

COMP767: Reinforcement Learning

Michael Noukhovitch

Winter 2018,

Notes written from Doina Precup's lectures.

Contents

1	Introduction	3
1.1	Definitions	3
1.2	Key Factors of RL	3
1.3	Classical Challenges	3
2	Bandit	3
2.1	Definition	3
2.2	Action Selection	4
2.3	Learning Rules	4
2.3.1	Averaging	4
2.3.2	Recency-Weighted Average	5
2.3.3	Optimistic	5
2.3.4	Upper Confidence Bound	5
2.3.5	Gradient-Bandit Algorithms	5
2.3.6	Associative Search	5
2.4	Evaluations	6
2.5	Conclusions	6
3	Markov Decision Processes	6
3.1	Markov	6
3.2	Returns and Values	7
3.3	Policy Iteration	7
3.4	Asynchronous DP	7
4	Monte Carlo Methods	7
4.1	MC Policy Evaluation	7
4.2	On-Policy MC Control	7
4.3	Advantages	7

1 Introduction

1.1 Definitions

Reinforcement learning is:

agent-oriented learning learning by interacting with an environment

trial and error only given delayed evaluative feedback

science of the mind one which is neither natural science nor applied technology

Framework:

1. agent perceives the **state** of the environment
2. based on the state, it chooses an **action**
3. the action gives the agent a **reward**
4. a **policy** aims to maximize the agent's **long term expected reward**

1.2 Key Factors of RL

- trial and error search
- environment is stochastic
- reward may be delayed
- balancing exploration and exploitation

1.3 Classical Challenges

- reward
- information is sequential
- delayed consequences
- balancing exploration/exploitation
- non-stationarity
- fleeting nature of time and online data

2 Bandit

2.1 Definition

One-armed bandit Simplest RL problem

- pull the lever
- get some reward

- choose the best lever!

k-armed bandit extends to k arms

- at every time step t , choose an action A_t from k possibilities
- receive a reward R_t dependent only on the action taken (i.i.d)
- $q_*(a) = \mathbb{E}[R_t | A_t = a], \forall a \in 1, \dots, k$

2.2 Action Selection

greedy the action with the current highest expected value (best one so far)

exploitation choosing the greedy action

exploration choosing not the greedy action

ϵ -greedy balance explore/exploit by choosing exploration (random) with probability ϵ

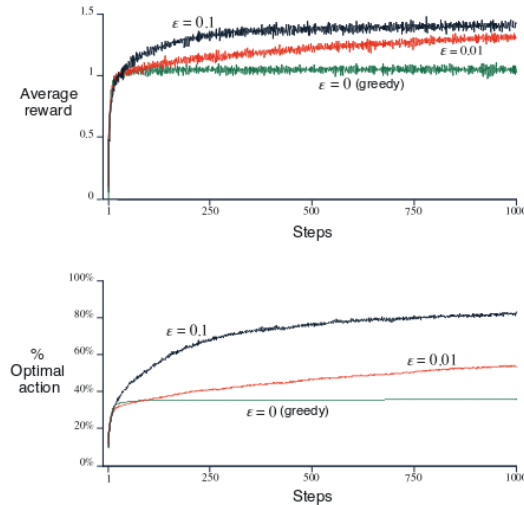


Figure 1: ϵ -greedy methods on 10-arm bandit

2.3 Learning Rules

Learn the best policy by learning the reward for an action

2.3.1 Averaging

For a single action, update the new estimate based on old estimate and step size (α), with all actions being equal

$$Q_{n+1} = Q_n + \alpha(R_n - Q_n)$$

2.3.2 Recency-Weighted Average

stationary if the true action values DO NOT change over time

if our bandit is non-stationary, then we need to put more weight on recent samples

$$Q_{n+1} = (1 - \alpha)^n Q_1 + \sum_{i=1}^n \alpha (1 - \alpha)^{n-i} R_i$$

2.3.3 Optimistic

Previously we assumed $Q_1(a) = 0$, but we can start optimistically (e.g. $Q_1(a) = 5$) to encourage early exploration

2.3.4 Upper Confidence Bound

Reduce exploration over time after starting confident

- estimate upper bound on true action values
- select the action with the largest upper bound

$$A_t = \operatorname{argmax}_a [Q_t(a) + c \sqrt{\frac{\log t}{N_t(a)}}]$$

2.3.5 Gradient-Bandit Algorithms

Don't need to learn specific rewards, just learn the **preference** $H_t(a)$, and try and make the probability of choosing an action $\pi_t(a)$ be proportional to it.

$$\begin{aligned} \pi_t(a) &\propto e^{H_t(a)} \\ &= \frac{e^{H_t(a)}}{\sum_b e^{H_t(b)}} \end{aligned}$$

if the reward for an action is better than average, increase its preference

$$H_{t+1} = H_t(a) + \alpha(R_t - \bar{R}_t)(1_{a=A_t} - \pi_t(a))$$

where $\bar{R}_t = \text{average } R_i$

2.3.6 Associative Search

associative a task where the situation/state of the agent changes the reward for an action

contextual bandit not just trial-and-error search, but also association between state and action values

full reinforcement learning trial-and-error search, association between state and action, and actions affecting the next state of the agent

2.4 Evaluations

regret the difference between best option and the one we chose $\max_a q_*(a) - q_t(a)$

expected total regret $\mathbb{E}[\sum_t \text{regret}_t]$ (optimal for UCB, Thomson sampling)

best response regret for T experimental trials after policy is fixed

2.5 Conclusions

- simple methods that can be built on
- learn from feedback
- appear to have a goal

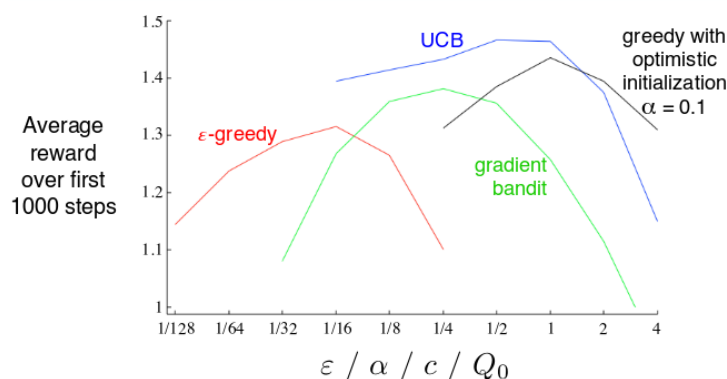


Figure 2: bandit algorithm comparison

3 Markov Decision Processes

3.1 Markov

markov property future independent of past given present

markov chain memoryless random process with states S and transition probabilities P , $\langle S, P \rangle$

markov reward process markov chain with values, rewards for states R , discount factor γ

return sum of discounted rewards $G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} \dots$

value function long-term value of state s , $v(s) = E[G_t | S_t = s]$

3.2 Returns and Values

3.3 Policy Iteration

3.4 Asynchronous DP

- choose

4 Monte Carlo Methods

4.1 MC Policy Evaluation

learn $v_\pi(s)$ given some number of episodes under π which contain s , using the average of the rewards after visiting s

every visit MC average returns for every visit to s

first visit MC average returns for only the first visit to s (in an episode)

4.2 On-Policy MC Control

on policy learn about policy currently executing

boltzmann exploration choose with probability $\exp(-\frac{P(s)}{T})$ where T is a temperature.
Over time, decrease from high T (equiprobable) to low T (biased towards best action)

4.3 Advantages

Several advantages over DP

- can learn directly from interaction with environment
- no need for full models
- no need to learn about all states
- less harmed by violating Markov property