Monday, March 7, 2022

(Heavily inspired by the Stanford RL Course of Prof. Emma Brunskill, but all potential errors are mine.)

SMU: Lecture 5

Bandits (Introduction)

Efficient Learning

much how fast

with a simulator or when playing computer games, but not, e.g., when optimizing an advertisement campaign...

still interesting and used in practice).

So far we only cared about whether our RL algorithms converge, not that

- We assumed that failed experiments (episodes) do not cost us anything (except, maybe, time). That is the case, e.g., when learning some strategy
- We can generally study efficient learning for MDPs but in this course we will only look at efficient learning for multi-armed bandits (which are simpler but

Multi-Armed Bandits



1



2

We can choose actions $\{1,2,3,4\}$ and each of them leads to a different distribution of rewards.





4

3

P[R = r | A = i]

Setting

Multi-armed bandit is essentially a degenerate MDP that contains a single state. **Definition:** A multi-armed bandit is given by: the action at time t.

distribution.

- A set A containing m actions a_1, a_2, \ldots, a_m (each can be thought of as "pulling an arm").
- Reward distributions $P[R_t = r | A_t = a]$, that is the distribution of rewards at time t given

- At each step, the agent takes an action and receives a reward sampled from the above
- The *informal* goal is to maximize the reward $\sum R_t$ of course, this is a random variable. t=1

Example

them.

This can be modelled using multi-armed bandits:

(different advertisements will have different quality).

- Your PR team created m different advertisements. You are now supposed to show these advertisements to people and maximize the number of times they click on

 - The action a_i corresponds to displaying the *i*-th advertisement from our collection.
 - We get reward 1 when the person clicks on the advertisement and 0 otherwise.
- Clearly, the probabilities $P[R_t = 1 | A_t = 1]$, $P[R_t = 1 | A_t = 2]$, ... will be different

Regret (1/3)

Action-value: $Q(a) = \mathbb{E}[R_t | A_t = a].$

Similar to MDPs where we had $Q^{\pi}(s, a)$. However, we do not need s because we now have only one state. So we could rewrite it as $Q^{\pi}(a)$. But then, since the action only affects the immediate reward and not to which state we get, the whole notion of policy is not very important for Q in this setting, so we drop that as well and end up with $Q(a) = \mathbb{E}[R_t | A_t = a]$.

- **Optimal value:** $a \in A$
- $a^* = \arg \max Q(a).$ **Optimal action:** $a \in A$
- $L_t = V^* Q(A_t).$ **Regret:**

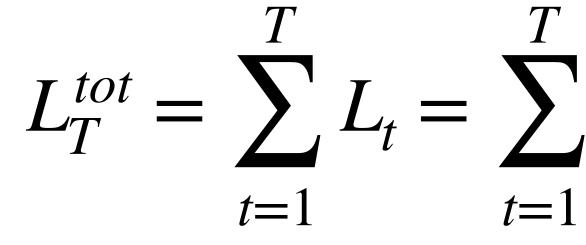
That is, regret is the "opportunity loss" at time t. Note that we use expected value in the definition of regret (recall how we defined Q(a)). That means we are not measuring regret directly in terms of what we observe. Since the parameters of bandits will generally be unknown, it also means we will not be able to compute regret directly.

Regret (2/3)

 $V^* = \max Q(a) = \max \mathbb{E}[R_t | A_t = a].$ $a \in A$

Regret (3/3)

Total regret:



Minimizing total regret is the same as maximizing the expected sum of rewards (i.e. return).

$L_T^{tot} = \sum_{t=1}^{T} L_t = \sum_{t=1}^{T} (V^* - Q(A_t)).$

Example

Consider again the example with advertisements, say we have 2 different advertisements that we can use, so $A = \{a_1, a_2\}$.

Suppose that:

So $\mathbb{E}[R_t | A_t = a_1] = 0.8$, $\mathbb{E}[R_t | A_t = a_2] = 0.5$. Let us have the following deterministic sequence of actions: What is the total regret of this episode? We have $V^* = 0.8$, $V^* - Q(a_1) = 0$, $V^* - Q(a_2) = 0.8 - 0.5 = 0.3$. So the total regret is: $0 + 0 + 0.3 + 0 + 0.3 + 0 + 0 + \dots + 0 = 0.6$.

- $P[\text{Person t clicks on ad} | A_t = a_1] = 0.8$, $P[\text{Person t clicks on ad} | A_t = a_2] = 0.5$

What We Want... (1/2)

We want to find algorithms where the regret will grow slowly with the number of time steps taken.

Note that:

When regret does not grow at all after some time, that means that we are already taking the optimal action.

Regret is the difference between best possible return and the return under our strategy. So when the regret grows slowly, it means we are already doing quite well.

What We Want... (2/2)

If we knew the expectations $\mathbb{E}[R_t | A_t = a]$ then the problem would be trivial, but it would not be reinforcement learning.

also not clear how long we should be estimating (because that actually depends on the values of $\mathbb{E}[R_t | A_t = a]$)... So we will need something smarter.

We could try to first estimate $\mathbb{E}[R_t | A_t = a]$ by taking actions completely randomly. However, then in this first part we would incur high regret and it is

Greedy Methods (Why They Would Not Work)

Initialization: Do several passes over all actions and compute estimates $\hat{Q}(a)$ for all $a \in A$. Maintain counter N(a) with the number of times an action was used. While (some stopping condition):

Select the action $a_t \in A$ which maximizes $\hat{Q}(a)$.

Use the selected action and observe r_t .

Set $N(a_t) := N(a_t) + 1$. Set $\hat{Q}(a_t) := \hat{Q}(a_t) + \frac{1}{N(a_t)}(r_t - Q(a_t))$.**

$$\star \star \left(\underbrace{\frac{Q(a_{t})}{=\frac{1}{N(a_{t})-1}(r_{i_{1}}+\ldots+r_{i_{t-1}})}}_{=\frac{1}{N(a_{t})-1}(r_{i_{1}}+\ldots+r_{i_{t-1}})} + \frac{1}{N(a_{t})}r_{i_{t}} - \frac{1}{N(a_{t})}Q(a_{t})}{Q(a_{t})} = \frac{\frac{N(a_{t})(r_{i_{1}}+\ldots+r_{i_{t-1}}) + (N(a_{t})-t)r_{i_{t}}}{(N(a_{t})-1)N(a_{t})}}{(N(a_{t})-1)N(a_{t})}} = \frac{1}{N(a_{t})}(r_{i_{1}}+r_{i_{2}}+\ldots+r_{i_{t-1}})}$$

Greedy Algorithm

$$\frac{1}{N(a_t)}Q(a_t) = \frac{N(a_t)(r_{i_1} + \dots + r_{i_{t-1}}) + (N(a_t) - t)r_{i_t} - (N(a_t) - 1)\frac{1}{N(a_t) - 1}(r_{i_1} + \dots + t)r_{i_{t-1}}}{(N(a_t) - 1)N(a_t)}$$



Why Greedy Will Not Work Well

Example (Continue with our previous example): $\mathbb{E}[R_t | A_t = a_1] = 0.8, \mathbb{E}[R_t | A_t = a_2] = 0.5.$

actions a couple of times).

(which can happen if we are unlucky in the initialization).

will grow linearly with time in this case.

- This will be similar to why purely greedy methods do not work well for RL (as we saw before, where we solved the problem by using ε -greedy methods.
- For greedy methods, we need some initialization (e.g. passing over all the
- Suppose that our initial estimates for Q are $\hat{Q}(a_1) = 0$ and $\hat{Q}(a_2) = 0.5$
- Then we will never select a_1 even though it is the optimal action. So regret

ɛ-Greedy Methods (Also not that great...)

Similarly to what we did in the previous lectures...

Initialization: Do several passes over all actions and compute estimates $\hat{Q}(a)$ for all $a \in A$. Maintain counter N(a) with the number of times an action was used. While (some stopping condition):

With probability $1 - \varepsilon$:

Select the action $a_t \in A$ which maximizes $\hat{Q}(a)$. Else:

Select an action $a_t \in A$ uniformly at random.

Use the selected action and observe r_t .

Set $N(a_t) := N(a_t) + 1$. Set $\hat{Q}(a_t) := \hat{Q}(a_t) + \frac{1}{N(a_t)}(r_t - Q(a_t)).$

E-Greedy (Basic Idea)

Regret of *E***-Greedy Methods**

regret growing linearly with the number of time steps - in every step we have probability $\varepsilon = \frac{\varepsilon}{|A|}$ of picking a suboptimal action (assuming no ties) which will incur a regret of at least $V^* - \max Q(a)$ $a \neq a^*$

So also not great...

We might try to set ε to be a function of t (as we did before) but it turns out to be tricky and need to know a lot about Q(a)'s in advance.

If we keep ε constant during the run of the ε -greedy algorithm then we will incur

Optimism Under Uncertainty

UCB Algorithm: Basic Idea

Upper-Confidence Bound (UCB) Algorithm

will change with time, that is why it is indexed by t).

take the action arg max $U_t(a)$. $a \in A$

After observing the reward, update the estimates.

- For every action $a \in A$, maintain an upper bound $U_t(a)$ (the upper bound
- In every time step t, take the action that has the maximum upper bound, i.e.

UCB Algorithm

Initialization:

Take every action $a \in A$ once and record the rewards in $\hat{Q}(a)$. t := 1

Loop:

Compute upper confidence bounds for all actions $a_i \in A$:

$$U_{t}(a_{i}) = \hat{Q}(a_{i}) + \sqrt{\frac{1}{2N(a_{i})} \log \frac{2t^{2}}{\delta}}$$

Use the action $a_t = \arg \max U_t(a)$ and observe the reward r_t . $a \in A$ Update $N(a_t) := N(a_1) + 1$ Update $\hat{Q}(a_t) := \hat{Q}(a_t) + \frac{1}{N(a_t)}(r_t - Q(a_t)).$

t := t + 1

With probability at least $1 - 2\delta m$, we have for the regret of the UCB algorithm:

 $L_T^{tot} \leq 21$

UCB Theorem

$$\int \frac{Tm}{2} \log \frac{T^2}{\delta}.$$

Sublinear regret!!!!

Next Part - Proof (It will not be part of exam)

Proof (1/12)

Claim: If all upper bounds $U_t(a_1), U_t(a_2), \ldots, U_t(a_m)$ satisfy $U_t(a_i) \ge Q(a_i)$, i.e. if none of them underestimates the true value, then for the action a_t selected at time t, it must hold

Easy to see why...

$U_t(a_t) \ge U(a^*) \ge Q(a^*) = V^*.$

Proof (2/12)

First, we will state an auxiliary statement (which you probably know from other courses).

Theorem (Hoeffding's Inequality): Let X_1, X_2, \ldots, X_N be independent random variables bounded on the interval [*a*; *b*]. Let $\overline{X}_N = \frac{1}{N} \sum_{i=1}^N X_i$. Then it holds $P\left[\overline{X}_N - \mathbb{E}[\overline{X}_N] \ge \xi\right] \le \exp\left(-\frac{1}{2}\right)$ $P\left[\mathbb{E}[\overline{X}_N] - \overline{X}_N \ge \xi\right] \le \exp\left(-\frac{1}{2}\right)$ $P\left[\left|\overline{X}_{N} - \mathbb{E}[\overline{X}_{N}]\right| \ge \xi\right] \le 2 \exp\left[\left|\overline{X}_{N}\right|\right]$

$$\frac{2N\xi^2}{(b-a)^2}\bigg),$$
$$\frac{2N\xi^2}{(b-a)^2}\bigg),$$
$$\left(\frac{2N\xi^2}{(b-a)^2}\right)$$

Proof (3/12)

i.e. number of times a_i was used.

We have
$$\mathbb{E}[\hat{Q}_t(a_i)] = Q(a_i).$$

We will want to find ξ_t (one value for each *t*) such that

$$P\left[\left|Q(a_i) - \hat{Q}(a_i)\right| \ge \xi_t\right] \le 2\exp\left(-\frac{2N_t(a_i)\xi_t^2}{(b-a)^2}\right) = \frac{\delta}{t^2},$$

where t is the current number of time steps.

Our \overline{X}_N will be $\hat{Q}_t(a_i)$, i.e. the estimate of $\hat{Q}(a_i)$, and our N will therefore be $N_t(a_i)$,

Proof(4/12)

We have

$P\left[\left|Q(a_i) - \hat{Q}(a_i)\right| \ge \xi_t\right]$

 $\frac{2N(a_i)}{(b-a_i)}$

For simplicity we will now assume that a = 0, b = 1.

$$\leq 2 \exp\left(-\frac{2N_t(a_i)\xi_t^2}{(b-a)^2}\right) = \frac{\delta}{t^2},$$

$$\frac{\xi_t^2}{z} = \log \frac{\delta}{2t^2},$$

$$\xi_t = (b-a)\sqrt{\frac{1}{2N_t(a_i)}\log\frac{2t^2}{\delta}}$$

Proof(5/12)

That is, the upper bounds $U_t(a_i)$ will be:

$$U_t(a_i) = \hat{Q}(a_i) + \sqrt{\frac{1}{2N_t(a_i)} \log \frac{2t^2}{\delta}}.$$

And we will also have lower bounds $L_t(a_i)$:

$$L_t(a_i) = \hat{Q}(a_i) - \sqrt{\frac{1}{2N_t(a_i)} \log \frac{2t^2}{\delta}}.$$

Proof (6/12)

 \leq

Let A_t be the action selected at time t. We will now bound the probability that at least some of the bounds are incorrect (we will see in a moment why we want this).

$$P\left[\bigvee_{t=1}^{T}\bigvee_{i=1}^{m}Q(a_{i})\notin[L_{t}(a_{i});U_{t}(a_{i})]\right]$$

$$\leq\sum_{t=1}^{T}\sum_{i=1}^{m}P[|Q(a_{i})-\hat{Q}_{t}(a_{i})|>\xi$$

 $\xi_t] \le \sum_{t=1}^T \sum_{i=1}^m \frac{\delta}{t^2} = m\delta \sum_{t=1}^T \frac{1}{t^2}.$

We can now use the famous identity $\sum_{t=1}^{\infty} \frac{1}{t^2} = \frac{\pi^2}{6}$ (which is smaller than 2).**

So we can bound:

$$P\left[\bigvee_{t=1}^{T}\bigvee_{i=1}^{m}U_{t}(a_{i})\notin\left[L_{t}(a_{i});U_{t}(a_{i})\right]\right]\leq 2m\delta.$$

That means that the probability that all lowe least $1 - 2m\delta$.

We will use this in a moment.

** We actually do not need this fancy resu slide)

Proof (7/12)

That means that the probability that all lower and upper bounds are valid at all time steps is at

** We actually do not need this fancy result to get the constant 2 (see the additional

Proof (8/12)

Let A_t be the **action selected at time** t.

We will now bound the probability that at least one of the upper bounds $U_1(A_1)$, $U_2(A_2)$, ... is lower than $U(a^*)$.

We can notice that the event that at least one action has wrong confidence bounds over the course of T time steps, formally written as

$$\bigvee_{t=1}^{T} \bigvee_{i=1}^{m} U_t(a_i) \notin [L_t(a_i); U_t(a_i)]$$

is a necessary condition for at least one of the upper bounds $U_1(A_1)$, $U_2(A_2)$, ... to be lower than $U(a^*)$.

Therefore we can bound this probability also by $1 - 2\delta m$.

Proof (9/12)

Let us now compute the regret of this algorithm:

$$\text{Regret}(T) = \sum_{t=1}^{T} \left(Q(a^*) - Q(A_t) \right) = \sum_{t=1}^{T} \left(U_t(A_t) - Q(A_t) + Q(a^*) - U_t(A_t) \right)$$

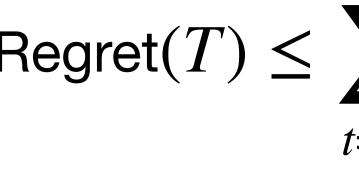
slide!) Hence we can bound the above as:

$$\operatorname{Regret}(T) \leq \sum_{t=1}^{T} \left(U_t(A_t) - Q(A_t) \right).$$

We have that $Q(a^*) < U_t(A_t)$ with probability at least $1 - 2m\delta$ (from the previous

Proof(10/12) $\operatorname{Regret}(T) \leq \sum_{t} \left(U_t(A_t) - Q(A_t) \right).$ t=1Recall that we defined $U_t(a_i) = \hat{Q}(a_i) + \sqrt{\frac{1}{2N_t(a_i)} \log \frac{2t^2}{\delta}}$ for all $a_i \in A$.

Now we will play with



Hence we get

$$\operatorname{Regret}(T) \leq \sum_{t=1}^{T} \left(\hat{Q}(A_t) + \sqrt{\frac{1}{2N_t(A_t)} \log \frac{2t^2}{\delta}} - Q(A_t) \right).$$

Proof (11/12)

Now we need to do something with

$$\operatorname{Regret}(T) \leq \sum_{t=1}^{T} \left(\hat{Q}(A_t) + \sqrt{\frac{1}{2N_t(A_t)} \log \frac{t^2}{\delta}} - Q(A_t) \right).$$

Since we have that, with probability at least $1 - 2\delta m$, we have for all $a_t \in A$

$$\left| \hat{Q}(a_t) - Q(a_t) \right| \le \sqrt{\frac{1}{2N_t(A_t)} \log \frac{t^2}{\delta}}.$$

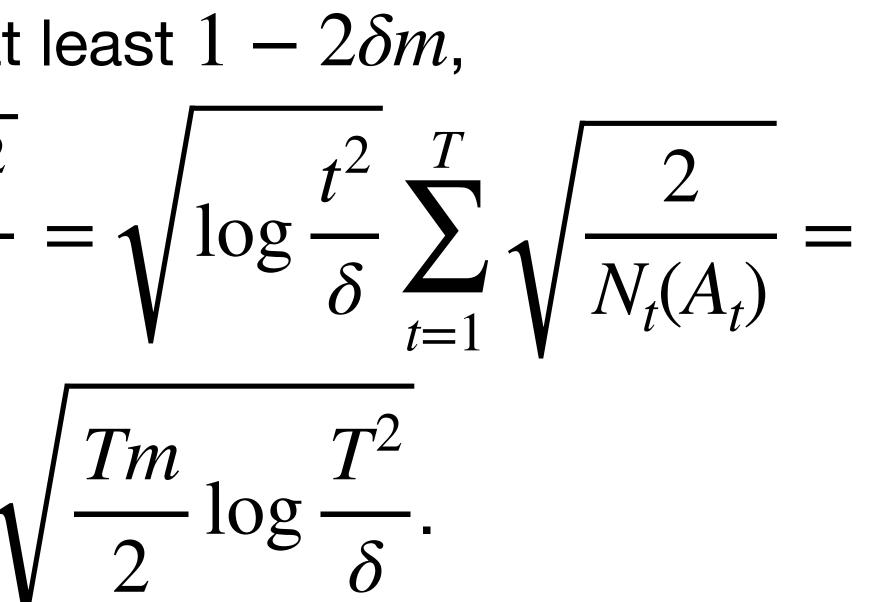
We can bound the regret, with probability at least $1 - 2\delta m$, as

$$\operatorname{Regret}(T) \leq \sum_{t=1}^{T} 2\sqrt{\frac{1}{2N_t(A_t)} \log \frac{t^2}{\delta}} = \sum_{t=1}^{T} \sqrt{\frac{2}{N_t(A_t)} \log \frac{t^2}{\delta}}.$$

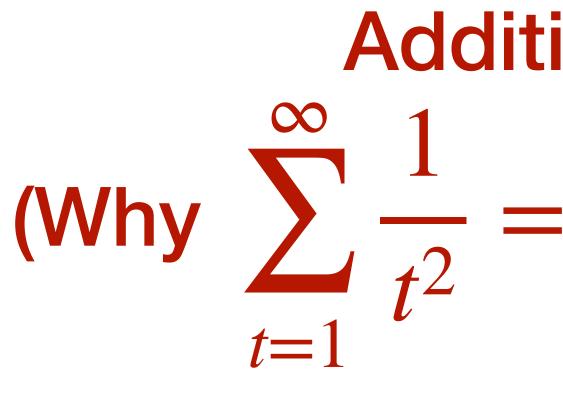
Proof (12/12)

Finally we have, with probability at least $1 - 2\delta m$,

$$\operatorname{Regret}(T) \leq \sum_{t=1}^{T} \sqrt{\frac{2}{N_t(A_t)} \log \frac{t^2}{\delta}} = \sqrt{2 \log \frac{t^2}{\delta}} \sum_{i=1}^{m} \sum_{j=1}^{N_T(a_i)} \sqrt{\frac{1}{j}} \leq 2\sqrt{\frac{1}{\delta}}$$

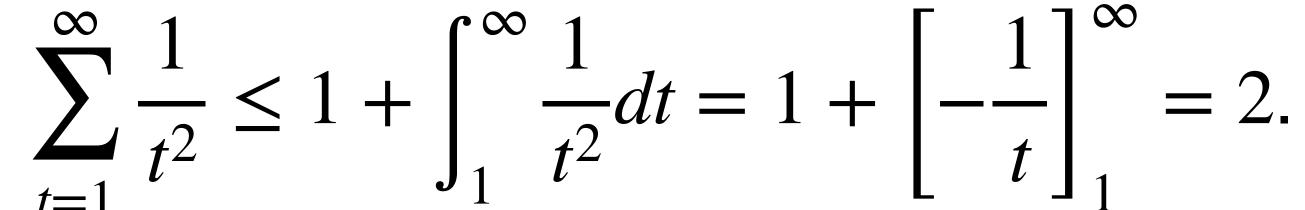


Sublinear regret!!!!



Bounding

```
(Why \sum_{t=1}^{\infty} \frac{1}{t^2} = \frac{\pi^2}{6} is not needed)
```



If you want to know more...

University Press, 2020.

Lattimore, Tor, and Csaba Szepesvári. Bandit algorithms. Cambridge

Available online: <u>https://tor-lattimore.com/downloads/book/book.pdf</u>

END OF SLIDES