

# Autonomous and Adaptive Systems

## Monte Carlo Methods

Mirco Musolesi

[mircomusolesi@acm.org](mailto:mircomusolesi@acm.org)

# Monte Carlo Methods

- ▶ Monte Carlo methods are ways of solving the reinforcement learning problems based on averaging the sample returns.
- ▶ We focus on episodic tasks.
  - ▶ Only on the completion of episodes are values estimates and policies changed.
- ▶ Monte Carlo methods sample and average *returns* for each state-action pair much like the multi-armed bandits methods and average *rewards* for each action.
  - ▶ However, we now consider *multiple states*; and
  - ▶ The return after taking an action in one state depends on the actions taken in later states in the same episodes.
  - ▶ In other words: this is the *full* reinforcement learning problem.

# Monte Carlo Methods

- ▶ If you have the full knowledge of the MDP you can *compute* the value functions (see Bellman equation, dynamic programming).
- ▶ We assume that we do not have full knowledge of the underlying MDP.
  - ▶ This is the case in general because the underlying dynamics and characteristics of the system are unknown (e.g., robot exploration) or because the system is too complex (e.g., games).
- ▶ Since we do not have full knowledge, we need to *learn* the values functions.

# Monte Carlo Methods

- ▶ We consider three problems:
  - ▶ The *prediction problem*: the estimation of  $v_\pi$  and  $q_\pi$  for a fixed policy  $\pi$ .
  - ▶ The *policy improvement problem*: the estimation of  $v_\pi$  and  $q_\pi$  while trying at the same time to improve the policy  $\pi$ .
  - ▶ The *control problem*: the estimation of an optimal policy  $\pi_*$ .

# Monte Carlo Prediction

- ▶ Goal: learning the state-value function for a given policy.
  - ▶ Recall: the value of a state is the expected return (expected cumulative future discounted reward) from that state.
- ▶ Obvious/simple solution: average the returns observed after visiting that state. As more returns are observed, the average should converge to the expected value.

# Monte Carlo Prediction

- ▶ More formally: we want to estimate  $v_{\pi}(s)$ , the value of a state  $s$  under policy  $\pi$ , given a set of episodes obtained by following  $\pi$  and passing through  $s$ .
- ▶ Each occurrence of a state  $s$  in an episode is called a *visit* to  $s$ .
- ▶ A state  $s$  can be visited multiple times.
- ▶ The first time a state  $s$  is visited in an episode is called the first visit to  $s$ .
- ▶ The *first-visit Monte Carlo method* estimates  $v_{\pi}(s)$  as the average of the returns following first visits to  $s$ , whereas the *every-visit Monte Carlo method* averages the returns following all the visits to  $s$ .

# First-visit Monte Carlo Prediction

Input: a policy  $\pi$  to be evaluated

Initialise:

$V(s) \in \mathbb{R}$  arbitrarily for all  $s \in \mathcal{S}$

$Returns(s) \leftarrow$  empty list for all  $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following  $\pi$ :  $S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode  $t = T - 1, T - 2, \dots, 0$ :

$G \leftarrow \gamma G + R_{t+1}$

If  $S_t$  does not appear in  $S_0, S_1, \dots, S_{t-1}$ :

Append  $G$  to  $Returns(S_t)$

$V(S_t) \leftarrow average>Returns(S_t)$

# Multi-visit Monte Carlo Prediction

- ▶ Every-visit Monte Carlo would be the same except without the check for  $S_t$  having occurred earlier in the episode.
- ▶ Both first-visit Monte Carlo and every-visit Monte Carlo converge to  $v_{\pi}(s)$  as the number of visits (or first visits) to  $s$  goes to infinity.



# Monte Carlo Estimation of Action Values

- ▶ The estimation of a state value makes sense when you have a model of the system.
  - ▶ With a model, state values alone are sufficient to determine a policy.
  - ▶ Without a model, it is necessary to estimate the value of each action in order for the value to be useful in suggesting a policy.
- ▶ The policy evaluation problem for action values is to estimate  $q_{\pi}(s, a)$ , the expected return when starting in state  $s$ , taking action  $a$  and then following policy  $\pi$ .
  - ▶ Remember we assume that the policy  $\pi$  is fixed.

# Monte Carlo Estimation of Action Values

- ▶ The methods for the Monte Carlo estimation of action values are essentially the same as those presented for state value, but now we talk about visits to the state-action pair instead of to a state.
- ▶ A state-action pair  $s, a$  is said to be visited in an episode if the state  $s$  is visited and the action  $a$  is taken in it.
- ▶ The first-visit method Monte Carlo method averages the returns following the first time in each episode that the state was visited and the action was selected.

# Maintaining Exploration

- ▶ But there is a problem: many state-action pairs might never be visited.
- ▶ If  $\pi$  is a deterministic policy, then in following  $\pi$ , we will observe returns only for one of the actions of each state.
  - ▶ No return to average -> no improvement with experience.
  - ▶ We cannot compare alternatives, because no alternatives are explored.
- ▶ To compare alternatives, we need to estimate the value of all the actions from each state, not only the one we currently prefer (according to our policy).
- ▶ How can we address this problem?

# Maintaining Exploration

- ▶ This is the general problem of *maintaining exploration*.
- ▶ One way to do this is to have the episodes starting in a state-action pair and that every pair has non-zero probability of being selected as start.
- ▶ This ensures that asymptotically (infinite number of episodes), all the state-action pairs will be visited an infinite number of times.

# On-policy and Off-policy Exploration

- ▶ The method described above is useful, but it cannot be applied in general.
  - ▶ Think about the case for example where you have exploration with the environment. You cannot start by “jumping” to a certain state-action pair at the beginning.
- ▶ The most common alternative is to ensure that all the state-action pairs are explored anyway.
- ▶ We need to explore these states, *not following* the current policy (for example with a stochastic policy). In other words, the exploration is not performed *on-policy* as done until now in this lecture, but *off-policy*.
  - ▶ Various methods are possible, such as *off-policy prediction via Importance Sampling* (not covered in this module - see Sutton and Barto Chapter 5.5).

# Policy Improvement

- ▶ We will focus now back on on-policy exploration, i.e., the policy is used to make decisions and to explore the various states.
- ▶ Until now we assumed that the policy was fixed. However, the policy itself can be *improved* while learning the value functions and, potentially, we might have to aim at obtaining an optimal policy.
- ▶ Methods used for improving a policy in order to reach the optimal policy are usually referred to as *Monte Carlo control*.

# Policy Improvement

- ▶ Remember until now we assumed that the policy was fixed.
  - ▶ Given that policy, we estimate the value functions.
- ▶ Now we consider how to improve the policy starting from an *on-policy method*.
  - ▶ The policy that we use to make decisions is that we are trying to improve. We do not use a separate policy to explore the state-action pairs (that would be an *off-policy method*).
- ▶ Policy improvement is done by making the policy greedy with respect to the current value functions.

# Policy Improvement

- ▶ In this case we have an action-value function.
  - ▶ No model is needed to construct the greedy policy.
- ▶ For any action-value function  $q$ , the corresponding greedy policy is the one that, for each  $s \in \mathcal{S}$ , deterministically chooses an action with maximal action-value:

$$\pi(s) \leftarrow \arg \max_a Q(s, a)$$

- ▶ We have formal results that ensures that this process of policy improvement leads to optimal policy (usually referred to Monte Carlo control).



# Monte Carlo Control without Exploring Starts

- ▶ An alternative method for Monte Carlo control (i.e., to achieve the optimal policy) is to use an  $\epsilon$ -greedy approach.
- ▶ Most of the time the action that has the maximal estimated action value is chosen but with a probability  $\epsilon$  an action is chosen at random.
  - ▶ This means that all non-greedy actions are given the minimal probability of selection  $\frac{\epsilon}{|\mathcal{A}(s)|}$  is given to all the non-greedy actions.
- ▶ Following these mechanism, it can be guaranteed that we will have an improvement of the policy (policy improvement theorem).

# Monte Carlo Control with Exploring Starts

Initialise:

$\pi(s) \in \mathcal{A}(s)$  arbitrarily for all  $s \in \mathcal{S}$

$Q(s, a) \in \mathbb{R}$  arbitrarily for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

$Returns(s, a) \leftarrow$  empty list for all  $s \in \mathcal{S}, a \in \mathcal{A}(s)$

Loop forever (for each episode):

Choose  $S_0 \in \mathcal{S}, A_0 \in \mathcal{A}(S_0)$  randomly

Generate an episode from  $S_0, A_0$  following  $\pi$ :  $S_0, A_0, R_1, S_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode  $t = T - 1, T - 2, \dots, 0$  :

$G \leftarrow \gamma G + R_{t+1}$

If  $S_t$  does not appear  $S_0, A_0, R_1, A_1, R_2, \dots, S_{T-1}, A_{T-1}$ :

Append  $G$  to  $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow average>Returns(S_t, A_t)$

$\pi(S_t) \leftarrow \arg \max_a Q(S_t, a)$

# References

- ▶ Chapter 5 of Sutton and Barto. Introduction to Reinforcement Learning. Second Edition. MIT Press. 2018.