hypothesis of Thm X holds.

EE6106 LECTURE 23 Date: 17 Apr 2024

Classical RL $\rightarrow$ Q-learning, SARSA

Discounted Cost Setting (S, A, P, C, α)

Assumption : Underlying MDP is unichain. For average cost MDP, so such assumption was made when the horizon was assumed to be made.

In the RL setting, we require a connectivity assumption as mistakes are compulsory and sub-optimal actions have to be taken.

Flavours of RL : 1) On-policy learning
2) Off-policy learning

On-policy $\longrightarrow$ The learning algorithm is in control and decides which action to take.

Off-policy $\longrightarrow$ The entity which decides what action to take is different that the learning agent. We can still learn from the feedback signals received by someone else.

Flavours of RL algorithms : 1) Model-based
2) Model-free

Model-based $\longrightarrow$ Algorithm is learning a "model" and hence learning (P, C)

Model-free $\longrightarrow$ Not learning P, C but directly trying to learn optimal policy. Learn value functions directly.

Note that we assume both S and A to be finite. Not doing stuff such as function approximation.

Lecture 23

1. Simple model-based on-policy approach

<u>Idea</u>: Play random actions until all (s,a) pairs are seen

Then, Construct $\hat{P}_{ij}(a)$, $\hat{C}(i,a)$

$$\hat{P}_{ij}(a) = \frac{\#\text{ played a and landed in j from i}}{\#\text{played a from i}}$$

For the policy, play ϵ-greedy policy. i.e. w.p. ε_t play randomly
& w.p. $1-\varepsilon_t$ play optimal action for $(S, A, \hat{P}, \hat{C}, \alpha)$

ε_t is chosen to be a decreasing function of time.

Typically, $\varepsilon_t \sim \frac{1}{t}$ or slower, such as $\varepsilon_t \sim \frac{1}{\sqrt{t}}$. (Borel-Cantelli Lemma)

$\downarrow$

ensure infinite exploration w.p. 1

This ensures $\hat{P} \xrightarrow{a.s.} P$, $\hat{C} \xrightarrow{a.s.} C$

The memory footprint is $O(S^2 A)$

ϵ_t is chosen to be a decreasing function of time.

Typically, $\epsilon_t \sim \frac{1}{t}$ or slower, such as $\epsilon_t \sim \frac{1}{\sqrt{t}}$. (Borel-Cantelli Lemma)

This ensures $\hat{P} \xrightarrow{a.s.} P$, $\hat{C} \xrightarrow{a.s.} C$ ⟵ ensure infinite exploration w.p. 1

The memory footprint is $O(S^2 A)$

↓

storing estimates of P.

$\boxed{2}$ Q-Learning (model-free off-policy algorithm)

Learns the Q-function, with a memory requirement of $O(SA)$.

$$Q^*(i,a) = C(i,a) + \alpha \sum P_{ij}(a) V^*(j)$$
$$= C(i,a) + \alpha \sum P_{ij}(a) \left[\min_{a'} Q^*(j,a')\right]$$

This is the Bellman equation in terms of the Q function

$$\hat{Q}(x_t, A_t) \longleftarrow (1-\gamma_t)\hat{Q}(x_t, A_t) + \gamma_t \left[C_t + \alpha \min_{a'} \hat{Q}(x_{t+1}, a')\right]$$

γ_t is the step-size parameter, gives memory of the earlier event

γ_t is also chosen as a decreasing function of time

Note that $\min_{a'} \hat{Q}(x_{t+1}, a')$ is an estimate of $V^*(X_{t+1})$.

Need: $\sum \gamma_t = \infty$, $\sum \gamma_t^2 < \infty$

Lecture 23

Q: What do we need for $\hat{Q}$ to converge to Q?

→ In the long run, every (s,a)-pair should be visited almost surely. Thus, we cannot have a stationary policy which drives Q-learning

"Tracks ODE" : $\dot{Q} = TQ - Q$, where T is an operator

$$\underbrace{Tu(i,a)}_{\text{a candidate function}} = C(i,a) + \alpha \sum P_{ij}(a) \left[\min_{a'} u(j,a') \right]$$

Note that the Bellman equation can be expressed as $TQ = Q$

The memory footprint now is $O(s,a)$.

$\boxed{3}$ SARSA (State, Action, Reward, State, Action)

Model-free, on-policy algorithm

$$A_t = \begin{cases} \text{Play random action w.p. } \varepsilon_t \\ \arg\min_a \hat{Q}(X_t, a) \quad \text{w.p. } 1-\varepsilon_t \end{cases}$$

$$\hat{Q}(X_t, A_t) \leftarrow (1-\gamma_t)\hat{Q}(X_t, A_t) + \gamma_t \left[C_t + \alpha \hat{Q}(X_{t+1}, A_{t+1}) \right]$$

There is no minimization term in SARSA. Note that the next step we WILL perform minimization w.p. $1-\varepsilon_t$

... action selection

$$A_t = \begin{cases} \text{Play random action} \\ \arg\min_a \hat{Q}(X_t, a) & \text{w.p. } 1-\varepsilon_t \end{cases}$$

$$\hat{Q}(X_t, A_t) \leftarrow (1-\gamma_t)\,\hat{Q}(X_t, A_t) + \gamma_t\left[C_t + \alpha\,\hat{Q}(X_{t+1}, A_{t+1})\right]$$

There is no minimization term in SARSA. Note that the next step we WILL perform minimization w.p. $1-\varepsilon_t$

Note that for Q-learning, we can use any action selection algorithm as long as it performs infinite exploration.

Typically choose $\varepsilon_t \sim \frac{1}{t}$ & $\gamma_t \sim \frac{1}{t}$

$$\Rightarrow \hat{Q} \xrightarrow{a.s.} Q$$

Note that these algorithms are notoriously slow. This is because only entry gets filled in one time-step

Function approximation : expresses value function as a linear combination of some basis functions.

4 Actor - Critic algorithm

GPI $\longrightarrow$ Generalized Policy Iteration

Original policy iteration searches for greedy improvements on the policy

Actor: Policy improvement (slower timescale)

Critic: Policy evaluation (faster timescale)

| Lecture 23

You have two timescales, one with a small & one with a large step size

In practice, use step size γ_t for actor & β_t for critic.

$$\beta_t = o(\gamma_t) \quad \gamma_t = o(\beta_t)$$

$\downarrow$

Little ~~of~~ Oh

SARSA

If we do, off-policy SARSA, under some stationary policy π, we will learn the value function corresponding to the policy $\pi \longrightarrow Q^{\pi}$ and V^{π}

All of these algorithms can be generalized for a time-average MDP setting.