, A_t)$$

A reward in t steps is worth γ^t times a reward now.

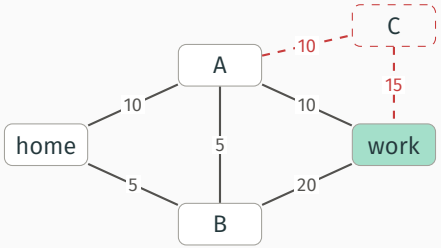
Evaluating a move



Every morning, we drive from home to work. At each crossroad, we choose the next road (times in minutes). Is “go to A” a good first move?

- A **policy** π says which road to take at each crossroad, for instance: from A, go to work; from B, go to work. The **value** of a crossroad is the time it still takes to reach work if we follow π: 10 from A, 20 from B
- The value of a **move**: the time of the road, plus the value of the crossroad where it leads. From home, “go to A” takes $10 + 10 = 20$, and “go to B” $5 + 20 = 25$
- If π goes from A to B instead, “go to A” takes $10 + 5 + 20 = 35$: the value of a move depends on what we do next

Random transitions



The road from A to work is closed with probability 0.3: diverted to C. The next state is random.

The value $V^\pi(s) = \mathbb{E}_\pi[G \mid S_0 = s]$ is an **average** over where we land:

- From A: $0.7 \times 10 + 0.3 \times (10 + 15) = 14.5$ minutes to go
- From home: via A, 24.5 min; via B, 25. With a closure probability of 0.4, B becomes better
- One equation per state: $V^\pi = (I - \gamma P_\pi)^{-1} r_\pi$ (lab, question 1; tutorial)

Bellman equations of a policy π (reward $r = -$time; $\gamma = 1$ here)

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a)\, V^\pi(s') \ , \qquad V^\pi(s) = Q^\pi\big(s, \pi(s)\big)$$

The best policy

Which policy is the best? Without closures, take at each crossroad the road to the shortest trip, *assuming we also drive best afterwards*:

$$\text{time}^\star(\text{home}) = \min\big(\underbrace{10 + \text{time}^\star(A)}_{\text{via A}},\ \underbrace{5 + \text{time}^\star(B)}_{\text{via B}}\big) \quad :$$

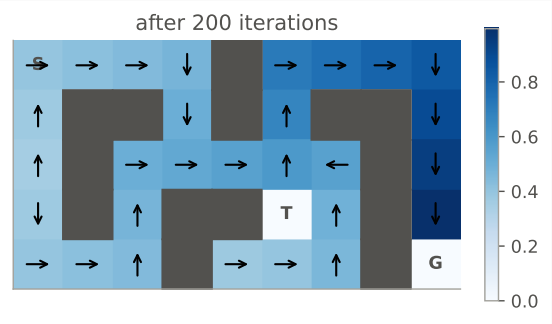
the equation of shortest paths (Bellman, 1958). With rewards, the min becomes a max.

Bellman optimality equation

$$V^\star(s) = \max_a Q^\star(s,a) \ , \qquad Q^\star(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a)\, V^\star(s')$$

- $V^\star = \max_\pi V^\pi$ in all states at once; to act, play $\arg\max_a Q^\star(s,a)$
- A bandit is an MDP with one state and $\gamma = 0$. In an MDP, the best immediate move is not always the best: home $\to$ B is the shortest first road

If we know the model: value iteration



Maze of the lab, slippery moves, $\gamma = 0.95$

If P and r are known, solve the optimality equation by applying it again and again:

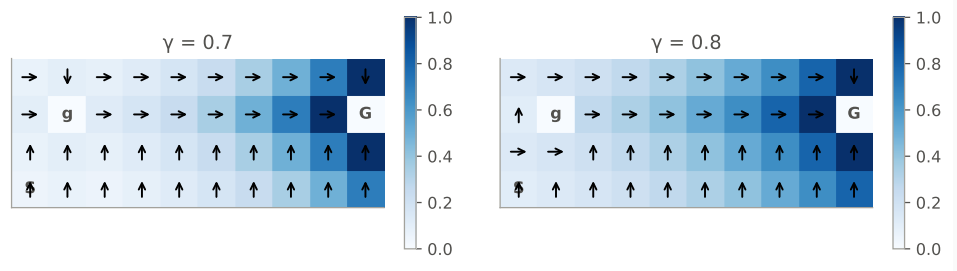
Value iteration

$V_0 = 0$, then $V_{k+1}(s) =$

$$\max_a \left(r(s,a) + \gamma \sum_{s'} P(s' \mid s,a)\, V_k(s') \right)$$

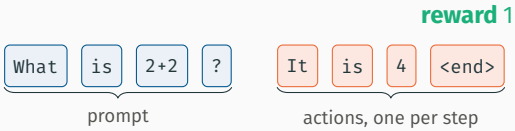
- Each iteration propagates the rewards one step further
- **Contraction**: $\|V_k - V^\star\|_\infty \leq \gamma^k \|V^\star\|_\infty$ (tutorial)

The discount factor



- Small reward 0.1 at 3 moves, large reward 1 at 11 moves: worth $0.1\gamma^2$ and γ^{10}
- The agent goes far when $\gamma^8 > 0.1$, that is, $\gamma > 0.75$
- γ sets the horizon, about $1/(1-\gamma)$ steps: it is part of the problem, not of the algorithm

Text generation as an MDP



- State: the prefix (prompt and tokens so far)
- Action: the next token, among $\sim 50\,000$
- Transition: deterministic, append the token

- Reward: at the end only, from a checker (verifiable answer) or a reward model learned from comparisons (session 6)
- The policy is the language model: session 9

Q-learning

Learning without the model

The agent does not know P and r: it only observes transitions (s, a, r, s').

Temporal difference target

$Q^\star(s,a) = \mathbb{E}\big[r + \gamma \max_{a'} Q^\star(s', a')\big]$: one observed transition gives a noisy target $r + \gamma \max_{a'} Q(s', a')$.

Q-learning (Watkins, 1989)

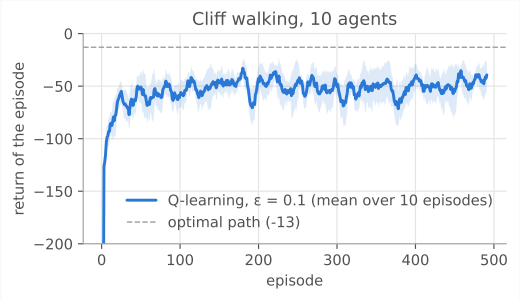
After each transition,

$$Q(s,a) \leftarrow Q(s,a) + \alpha\big(r + \gamma \max_{a'} Q(s', a') - Q(s,a)\big) \ ,$$

with the target r alone if s' is terminal. Play ε-greedy with respect to Q.

- The incremental mean of session 1, with a target that moves as Q improves
- Converges to $Q^\star$ if every pair (s, a) is played infinitely often and α decreases suitably

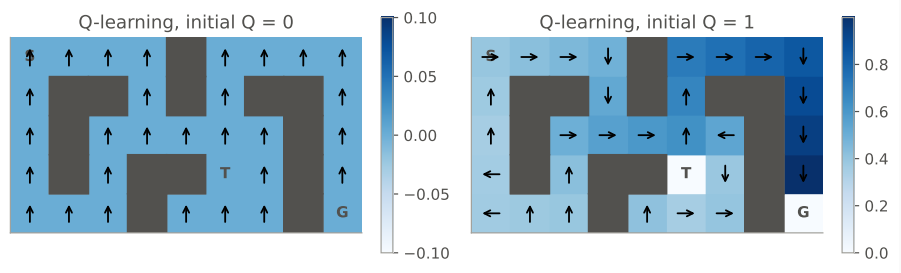
Cliff walking



Each move costs -1; the cliff C costs -100 and sends back to the start.

- After about 100 episodes, the greedy policy is the shortest path, along the cliff: -13
- During training, the return stays around -47: with $\varepsilon = 0.1$, a random move next to the cliff is often a fall
- Q-learning learns the greedy policy, not the one it plays: it is off-policy

Exploration is harder in an MDP



- The goal is 16 moves away: random moves almost never reach it, and with $Q = 0$ nothing is learned (left)
- **Optimistic initialization** ($Q = 1$): every untried action looks good, so the agent tries them in turn, as UCB (right)
- Systematic exploration in MDPs: bonuses, models of the transitions (project 3)

Summary

Method	Needs	Output
Policy evaluation	model, policy	V^π, by a linear system
Value iteration	model	$V^\star$, error $\leq \gamma^k \|V^\star\|_\infty$
Q-learning	transitions only	$Q^\star$, with enough exploration

Tutorial 7

- Bellman equations, contraction, value iteration
- The greedy policy is optimal
- Q-learning by hand

Lab 7

- Policy evaluation, value iteration
- Discount factor
- Q-learning, cliff walking

Reading

- Sutton and Barto, chapters 3, 4 and 6

Projects

Projects

	Topic	Sessions
1	Recommendation under drift: contextual bandits that forget	3, 5
2	Bayesian optimization: tuning a model with 40 trainings	2, 3
3	Exploration in reinforcement learning: optimism on hard MDPs	7
4	Model arena: finding the best models with few duels	6
5	Rating the players of a team game: Elo, newcomers, matchmaking	6

Organization

- In pairs, one topic
- Three guided tasks, each with a figure and a short answer
- Then “going further”: your choice of direction, from the suggestions or your own idea

Provided

- A starter notebook per topic: simulator, evaluation, plots; you complete the cells `...  # TODO`
- The subject (PDF): what is expected, what to read

Topics 1 and 2

1. Recommendation under drift

- 10 items, users with profiles; the appeals change at unknown times
- Task 1: LinUCB does not recover after a change
- Task 2: a sliding window; bias and variance
- Task 3: choose offline, by replay on logs (session 5)
- Going further: discounting instead of a window, detecting the changes

2. Bayesian optimization

- 4 hyperparameters of a small network, 40 noisy trainings
- Task 1: GP-UCB, with the Gaussian process given
- Task 2: against random search, on 20 seeds
- Task 3: exploration, the parameter β
- Going further: the kernel, expected improvement, cheap trainings (Hyperband)

Topics 3 and 4

3. Exploration in reinforcement learning

- A river to swim upstream, a slippery maze with a distant goal
- Task 1: ε-greedy fails, and why
- Task 2: optimism in Q-learning (initialization, count bonus)
- Task 3: optimism in a learned model (MBIE-EB)
- Going further: posterior sampling, longer rivers

4. Model arena

- 20 models, noisy annotators, some cyclic preferences
- Task 1: uniform duels, win rates and Bradley-Terry
- Task 2: focus the duels on the best models
- Task 3: when Bradley-Terry is wrong
- Going further: a stopping rule, Thompson sampling on the scores

5. Rating the players of a team game

- Matches of 5 against 5 in a league of 200 players who improve, leave, come back with a new account, play with their friends
- Task 1: Elo for teams, and the step k
- Task 2: a larger step for the newcomers
- Task 3: matchmaking with the ratings: fairer matches, better ratings?
- Going further: an uncertainty per player (Glicko, TrueSkill), the parties of friends, matches that are both fair and informative

Rules

To hand in

- The completed notebook, executed, running in less than 15 minutes
- For each task and for your own direction: the code, the figure, an answer of a few sentences, with error bars

Grading

- The tasks: code, figures, and the quality of the answers
- The direction you chose to go further
- The defense: 10 minutes, with questions on the code

Calendar

- November 4: pairs and topics, by email
- Session 8 (November 25): Task 1 done, questions
- December 8: notebook due
- Session 10 (December 9): defenses

Language models

- Allowed for writing code, but the defense checks that you understand every line of it