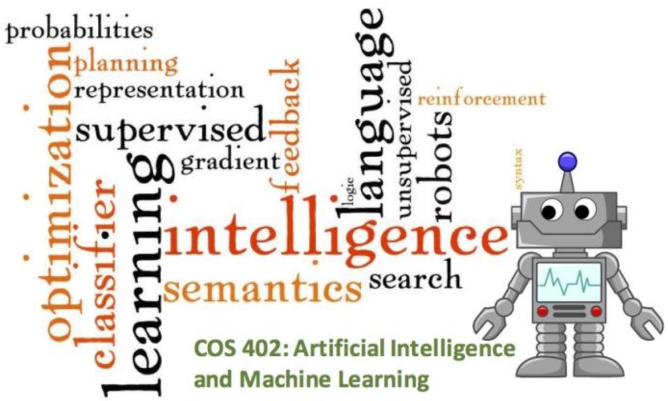


COS 402 – Machine Learning and Artificial Intelligence
Fall 2016

Lecture 22

Exploration & Exploitation in Reinforcement Learning: MAB, UCB, Exp3

How to balance exploration and exploitation in reinforcement learning

- Exploration:
  - try out each action/option to find the best one, gather more information for long term benefit
- Exploitation:
  - take the best action/option believed to give the best reward/payoff, get the maximum immediate reward given current information.
- Questions: Exploration-exploitation problems in real world?
  - Select a restaurant for dinner
  - …

Balancing exploration and exploitation

- Exploration:
  - try out each action/option to find the best one, gather more information for long term benefit
- Exploitation:
  - take the best action/option believed to give the best reward/payoff, get the maximum immediate reward given current information.
- Questions: Exploration-exploitation problems in real world?
  - Select a restaurant for dinner
  - Medical treatment in clinical trials
  - Online advertisement
  - Oil drilling
  - …

What's in the picture?

Picture is taken from Wikipedia

Multi-armed bandit problem

- A gambler is facing at a row of slot machines. At each time step, he chooses one of the slot machines to play and receives a reward. The goal is to maximize his return.

A row of slot machines in Las Vegas

Multi-armed bandit problem

- Stochastic bandits
  - Problem formulation
  - Algorithms
- Adversarial bandits
  - Problem formulation
  - Algorithms
- Contextual bandits

Multi-armed bandit problem

- Stochastic bandits:
  - K possible arms/actions: $1 \le i \le K$,
  - Rewards $x_i(t)$ at each arm i are drawn iid, with an expectation/mean u_i, unknown to the agent/gambler
  - $x_i(t)$ is a bounded real-valued reward.
  - Goal : maximize the return(the accumulative reward.) or minimize the expected regret:
  - Regret = $u^* T - \sum_{t=1}^{T} E[x_{i_t}(t)]$, where
    - $u^* = \max_i[u_i]$, expectation from the best action

Multi-armed bandit problem: algorithms?

- Stochastic bandits:
  - Example: 10-armed bandits
  - Question: what is your strategy? Which arm to pull at each time step t?

Multi-armed bandit problem: algorithms

- Stochastic bandits: 10-armed bandits
- Question: what is your strategy? Which arm to pull at each time step t?
- 1. Greedy method:
  - At time step t, estimate a value for each action
    - $Q_t(a) = \frac{\textit{sum of rewards when a taken prior to t}}{\textit{number of times a taken prior to t}}$
  - Select the action with the maximum value.
    - $A_t = \underset{a}{\operatorname{argmax}}\, Q_t(a)$
  - Weaknesses?

Multi-armed bandit problem: algorithms

- 1. Greedy method:
  - At time step t, estimate a value for each action
    - $Q_t(a) = \frac{sum\ of\ rewards\ when\ a\ taken\ prior\ to\ t}{number\ of\ times\ a\ taken\ prior\ to\ t}$
  - Select the action with the maximum value.
    - $A_t = \underset{a}{\operatorname{argmax}}\, Q_t(a)$
- Weaknesses of the greedy method:
  - Always exploit current knowledge, no exploration.
  - Can stuck with a suboptimal action.

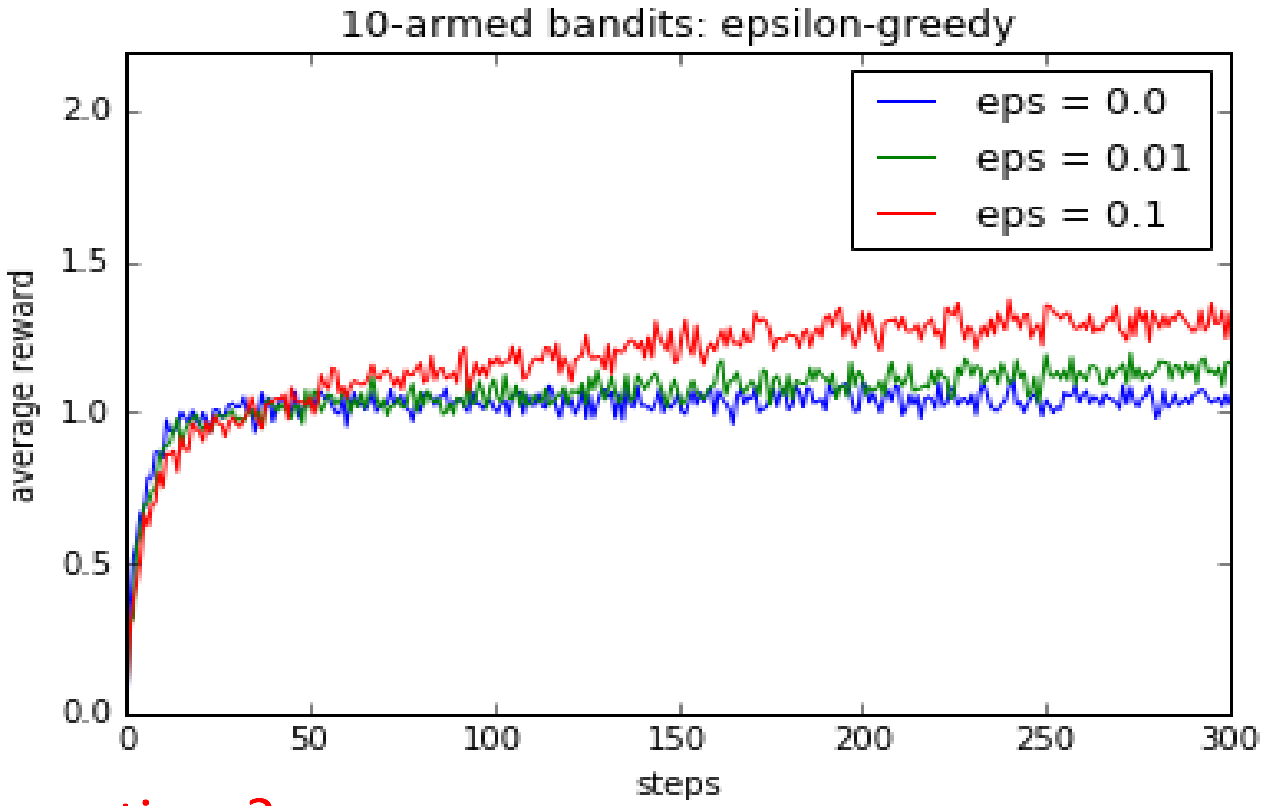
Multi-armed bandit problem: algorithms

- 2. ε-Greedy methods:
  - At time step t, estimate a value for each action
    - $Q_t(a) = \frac{sum\ of\ rewards\ when\ a\ taken\ prior\ to\ t}{number\ of\ times\ a\ taken\ prior\ to\ t}$
  - With probability 1- ε, Select the action with the maximum value.
    - $A_t = \underset{a}{\operatorname{argmax}}\, Q_t(a)$
  - With probability ε, select an action randomly from all the actions with equal probability.

Experiments:

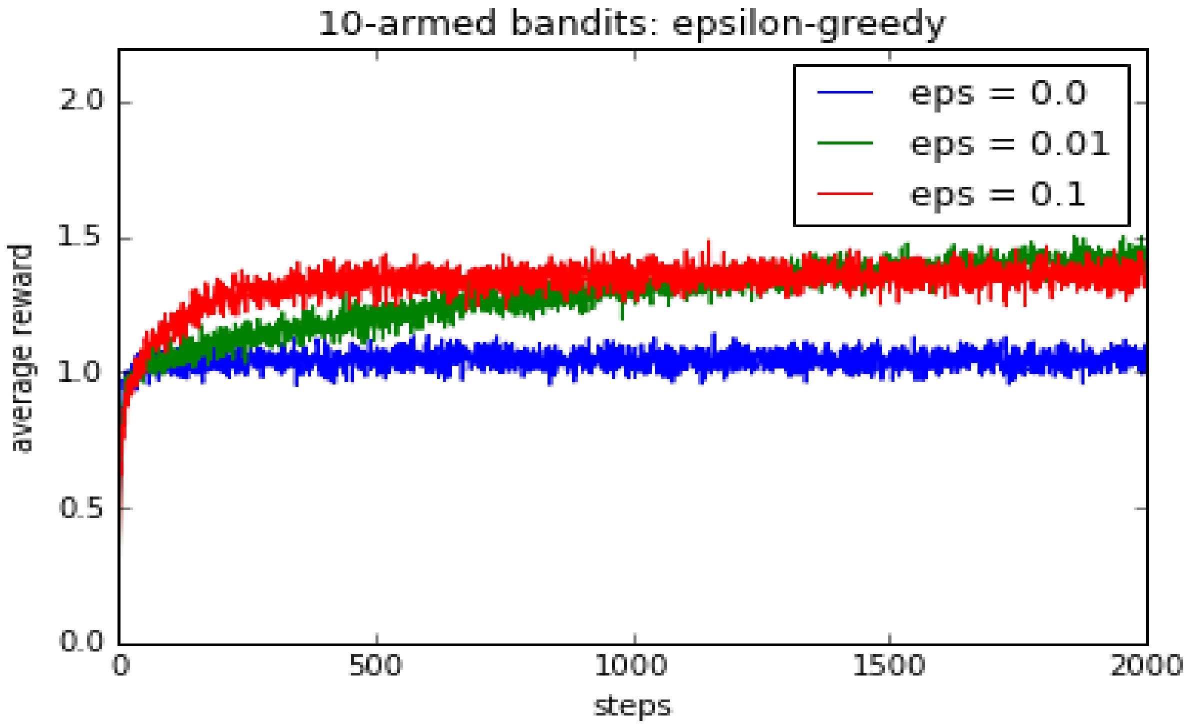
- Set up: (one run)
  - 10-armed bandits
  - Draw u_i from Gaussian(0,1), $i = 1,\ldots,10$
    - the expectation/mean of rewards for action i
  - Rewards of action i at time t: $x_i(t)$
    - $x_i(t) \sim \text{Gaussian}(u_i, 1)$
  - Play 2000 rounds/steps
  - Average return at each time step
- Average over 1000 runs

Experimental results: average over 1000 runs



Observations?

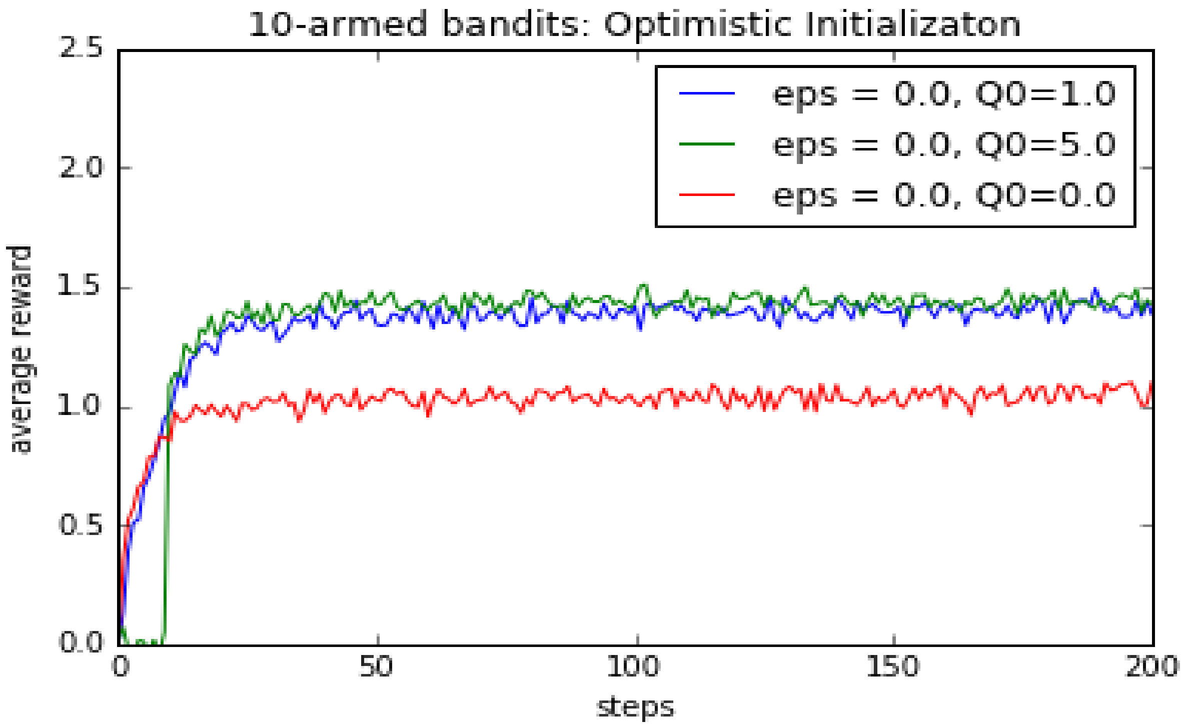
What will happen if we run more steps?



Observations:

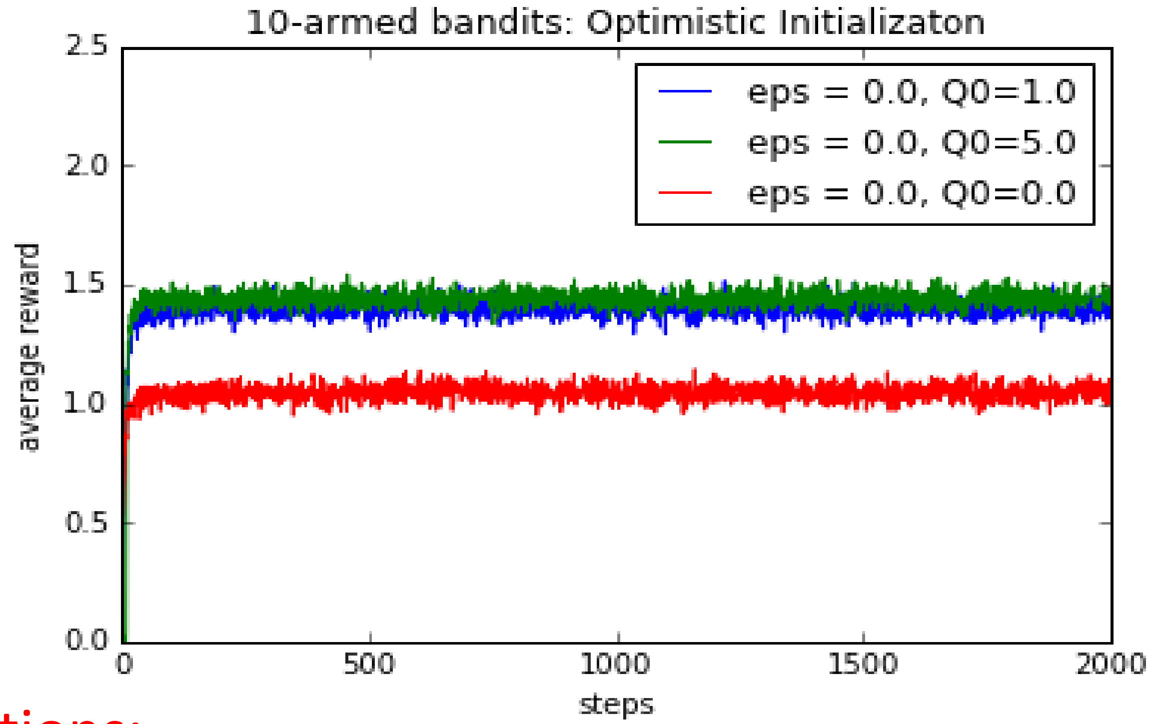
- Greedy method improved faster at the very beginning, but level off at a lower level.
- ε- Greedy methods continue to Explore and eventually perform better.
- The $\varepsilon = 0.01$ method improves slowly, but eventually performs better than the $\varepsilon = 0.1$ method.

Improve the Greedy method with optimistic initialization



Observations:

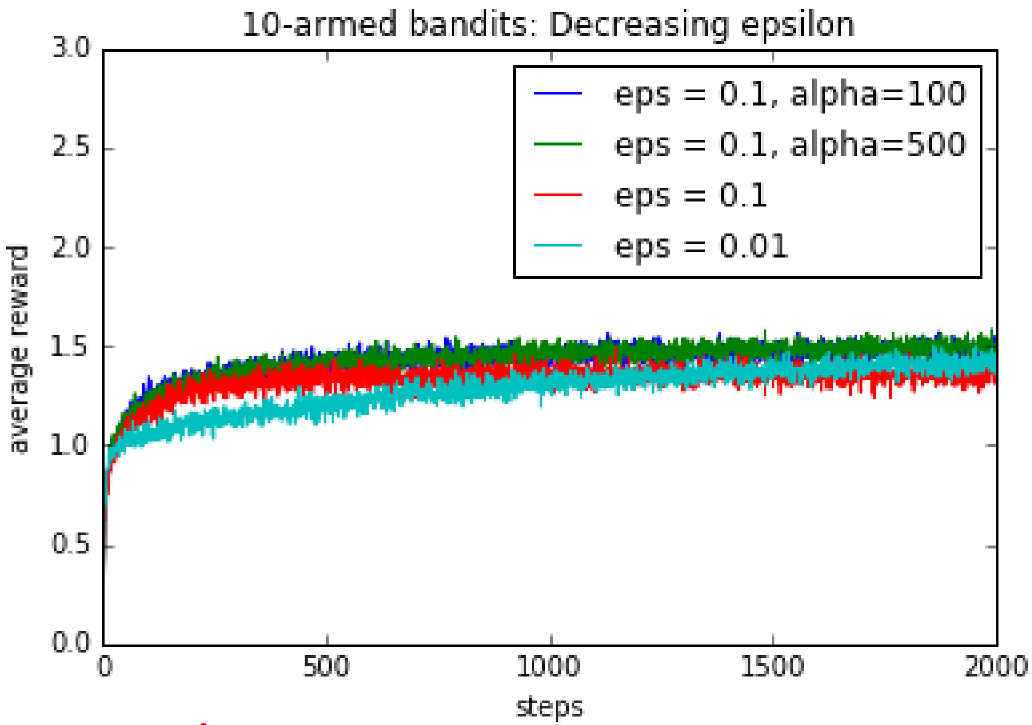
Greedy with optimistic initialization



Observations:

- Big initial Q values force the Greedy method to explore more in the beginning.
- No exploration afterwards.

Improve ε-greedy with decreasing ε over time



Decreasing over time:

- $\varepsilon_t = \varepsilon_t * (\alpha)/(t+\alpha)$
- Improves faster in the beginning, also outperforms fixed ε-greedy methods in the long run.

Weaknesses of ε-Greedy methods:

- ε-Greedy methods:
  - At time step t, estimate a value $Q_t(a)$ for each action
  - With probability 1- ε, Select the action with the maximum value.
    - $A_t = \underset{a}{\operatorname{argmax}} Q_t(a)$
  - With probability ε, select an action randomly from all the actions with equal probability.
- Weaknesses:

Weaknesses of ε-Greedy methods:

- ε-Greedy methods:
  - At time step t, estimate a value $Q_t(a)$ for each action
  - With probability 1- ε, Select the action with the maximum value.
    - $A_t = \underset{a}{\operatorname{argmax}} \, Q_t(a)$
  - With probability ε, select an action randomly from all the actions with equal probability.
- Weaknesses:
  - Randomly selects a action to explore, does not explore more "promising" actions.
  - Does not consider confidence interval. If an action has been taken many times, no need to explore it.

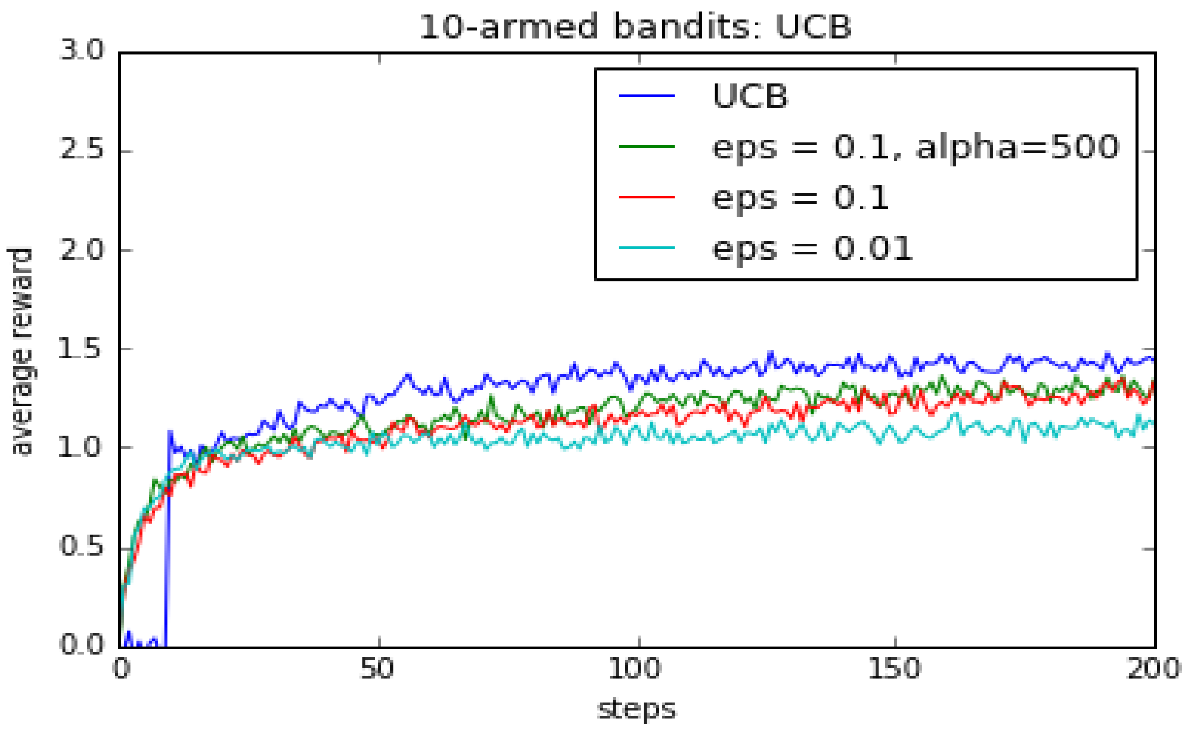
Upper Confidence Bound algorithm

- Take each action once.
- At any time t > K,
  - $A_t = \underset{a}{\operatorname{argmax}} \left(Q_t(a) + \sqrt{\frac{2 ln t}{N_a(t)}}\right)$, where
    - $Q_t(a)$ is the average reward obtained from action a,
    - $N_a(t)$ is the number of times action a has been taken.

Upper Confidence Bounds

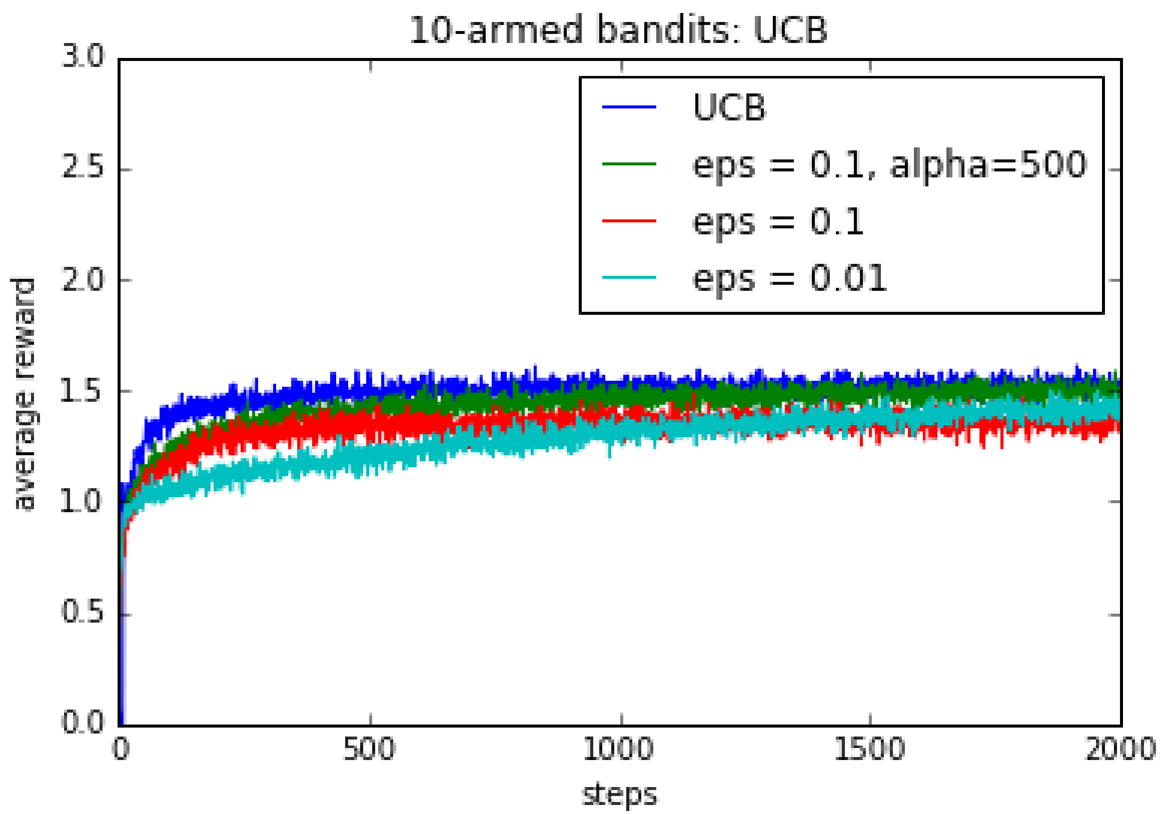
- $A_t = \underset{a}{\operatorname{argmax}} \left(Q_t(a) + \sqrt{\frac{2lnt}{N_a(t)}}\right)$
- $\sqrt{\frac{2lnt}{N_a(t)}}$ is confidence interval for the average reward,
- Small $N_a(t)$-> large confidence interval($Q_t(a)$ is more uncertain)
- large $N_a(t)$-> small confidence interval($Q_t(a)$ is more accurate)
- select action maximizing upper confidence bound.
  - Explore actions which are more uncertain, exploit actions with high average rewards obtained.
  - UCB: balance exploration and exploitation.
  - As t -> infinity, select optimal action.

How good is UCB? Experimental results



Observations:

How good is UCB? Experimental results



Observations: After initial 10 steps, improves quickly and outperforms ε-Greedy methods.

How good is UCB? Theoretical guarantees

- Expected regret is bounded by

$$8{*}\sum_{u_i < \text{u*}} \frac{lnt}{\text{u*-}\ u_i} + \left(1 + \frac{\pi^2}{3}\right) \sum_{i=1}^{K} (\text{u*-}\ u)$$

- Achieves the optimal regret up to a multiplicative constant.
- Expected regret grows in O(lnt)

How is the UCB derived?

- Hoeffding's Inequality Theorem:
- Let be X_i, …, X_t i.i.d. random variables in [0,1], and let $\overline{X}_t$ $be\ the\ sample\ mean, then$
  - $P[E[\overline{X}_t] \geq \overline{X}_t + u] \leq e^{-2tu^2}$
- Apply to the bandit rewards,
  - $P[Q(a) \geq Q_t(a) + V_t(a)] \leq e^{-2N_t(a)V_t(a)^2}$,
  - $2*V_t(a)$ is size of the confidence interval for Q(a)
  - $p = e^{-2N_t(a)Vt(a)2}$ is the probability of making an error

How is the UCB derived?

- Apply to the bandit rewards,
  - $P[Q(a) > Q_t(a)+V_t(a)] \leq e^{-2N_t(a)V_t(a)^2}$
- Let $p = e^{-2N_t(a)V_t(a)^2} = t^{-4}$,
- Then $V_t(a) = \sqrt{\frac{2lnt}{N_a(t)}}$
- Variations of UCB1(the basic one here):
  - UCB1_tuned, UCB3, UCBv, etc.
  - Provide and prove different upper bounds for the expected regret.
  - Add a parameter to control the confidence level

Multi-armed bandit problem

- Stochastic bandits
  - Problem formulation
  - Algorithms:
    - Greedy, ε- greedy methods, UCB
    - Boltzmann exploration(Softmax)
- Adversarial bandits
  - Problem formulation
  - Algorithms
- Contextual bandits

Multi-armed bandit problem

- Boltzmann exploration(Softmax)
  - Pick a action with a probability that is proportional to its average reward.
  - Actions with greater average rewards are picked with higher probability.
- The algorithm:
  - Given initial empirical means $u_1(0),\ldots,u_K(0)$
  - $p_i(t+1) = \dfrac{e^{u_i(t)/\tau}}{\sum_{j=1}^{K} e^{u_j(t)/\tau}}$, i =1,…,K
  - *What is the use of* τ ?
  - What happens as $\tau \rightarrow infinity$?

Multi-armed bandit problem

- Boltzmann exploration(Softmax)
  - Pick a action with a probability that is proportional to its average reward.
  - Actions with greater average rewards are picked with higher probability.
- The algorithm:
  - Given initial empirical means $u_1(0),\ldots,u_K(0)$
  - $p_i(t+1) = \dfrac{e^{u_i(t)/\tau}}{\sum_{j=1}^{K} e^{u_j(t)/\tau}}$, $i = 1,\ldots,K$
  - τ controls the choice.
  - $as\ \tau \rightarrow infinity, selects\ uniformly.$

Multi-armed bandit problem

- Stochastic bandits
  - Problem formulation
  - Algorithms:
    - ε- greedy methods, UCB
    - Boltzmann exploration(Softmax)
- Adversarial bandits
  - Problem formulation
  - Algorithms
- Contextual bandits

Non-stochastic/adversarial multi-armed bandit problem

- Setting:
  - K possible arms/actions: $1 \leq i \leq K$,
  - each action is denoted by an assignment of rewards. $x(1)$, $x(2)$, …, of vectors $x(t)=(x_1(t), x_2(t),\ldots,x_K(t))$
  - $x_i(t) \in [0,1]$: reward received if action i is chosen at time step t (can generalize to in the range[a, b])
  - Goal : maximize the return(the accumulative reward.)
- Example. K=3, an adversary controls the reward.

k	t=1	t=2	t=3	t=4	…
1	0.2	0.7	0.4	0.3	
2	0.5	0.1	0.6	0.2	
3	0.8	0.4	0.5	0.9	

Adversarial multi-armed bandit problem

- Weak regret = $G_{max}(T)-G_A(T)$
- $G_{max}(T)=\max_j \sum_{t=1}^{T} x_j(t)$, return of the single global best action at time horizon T
- $G_A(T)=\sum_{t=1}^{T} x_{i_t}(t)$, return at time horizon T of algorithm A choosing actions i_1, i_2,..
- Example. $G_{max}(4)=?$, $G_A(4)=?$
  - (*indicates the reward received by algorithm A at a time step t)

K/T	t=1	t=2	t=3	t=4	...
1	0.2	0.7	0.4	0.3*	
2	0.5*	0.1	0.6*	0.2	
3	0.8	0.4*	0.5	0.9	

Adversarial multi-armed bandit problem

- Example. $G_{max}(4)=?$, $G_A(4)=?$
  - $G_{max}(4)= \max_j \sum_{t=1}^{T} x_j(t) = \max\{G_1(4), G_2(4), G_3(4)\}$

    $=\max\{1.6, 1.4, 2.4\}=2.4$
  - $G_A(4))=\sum_{t=1}^{T} x_{i_t}(t)=0.5+0.4+0.6+0.3=1.8$
- Evaluate a randomized algorithm A: Bound on the expected regret for a A: $G_{max}(T)-E[G_A(T)]$

K/T	t=1	t=2	t=3	t=4	…
1	0.2	0.7	0.4	0.3*	
2	0.5*	0.1	0.6*	0.2	
3	0.8	0.4*	0.5	0.9	

Exp3: exponential-weight algorithm for exploration and exploitation

- Parameter: $\gamma \in (0,1]$
- Initialize:
  - $w_i(1) = 1$ for $i = 1, .., K$
- for $t=1, 2, \ldots$
  - Set $p_i(t) = (1-\gamma)\frac{w_i(t)}{\sum_{j=1}^{K} w_j(t)} + \gamma . \frac{1}{K}$, $i = 1,\ldots,K$
  - Draw i_t randomly from $p_1(t), \ldots, p_K(t)$
  - Receive reward $x_{i_t} \in [0,1]$
  - For $j = 1,\ldots,K$
    - $\widehat{x_j(t)} = x_j(t)/p_j(t)$ if $j=i_t$
    - $\widehat{x_j(t)} = 0$ otherwise
    - $w_j(t+1) = w_j(t) * \exp(\gamma * \widehat{x_j(t)}/K)$

Question: what happens at time step 1? At time step 2?
What happens as γ --> 1? Why $x_j(t)/p_j(t)$?

Exp3: Balancing exploration and exploitation

- $p_i(t) = (1-\gamma)\frac{w_i(t)}{\sum_{j=1}^{K} w_j(t)} + \gamma . \frac{1}{K}$, i = 1,…,K

- Balancing exploration and exploitation in Exp3
  - The distribution P(t) is a mixture of the uniform distribution and a distribution which assigns to each action a probability mass exponential in the estimated cumulative reward for that action.
  - Uniform distribution encourages exploration
  - The other probability encourages exploitation
  - The parameter γ controls the exploration.
  - $x_j(t) / p_j(t)$ compensate the reward of actions that are unlikely to be chosen

Exp3: How good is it?

- Theorem:

  For any K>0 and for any $\gamma \in (0,1]$,

  $G_{max} - E[G_{Exp3}] \leq (e-1) * \gamma * G_{max} + K * \ln(K)/\gamma$

  Holds for any assignment of rewards and for any T>0
- Corollary:
  - For any T > 0 , assume that $g \geq G_{max}$ and that algorithm Exp3 is run with the parameter
  - $\gamma = \min \{1, \sqrt{\frac{K \ln K}{(e-1)g}}\}$,
  - Then $G_{max} - E[G_{Exp3}] \leq 2\sqrt{(e-1)}\sqrt{gK\ln k} \leq 2.63\sqrt{gK\ln k}$

A family of Exp3 algorithms

- Exp3, Exp3.1, Exp3.P, Exp4, …
- Better performance
  - tighter upper bound,
  - achieve small weak regret with high probability, etc.
- Changes based on Exp3
  - How to choose γ
  - How to initialize and update the weights w

Summary: Exploration & exploitation in Reinforcement learning

- Stochastic bandits
  - Problem formulation
  - Algorithms
    - ε- greedy methods, UCB, Boltzmann exploration(Softmax)
- Adversarial bandits
  - Problem formulation

u*- u

  - Algorithms: Exp3
- Contextual bandits
  - At each time step t, observes a feature/context vector.
  - Uses both context vector and rewards in the past to make a decision
  - Learns how context vector and rewards relate to each other.

References:

- “Reinforcement Learning: An Introduction” by Sutton and Barto
  - https://webdocs.cs.ualberta.ca/~sutton/book/the-book-2nd.html
- “The non-stochastic multi-armed bandit problem” by Auer, Cesa-Bianchi, Freund, and Schapire
  - https://cseweb.ucsd.edu/~yfreund/papers/bandits.pdf
- “Multi-armed bandits” by Michael
  - http://blog.thedataincubator.com/2016/07/multi-armed-bandits-2/
- …

Take home questions?

- 1. What is the difference between MABs(multi-armed bandits) and MDPs(Markov Decision Processes)?
- 2. How are they related?