Lecture 11

Lecture 11-13 Fast Reinforcement Learning

Idea

Move towards algorithms that can be more computational-wise and sample-wise efficient?

Also mean that the optimal decision can be calculated quickly (require less time to react) and need less data to compute on.

Key terms

1. Bandits
2. Multi-armed bandits
3. Regret
4. Upper confidence bound (UCB)
5. Hoeffding's Inequality

Bandits

Actions have no influence on
next observation & reward

Slot machine

Bandits

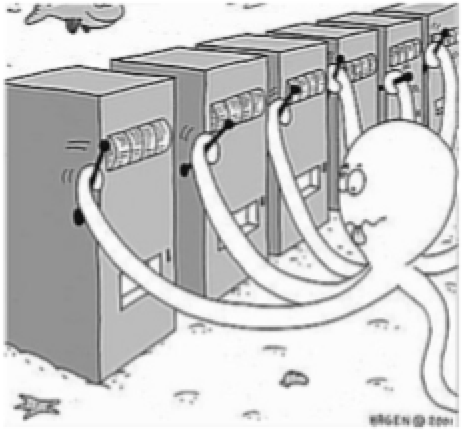
Actions have no influence on
next observation & reward

Slot machine = one-armed bandit

= each action is independent

Multi-armed Bandits

Multiple slot machine with multiple arms



Can choose which arm(action) to pull and the slot machine will return the rewards

Regret

Opportunity loss

Just an indicator for
algorithm evaluation

Regret

Opportunity loss

$$l_t = \mathbb{E}[V^* - Q(a_t)]$$

Difference of reward from current pull of the arm with the optimal value

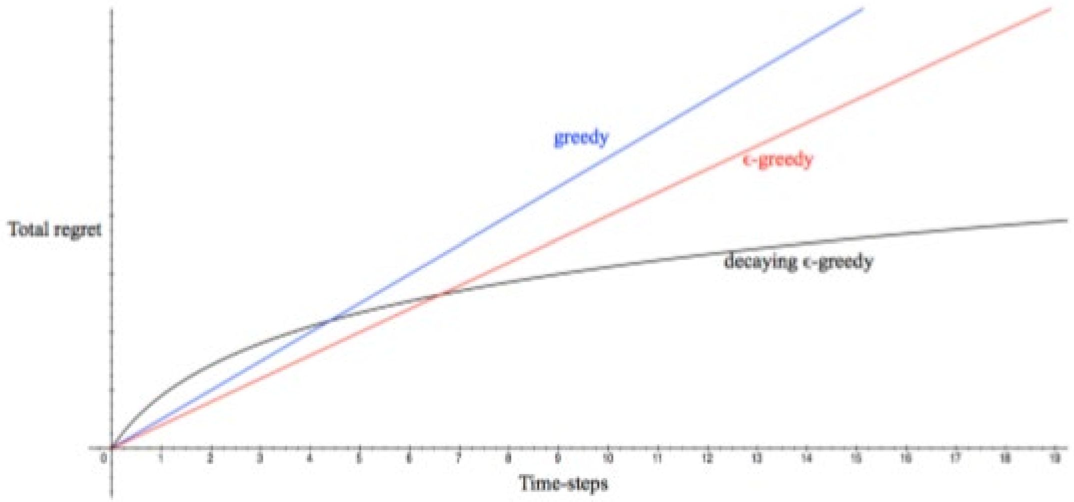
Or in other words this part is the Gap $\Delta_i = V^* - Q(a_i)$

Then calculate total regret = regret for all timesteps

$$\begin{aligned} L_t &= \mathbb{E}\left[\sum_{\tau=1}^{t} V^* - Q(a_\tau)\right] \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)](V^* - Q(a)) \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)]\Delta_a \end{aligned}$$

Count of the given action (arm pulled)

Upper confidence bound



Aim is to get sublinear regret

Because linear regret just means getting worse over time (continuously not picking the optimal action)

But first, In order to define the upper confidence bound...

Hoeffding's Inequality

$$\mathbb{P}\left[\mathbb{E}[X] > \bar{X}_n + u\right] \leq \exp(-2nu^2)$$

$$\bar{X}_n + u \geq E[X] \quad \text{w.prob} \geq 1 - \delta/t^2$$

Upper bound can deviate from the expected value by more than a certain amount

Hoeffding's Inequality

$$\mathbb{P}\left[\mathbb{E}[X] > \bar{X}_n + u\right] \leq \exp(-2nu^2)$$

$$\bar{X}_n + u \geq E[X] \quad \text{w.prob} \geq 1 - \delta/t^2$$

Where...

$$u = \sqrt{\frac{1}{2n}\log(t^2/\delta)}$$

Upper confidence bound

$$U_t(a_t) = \hat{Q}(a_t) + \sqrt{\frac{1}{2n(a_t)}\log\left(t^2/\delta\right)}$$

Upper confidence bound

$$U_t(a_t) = \hat{Q}(a_t) + \sqrt{\frac{1}{2n(a_t)}\log(t^2/\delta)}$$

UCB1 algorithm is choosing the action (arm) with the best upper bound value

$$a_t = \arg\max_{a \in \mathcal{A}} [\hat{Q}(a) + \sqrt{\frac{2 \log t}{N_t(a)}}]$$

Upper confidence bound

$$U_t(a_t) = \hat{Q}(a_t) + \sqrt{\frac{1}{2n(a_t)}\log(t^2/\delta)}$$

UCB1 algorithm is choosing the action (arm) with the best upper bound value

$$a_t = \arg\max_{a \in \mathcal{A}} [\hat{Q}(a) + \sqrt{\frac{2 \log t}{N_t(a)}}]$$

Key terms of lecture 12

1. Bayesians bandit
2. Thompson Sampling
3. Probably Approximately Correct (PAC)

Bayesian

$$\overset{\text{Posterior}}{P(A|B)} = \frac{\overset{\text{Likelihood}}{P(B|A)} * \overset{\text{Prior}}{P(A)}}{\underset{\text{Evidence}}{P(B)}}$$

Bayesian bandit

$$\underbrace{P(A|B)}_{\text{Posterior}} = \frac{\overbrace{P(B|A)}^{\text{Likelihood}} * \overbrace{P(A)}^{\text{Prior}}}{\underbrace{P(B)}_{\text{Evidence}}}$$

$$p(\phi_i|r_{i1}) = \frac{p(r_{i1}|\phi_i)p(\phi_i)}{\int_{\phi_i} p(r_{i1}|\phi_i)p(\phi_i)d\phi_i}$$

Bayesian bandit

Likelihood

Prior

Posterior

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Evidence

Data likelihood

Reward of particular action depends on this parameter

$$p(\phi_i|r_{i1}) = \frac{p(r_{i1}|\phi_i)p(\phi_i)}{\int_{\phi_i} p(r_{i1}|\phi_i)p(\phi_i)d\phi_i}$$

Distribution of rewards

Bayesian bandit

$$\underbrace{P(A|B)}_{\text{Posterior}} = \frac{\overbrace{P(B|A)}^{\text{Likelihood}} * \overbrace{P(A)}^{\text{Prior}}}{\underbrace{P(B)}_{\text{Evidence}}}$$

Conjugate....

$$p(\phi_i|r_{i1}) = \frac{p(r_{i1}|\phi_i)p(\phi_i)}{\int_{\phi_i} p(r_{i1}|\phi_i)p(\phi_i)d\phi_i}$$

Bayesian bandit

$$Regret(\mathcal{A}, T; \theta) = \sum_{t=1}^{T} \mathbb{E}\left[Q(a^*) - Q(a_t)\right]$$

$$BayesRegret(\mathcal{A}, T; \theta) = \mathbb{E}_{\theta \sim p_0}\left[\sum_{t=1}^{T} \mathbb{E}\left[Q(a^*) - Q(a_t) | \theta\right]\right]$$

Thompson Sampling

Because computing optimal action from posterior can be difficult, a simpler approach would be as following

Thompson Sampling

Initialize prior over each arm a, $p(\mathcal{R}_a)$
loop
    For each arm a **sample** a reward distribution $\mathcal{R}_a$ from posterior
    Compute action-value function $Q(a) = \mathbb{E}[\mathcal{R}_a]$
    $a_t = \arg\max_{a \in \mathcal{A}} Q(a)$
    Observe reward r
    Update posterior $p(\mathcal{R}_a|r)$ using Bayes law
end loop

Easier understood through Bernoulli toy example

In attempt to understand...

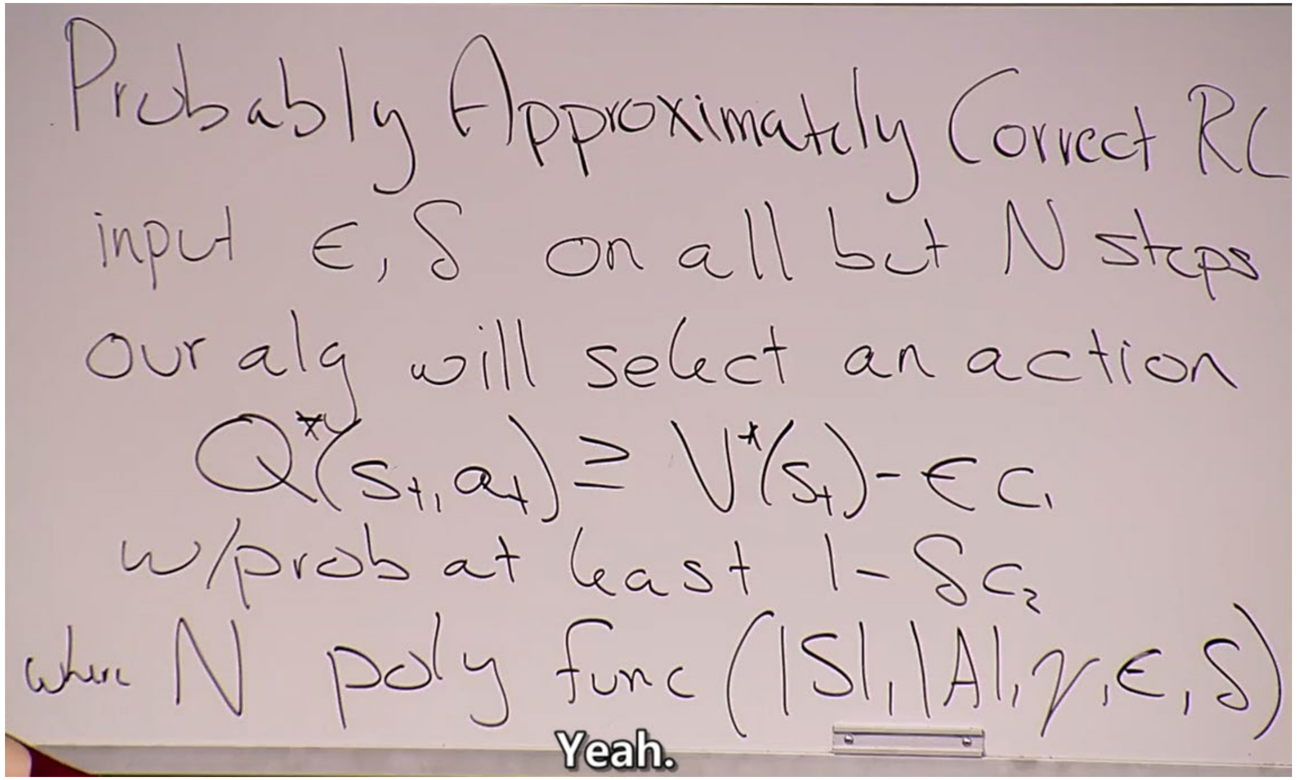
PAC Probably Approximately Correct

Input epsilon and delta in all but N steps

The algorithm selects action which its true Q value will be greater than the best possible value of that state subtracted by epsilon

With probability of at least 1 – delta

N is a polynomial function of size of S, A and gamma, epsilon and delta

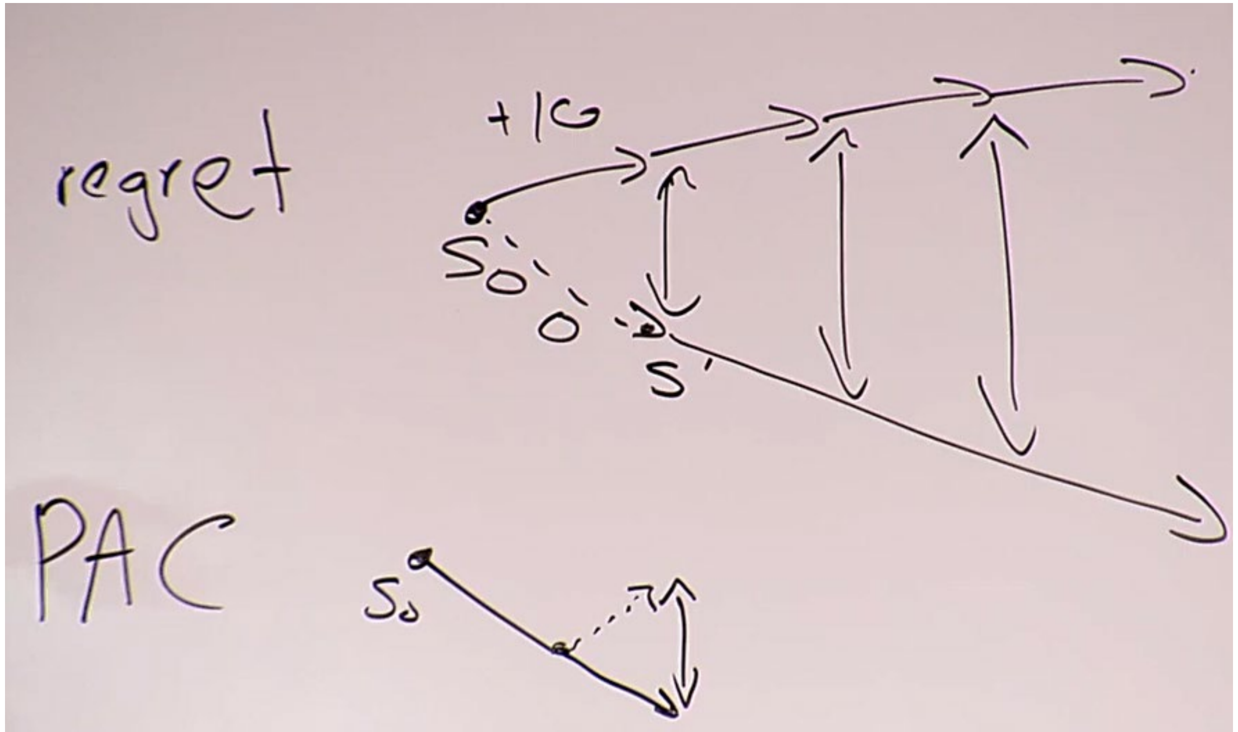


In attempt to understand…

PAC Probably Approximately Correct

PAC can be though of as making the most out of its circumstances

While regret judges on whether good decisions have been made from the beginning or not.



In attempt to understand…

PAC Probably Approximately Correct

Sufficient condition for PAC

1. Optimism $\underbrace{Q_t(s,a)}$ $\geq Q^*(s,a) - \epsilon \quad \forall s,a,t$

computed value should be optimistic with respect to real Q values

2. Accuracy $V_t(s) - \underbrace{V^{\pi_t}_{M}}(s) \leq \epsilon$

will define further. MDP related to true MDP & MDP defined in MBIE-EB

3) Bounded learning complexity:
- total # of updates to Q
- # times visit an "unknown" (s,a) pair

bounded by $\xi(\epsilon, \delta)$

In attempt to understand…

MBIE-EB Model Based Interval Estimation with Exploration Bonus

$$
\begin{array}{l}
\text{Given } \epsilon,\ \delta,\ m \\
\beta = \frac{1}{1-\gamma}\sqrt{0.5 \ln(2|S||A|m/\delta} \\
n_{sas}(s,a,s') = 0 \ s \in S,\ a \in A,\ s' \in S \\
rc(s,a) = 0,\ n_{sa}(s,a) = 0,\ \tilde{Q}(s,a) = 1/(1-\gamma)\ \forall\ s \in S,\ a \in A \\
t = 0,\ s_t = s_{init} \\
\textbf{loop} \\
\quad a_t = \arg\max_{a \in \mathcal{A}} Q(s_t, a) \\
\quad \text{Observe reward } r_t \text{ and state } s_{t+1} \\
\quad n_{sa}(s_t,a_t) = n(s_t,a_t) + 1,\ n_{sas}(s_t,a_t,s_{t+1}) = n_{sas}(s_t,a_t,s_{t+1}) + 1 \\
\quad rc(s_t,a_t) = \frac{rc(s_t,a_t)n_{sa}(s_t,a_t)+r_t}{(n_{sa}(s_t,a_t)+1)} \\
\quad \hat{R}(s,a) = \frac{rc(s_t,a_t)}{n(s_t,a_t)} \text{ and } \hat{T}(s'|s,a) = \frac{n_{sas}(s_t,a_t,s')}{n_{sa}(s_t,a_t)}\ \forall s' \in S \\
\quad \textbf{while} \text{ not converged } \textbf{do} \\
\quad\quad \tilde{Q}(s,a) = \hat{R}(s,a) + \gamma \sum_{s'} \hat{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a') + \underbrace{\frac{\beta}{\sqrt{n_{sa}(s,a)}}}\ \forall\ s \in S,\ a \in A \\
\quad \textbf{end while} \\
\textbf{end loop}
\end{array}
$$

Backup

Bonus

Transition model