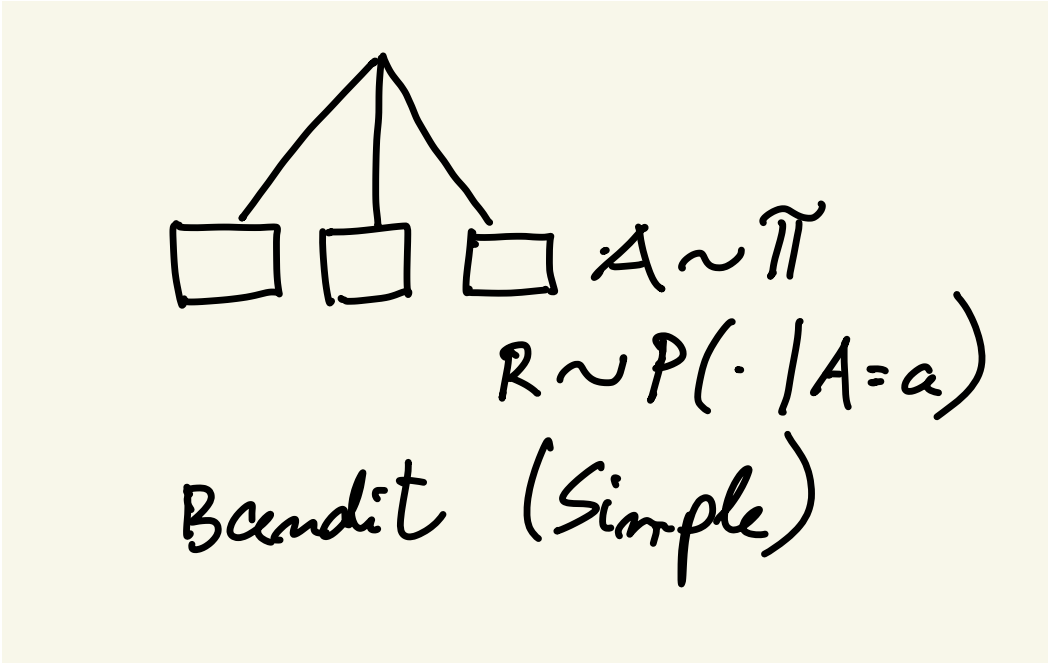
Multi-arm Bandits Part 2

Sutton and Barto, Chapter 2

The simplest reinforcement learning problem

Recall: Multi-armed bandits

- No x, take an action, observe a reward immediately
- So, a degenerate tree (not truly sequential)
- This is what we call a simple (multi-arm) bandit problem
- Focus on exploration, not credit assignment



Recall: *k*-armed Bandit Problem

- On each of an infinite sequence of *time steps*, $t=1, 2, 3, \ldots$, you choose an action A_t from k possibilities, and receive a real-valued *reward* R_t

- The reward depends only on the action taken; it is identically, independently distributed (i.i.d.):

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a], \quad \forall a \in \{1, \ldots, k\}$$ *true values*

- These true values are *unknown*. The distribution is unknown

- Nevertheless, you must maximize your total reward

- You must both try actions to learn their values (explore), and prefer those that appear best (exploit)

Recall: Action-Value Methods

- Methods that learn action-value estimates and construct a policy based on them
- Estimates can be maintained incrementally, eg:

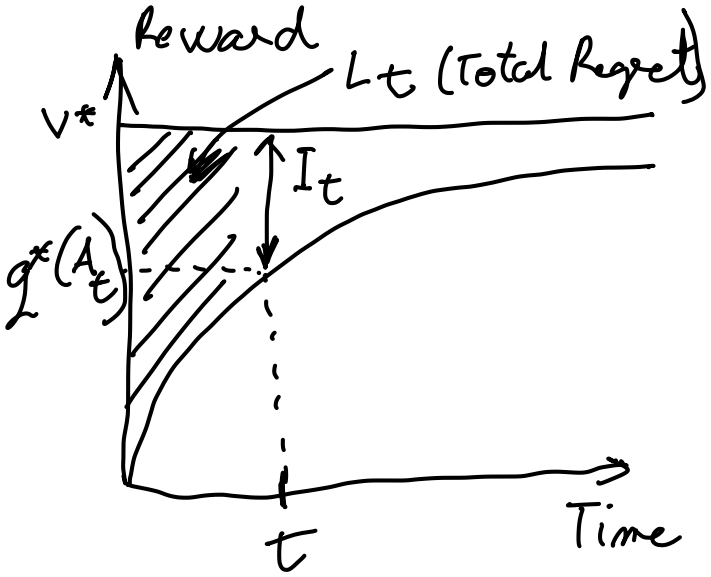
$$Q_{n+1} = Q_n + \frac{1}{n}\Big[R_n - Q_n\Big]$$

- ϵ-greedy: choose the action with maximum Q_t with high probability, uniformly randomly otherwise
- UCB: maintain an upper bound on the action value, choose greedily based on value plus upper bound

$$A_t \doteq \arg\max_a \left[Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$$

Formally: What do bandit algorithms optimize?

- The best possible action: $a^* = \arg\max_a q^*(a)$
- The value of the best possible action: $v^* = q^*(a^*)$
- Regret at time step t: $I_t = \mathbb{E}[v^* - q^*(A_t)]$
- Total regret up to time t: $L_t = \mathbb{E}\left[\sum_{\tau=0}^{t} I_t\right]$



Counting regret

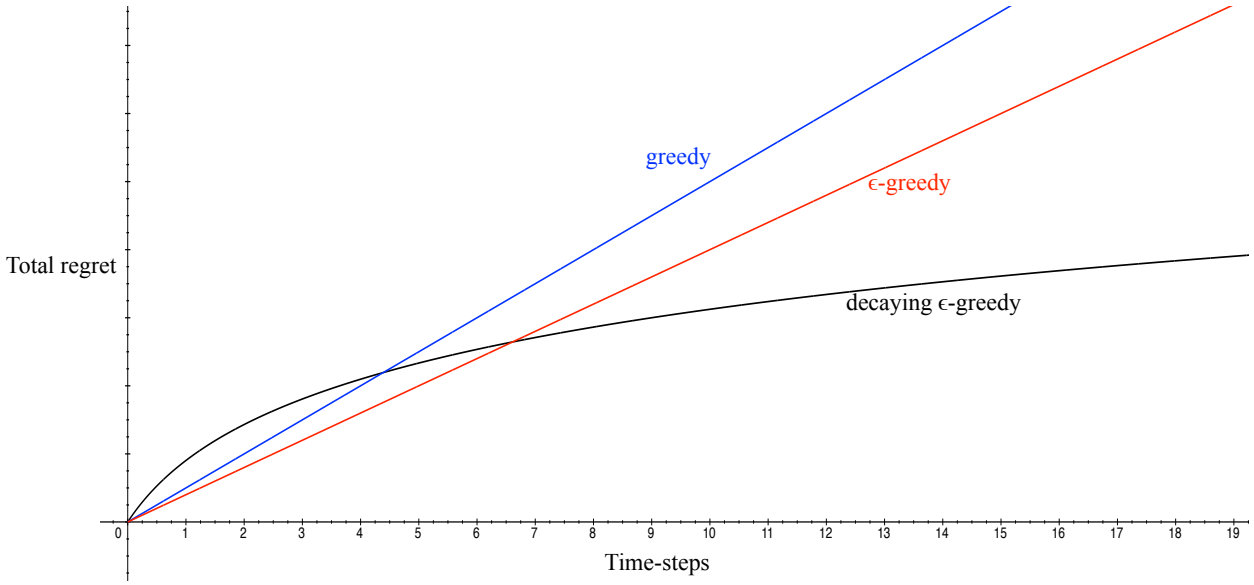
- The expected number of times action *a* has been chosen up to time t: $N_t(a)$
- The gap of action *a*: $\Delta_a = v^* - q^*(a)$
- Note that the optimal action(s) has gap 0
- Regret can then be computed from gaps and counts!

$$
\begin{aligned}
L_t &= \mathbb{E}\left[v^* - q^*(A_t)\right] \\
&= \sum_{a \in \mathcal{A}} \mathbb{E}\left[N_t(a)\right](v^* - q^*(a)) \\
&= \sum_{a \in \mathcal{A}} \mathbb{E}\left[N_t(a)\right] \Delta_a
\end{aligned}
$$

Observations

- Regret is a useful theoretical tool for comparing bandit/RL algorithms
- You can't compute it empirically (except in toy problems)
- But we can *bound it over all problems*
- Maximizing reward is equivalent to minimizing regret
- Worse actions lead to more regret
- Ideally, we minimize the number of time steps on which high regret actions are chosen

Linear vs sublinear regret



- If an algorithm forever explores it will have linear total regret
- If an algorithm never explores it will have linear total regret
- Is it possible to achieve sublinear total regret?

Epsilon-greedy regret

- With probability $(1 - \epsilon)$ select greedy action $A_t = \arg\max_a Q_t(a)$
- With probability ϵ select uniformly at random
- Selecting action a incurs regret Δ_a
- Therefore, the probability of choosing any action at time step t is *at least*: $\frac{\epsilon}{|\mathcal{A}|}$
- So instantaneous regret is bounded as: $\mathbb{E}[I_t] \geq \frac{\epsilon}{|\mathcal{A}|} \sum_a \Delta_a$
- And total regret: $L_t = \sum_{\tau=1}^{t} \mathbb{E}[I_\tau] \geq t \frac{\epsilon}{|\mathcal{A}|} \sum_a \Delta_a$

Improving on linear regret

- Fixed ϵ leads to linear regret !
- What if we reduced the frequency of suboptimal actions over time?
- Ie introduce a decay: $\epsilon_t \to 0$ as $t \to \infty$
- Let $g = \min_{a:\Delta_a>0} \Delta_a$ be the gap of the second-best action
- Let $\epsilon_t = \min\left(1, \frac{c\,|\mathcal{A}|}{g^2 t}\right)$ where $c > 0$ is a constant
- We can show that this algorithm has logarithmic regret!

What is the optimal achievable regret?

- The difficulty of a bandit problem depends on how similar the optimal arm is to all the rest
- The closer the means and the more similar the reward distribution, the harder the problem
- Distribution similarity can be described by the KL divergence between the reward distribution of arm *a* compared to the optimal arm
- Lai and Robins (1979): for any multi-armed bandit asymptotic regret is at least logarithmic in the number of steps:

$$\lim_{t \to \infty} L_t \geq \log t \sum_{a:\Delta_a > 0} \frac{\Delta_a}{KL(\mathcal{R}_a || \mathcal{R}_{a*})} = O(\log t)$$

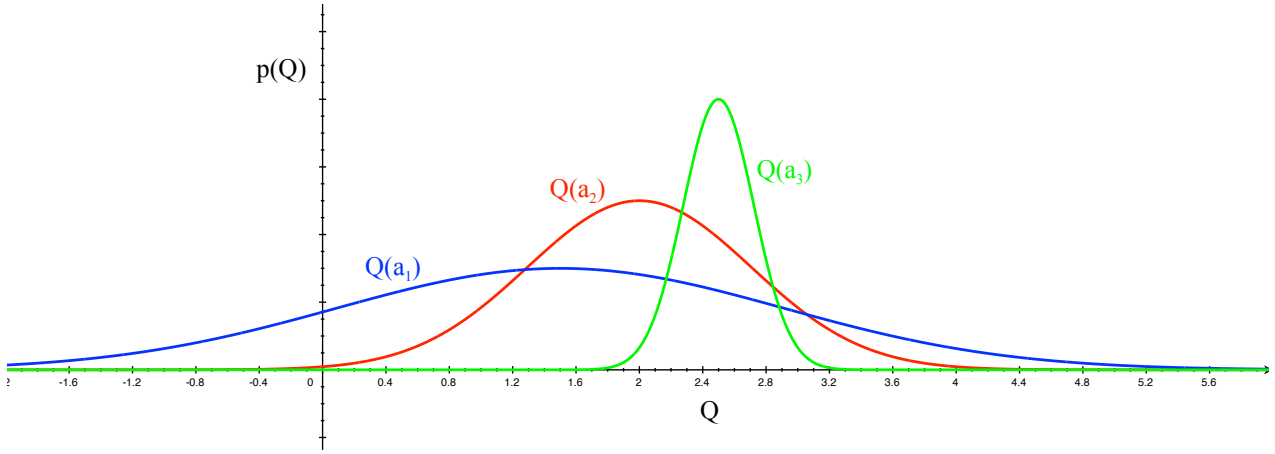
Achieving optimal regret

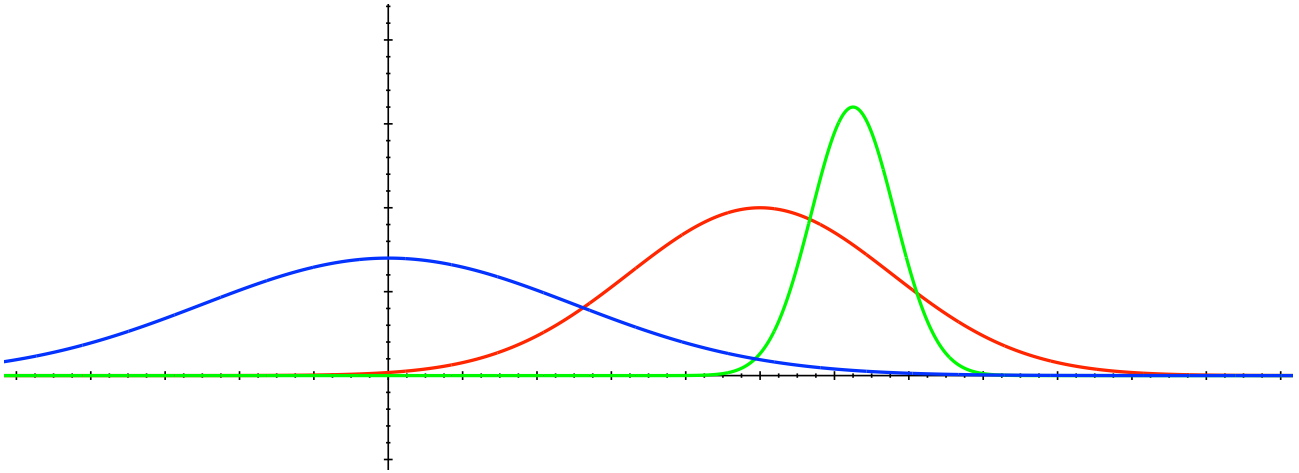
- Decaying epsilon can do this, but requires knowledge of the action gap (which is not known in practice)

- Are there other algorithms that achieve logarithmic asymptotic regret?

Recall: Optimism in the face of uncertainty

- Choose actions about which you are very uncertain





UCB

- Choose greedily wrt $A_t = \arg\max_a (Q_t(a) + U_t(a))$
- Where the upper bound: $U_t = c\sqrt{\frac{log(t)}{N_t(a)}}$
- Why did we pick U this way?

Hoeffding Inequality

Theorem (Hoeffding's Inequality)

Let $X_1, ..., X_t$ be i.i.d. random variables in [0,1], and let $\overline{X}_t = \frac{1}{\tau} \sum_{\tau=1}^{t} X_\tau$ be the sample mean. Then

$$\mathbb{P}\left[\mathbb{E}[X] > \overline{X}_t + u\right] \leq e^{-2tu^2}$$

From Hoeffding to UCB

- Apply Hoeffding to a bandit problem for action a:
  $P\left[q^*(a) > Q_t(a) + U_t(a)\right] \leq e^{-2N_t(a)U_t(a)^2}$

- Pick a probability p that the true value exceeds the upper bound
  $$e^{-2N_t(a)U_t(a)^2} = p$$
  and solve for U:
  $$U_t(a) = \sqrt{\frac{-\log p}{2N_t(a)}}$$

- Now reduce p as we observe more rewards, eg $p = t^{-m}$

- For m=8 you get the classic version of UCB!

- Regret: $\lim_{t\to\infty} L_t \leq 8\log(t) \sum_{a:\Delta_a>0} \Delta_a$

Summary so far

- Logarithmic regret is the best we can hope for in K-armed bandits
- In a stationary problem, epsilon-greedy with fixed epsilon provides linear regret (worst possible)
- Decaying epsilon can provide logarithmic regret only if you know the optimality gap (how much is the difference in expected reward between the top 2 actions)
- UCB is regret-optimal!
- Next: gradient-based algorithms

Gradient-Bandit Algorithms

- Let $H_t(a)$ be a learned *preference* for taking action a

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

Note that this allows us to work with unnormalized preferences and turn them into probabilities!

Same idea as using potentials in graphical models

Softmax (Boltzmann) Exploration

- Let $H_t(a)$ be a learned *preference* for taking action a

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

Consider $H_t(a) = Q_t(a)/T$

This is Boltzmann or softmax exploration!

If the temperature T is very large (towards infinity) - same as uniform

If temperature T goes to 0, same as greedy

Very popular method in practice due to simplicity

But can we derive how preferences should be updated?

Gradient-Bandit Algorithms

- Let $H_t(a)$ be a learned *preference* for taking action a

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

$$H_{t+1}(A_t) \doteq H_t(A_t) + \alpha\big(R_t - \bar{R}_t\big)\big(1 - \pi_t(A_t)\big)$$

$$\bar{R}_t \doteq \frac{1}{t}\sum_{i=1}^{t} R_i$$

Gradient-Bandit Algorithms

- Let $H_t(a)$ be a learned *preference* for taking action a

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

$$H_{t+1}(a) \doteq H_t(a) + \alpha\big(R_t - \bar{R}_t\big)\big(\mathbf{1}_{a=A_t} - \pi_t(a)\big), \qquad \forall a,$$

$$\bar{R}_t \doteq \frac{1}{t}\sum_{i=1}^{t} R_i$$

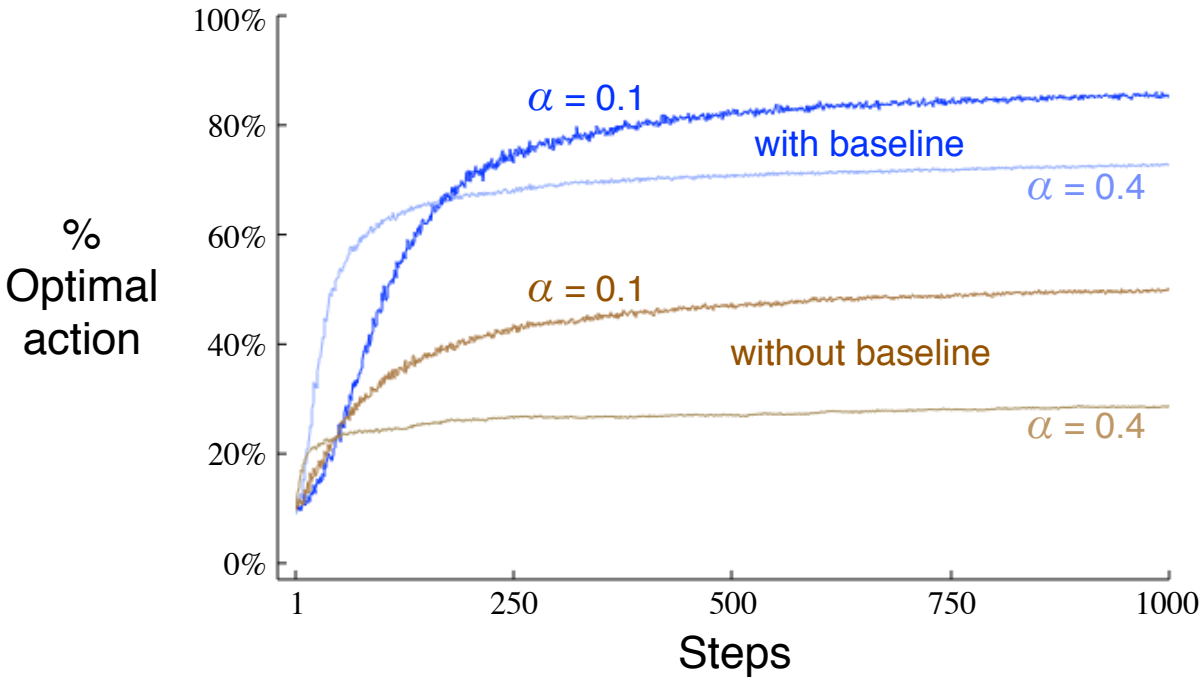
Gradient-Bandit Algorithms

- Let $H_t(a)$ be a learned *preference* for taking action a

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

$$H_{t+1}(a) \doteq H_t(a) + \alpha\big(R_t - \bar{R}_t\big)\big(\mathbf{1}_{a=A_t} - \pi_t(a)\big), \qquad \forall a,$$

$$\bar{R}_t \doteq \frac{1}{t}\sum_{i=1}^{t} R_i$$



Derivation of gradient-bandit algorithm

In exact *gradient ascent*:

$$H_{t+1}(a) \doteq H_t(a) + \alpha \frac{\partial\, \mathbb{E}\,[R_t]}{\partial H_t(a)}, \tag{1}$$

where:

$$\mathbb{E}[R_t] \doteq \sum_b \pi_t(b) q_*(b),$$

$$\begin{aligned}
\frac{\partial\, \mathbb{E}[R_t]}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)} \left[\sum_b \pi_t(b) q_*(b) \right] \\
&= \sum_b q_*(b) \frac{\partial\, \pi_t(b)}{\partial H_t(a)} \\
&= \sum_b \left(q_*(b) - X_t \right) \frac{\partial\, \pi_t(b)}{\partial H_t(a)},
\end{aligned}$$

where X_t does not depend on b, because $\sum_b \frac{\partial\, \pi_t(b)}{\partial H_t(a)} = 0$.

$$
\begin{aligned}
\frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)} &= \sum_b \big(q_*(b) - X_t\big) \frac{\partial \pi_t(b)}{\partial H_t(a)} \\
&= \sum_b \pi_t(b) \big(q_*(b) - X_t\big) \frac{\partial \pi_t(b)}{\partial H_t(a)} / \pi_t(b) \\
&= \mathbb{E}\left[\big(q_*(A_t) - X_t\big) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} / \pi_t(A_t) \right] \\
&= \mathbb{E}\left[\big(R_t - \bar{R}_t\big) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} / \pi_t(A_t) \right],
\end{aligned}
$$

where here we have chosen $X_t = \bar{R}_t$ and substituted R_t for $q_*(A_t)$, which is permitted because $\mathbb{E}[R_t | A_t] = q_*(A_t)$.

For now assume: $\frac{\partial \pi_t(b)}{\partial H_t(a)} = \pi_t(b)\big(\mathbf{1}_{a=b} - \pi_t(a)\big)$. Then:

$$
\begin{aligned}
&= \mathbb{E}\big[\big(R_t - \bar{R}_t\big) \pi_t(A_t) \big(\mathbf{1}_{a=A_t} - \pi_t(a)\big) / \pi_t(A_t) \big] \\
&= \mathbb{E}\big[\big(R_t - \bar{R}_t\big) \big(\mathbf{1}_{a=A_t} - \pi_t(a)\big) \big].
\end{aligned}
$$

$$
H_{t+1}(a) = H_t(a) + \alpha\big(R_t - \bar{R}_t\big)\big(\mathbf{1}_{a=A_t} - \pi_t(a)\big), \text{ (from (1), QED)}
$$

Thus it remains only to show that

$$\frac{\partial \pi_t(b)}{\partial H_t(a)} = \pi_t(b)\big(\mathbf{1}_{a=b} - \pi_t(a)\big).$$

Recall the standard quotient rule for derivatives:

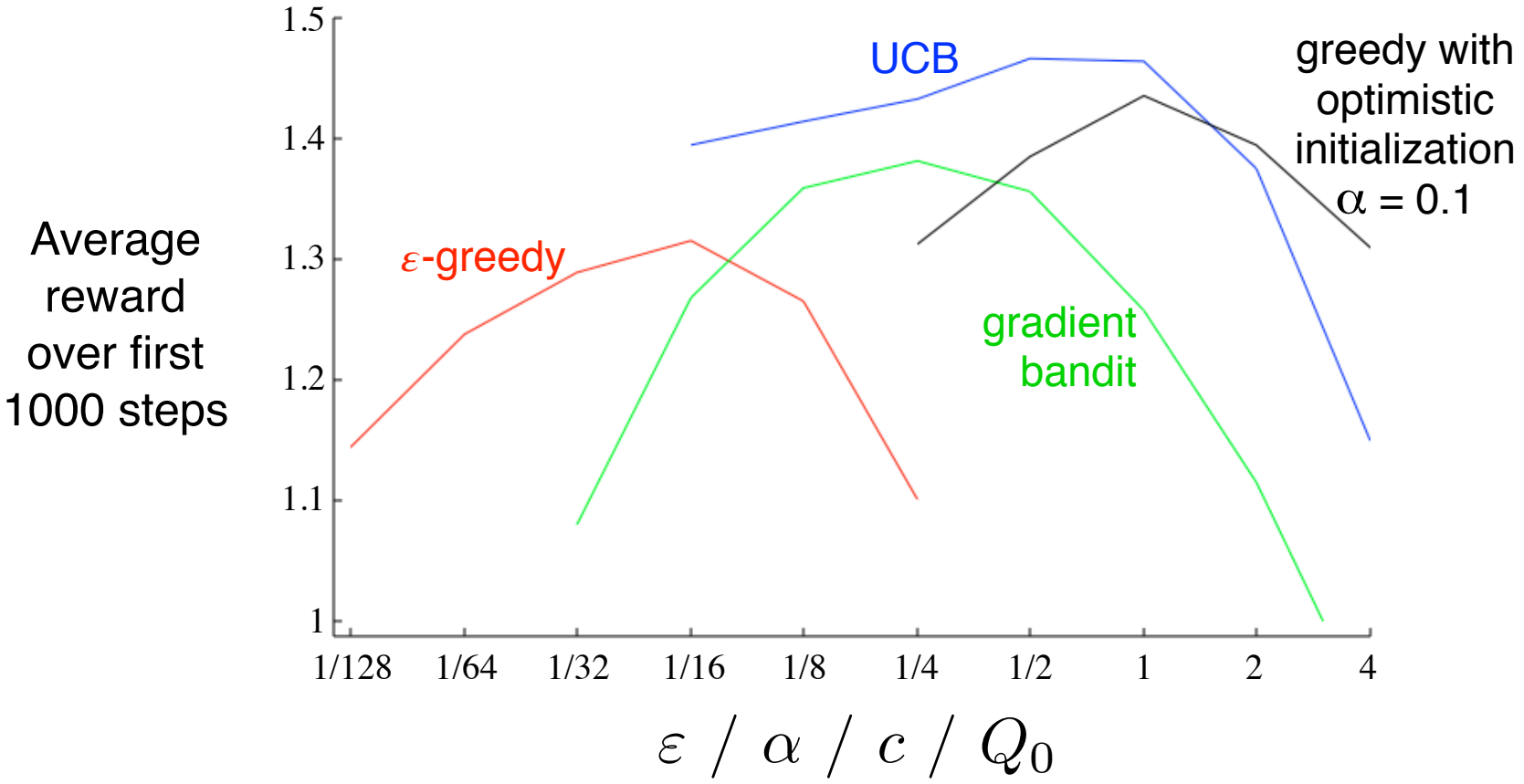
$$\frac{\partial}{\partial x}\left[\frac{f(x)}{g(x)}\right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x)\frac{\partial g(x)}{\partial x}}{g(x)^2}.$$

Using this, we can write...

Quotient Rule: $\frac{\partial}{\partial x}\left[\frac{f(x)}{g(x)}\right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x)\frac{\partial g(x)}{\partial x}}{g(x)^2}$

$$
\begin{aligned}
\frac{\partial\, \pi_t(b)}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)}\, \pi_t(b) \\
&= \frac{\partial}{\partial H_t(a)} \left[\frac{e^{H_t(b)}}{\sum_{c=1}^{k} e^{H_t(c)}} \right] \\
&= \frac{\frac{\partial e^{H_t(b)}}{\partial H_t(a)} \sum_{c=1}^{k} e^{H_t(c)} - e^{H_t(b)} \frac{\partial \sum_{c=1}^{k} e^{H_t(c)}}{\partial H_t(a)}}{\left(\sum_{c=1}^{k} e^{H_t(c)}\right)^2} && \text{(Q.R.)} \\
&= \frac{\mathbf{1}_{a=b} e^{H_t(a)} \sum_{c=1}^{k} e^{H_t(c)} - e^{H_t(b)} e^{H_t(a)}}{\left(\sum_{c=1}^{k} e^{H_t(c)}\right)^2} && \left(\tfrac{\partial e^x}{\partial x} = e^x\right) \\
&= \frac{\mathbf{1}_{a=b} e^{H_t(b)}}{\sum_{c=1}^{k} e^{H_t(c)}} - \frac{e^{H_t(b)} e^{H_t(a)}}{\left(\sum_{c=1}^{k} e^{H_t(c)}\right)^2} \\
&= \mathbf{1}_{a=b} \pi_t(b) - \pi_t(b)\pi_t(a) \\
&= \pi_t(b)\big(\mathbf{1}_{a=b} - \pi_t(a)\big). && \text{(Q.E.D.)}
\end{aligned}
$$

Summary Comparison of Bandit Algorithms



Classes of bandit algorithms

- Epsilon-greedy (simple randomization)
- Optimism in the face of uncertainty: optimistic initialization, UCB
- Gradient-based policy optimization
- Softmax / Boltzmann exploration (similar in shape to gradient-based but relies on value function estimation)
- *One more class: probability matching*

Probability matching

- *Select action a according to the probability of it being optimal:* $\pi(A_t = a \,|\, H_t) = \mathbb{P}[q^*(a) \geq q^*(a') \forall a' \neq a \,|\, H_t]$ *where* $H_t = \langle A_1 R_1 \ldots A_{t-1}, R_{t-1} \rangle$

- Note that probability matching is optimistic in the face of uncertainty - because uncertain action typically have a higher probability of being considered optimal

- How can we implement this idea?

Intuition

- If we knew the problem (reward distribution for each arm) we could easily compute the optimal action
- Initially, we have *uncertainty about the problem*
- Let's model the uncertainty directly, using a probability distribution over the problem parameters!
- This is an instance of *Bayesian reasoning*

Detour Example: Coin Toss

- Suppose you flip a coin and observe numbers of heads and tails N_H, N_T
- Maximum likelihood estimation says the probability of heads is: $\frac{N_H}{N_H + N_T}$
- But if you knew the coin is probably biased? Could you incorporate this information somehow?

Detour Example: Bayesian Coin Toss

- Coin toss: x ~ Bernoulli(θ)
- Let's assume that
  - θ ~ Beta(α_H, α_T)
  - $P(\theta) \propto \theta^{\alpha_H - 1} (1 - \theta)^{\alpha_T - 1}$

Beta distribution

Prior

- $P(\theta|X) = \frac{P(X|\theta)P(\theta)}{\sum_\theta P(X|\theta)}$

Posterior

The prior is conjugate!

More generally: Bayesian Reasoning

- Assume the parameters you're interested in have some prior distribution $p_0(\theta)$
- After some dataset *D* comes in, compute a *posterior:* $P(\theta|D) \propto P(D|\theta)p_0(\theta)$
- Now you can sample from the posterior!
- Advantages:
  - provides a good uncertainty estimate for θ
  - can incorporate existing knowledge through the prior
  - Converges in the limit to the same answer as max likelihood estimation but can give better estimates when you have small samples
- Disadvantage: Expensive
- Usually practiced with conjugate priors (eg Beta for Bernoulli distributions, Normal for Normal distributions…)

Back to bandits: Thompson sampling

- Instantiation of probability matching / Bayesian reasoning for bandits (developed in the 1930s)
- Idea: we are interested in the parameters of the reward distribution for each arm $\mathscr{R}_a, \forall a$
- So maintain a probability distribution over them!
- Eg if the distributions are Bernoulli, maintain a Beta distribution, with some prior (maybe equal probability) and update as data comes in
- Eg if the distributions are normal, maintain a normal over the mean, or mean and standard deviation

Algorithm

- Start with a prior over the reward distributions $p_0(\mathscr{R}_a), \forall a$
- Repeat
    1. Sample a bandit problem, aka rewards from the distributions: $\mathscr{R}_t \sim p(\mathscr{R}_a, \forall a \,|\, H_t)$
    2. Compute the best action for problem $A_t = a^*(\mathscr{R}_t)$
    3. Note this can be done easily for many problems of interest!
    4. Pull arm A_t and observe reward R_t
    5. Update the history: $H_{t+1} = \langle H_t, A_t, R_t \rangle$ and posterior $p(\mathscr{R}_a, \forall a \,|\, H_{t+1})$

Efficient implementation

- Instead of maintaining the whole history, if we have conjugate priors, we can incrementally update the posterior
- This can in fact be done using an equivalent sample size trick: imagine you have some data sampled from the prior, which is added to your dataset
- For example:

Algorithm 1: Thompson Sampling for Bernoulli bandits

$S_i = 0, F_i = 0.$

foreach $t = 1, 2, \ldots,$ **do**

  For each arm $i = 1, \ldots, N$, sample $\theta_i(t)$ from the Beta$(S_i + 1, F_i + 1)$ distribution.

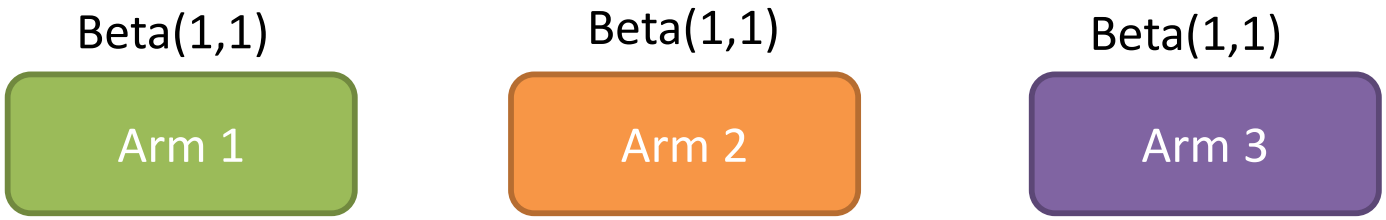
  Play arm $i(t) := \arg\max_i \theta_i(t)$ and observe reward r_t.

  If $r = 1$, then $S_i = S_i + 1$, else $F_i = F_i + 1$.

end

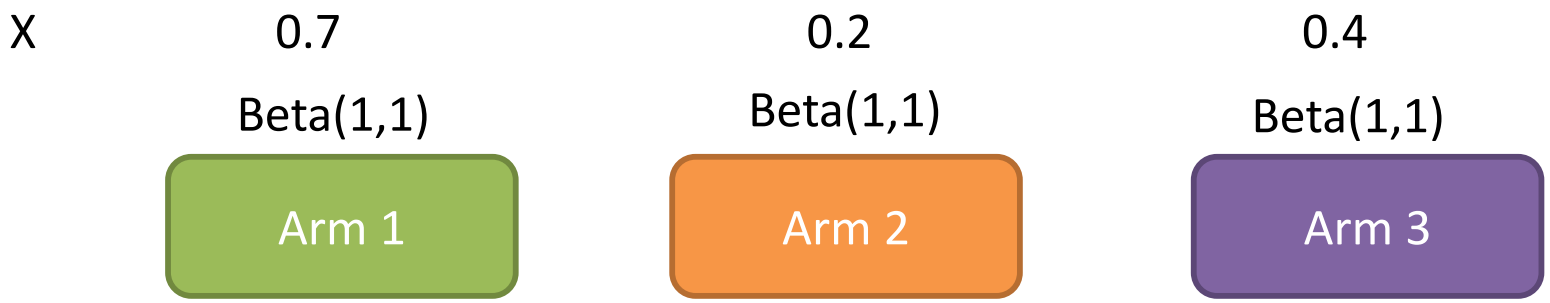
Example

- Start with a prior



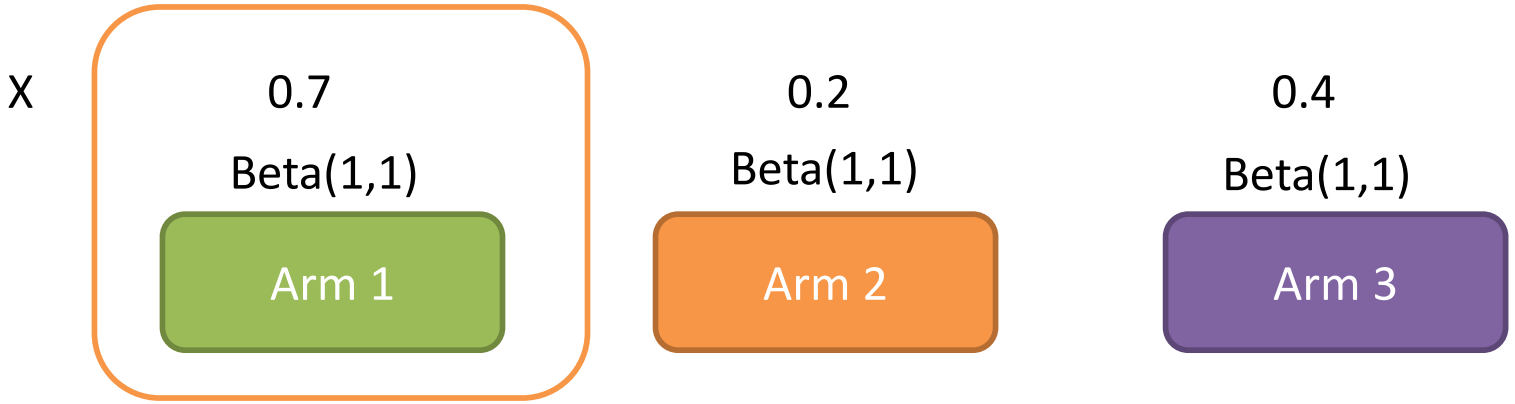
Example

- Sample a problem (bandit) from the prior



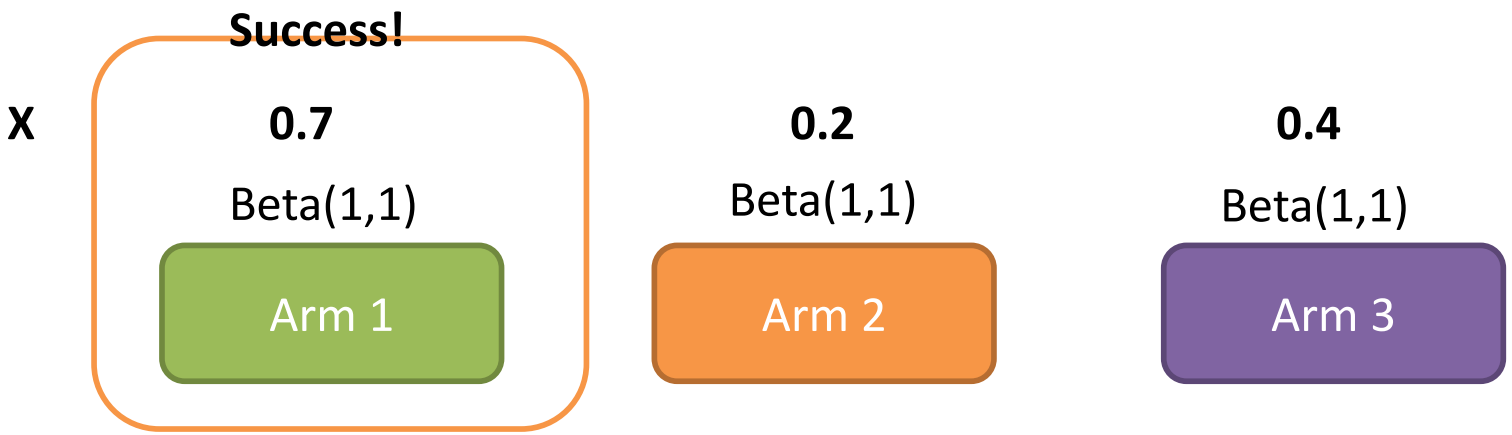
Example

- Compute the solution to the problem (best arm)



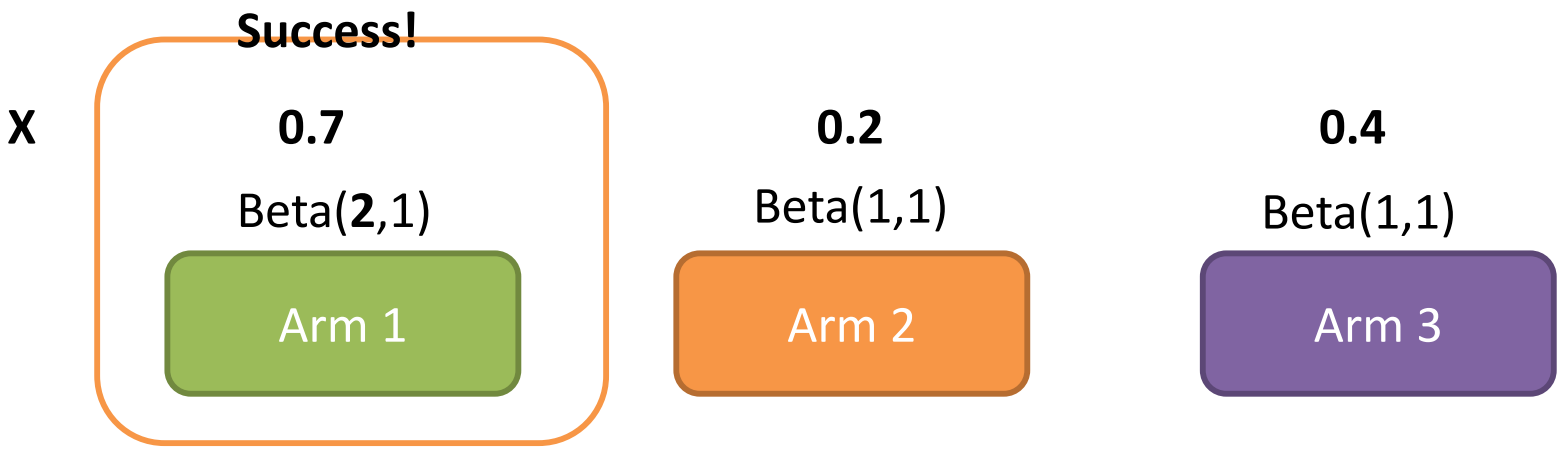
Example

- Execute the action in the real environment and observe its outcome (the reward)



Example

- Update the posterior to incorporate the observed data



Properties

- Like UCB, Thompson sampling is asymptotically optimal, ie. achieves $O(\log t)$ regret

- Took almost 80 years to prove that!! (https://arxiv.org/abs/1111.1797)

- Empirically, Thompson sampling works well for small sample sizes, especially if you know something about the problem

Problem space

	Single State	Associative
Instructive feedback		
Evaluative feedback		

Problem space

	Single State	Associative
Instructive feedback		
Evaluative feedback	Bandits (Function optimization)	

Problem space

	Single State	Associative
Instructive feedback		Supervised learning
Evaluative feedback	Bandits (Function optimization)	

Problem space

	Single State	Associative
Instructive feedback	Averaging (Imitiation)	Supervised learning
Evaluative feedback	Bandits (Function optimization)	

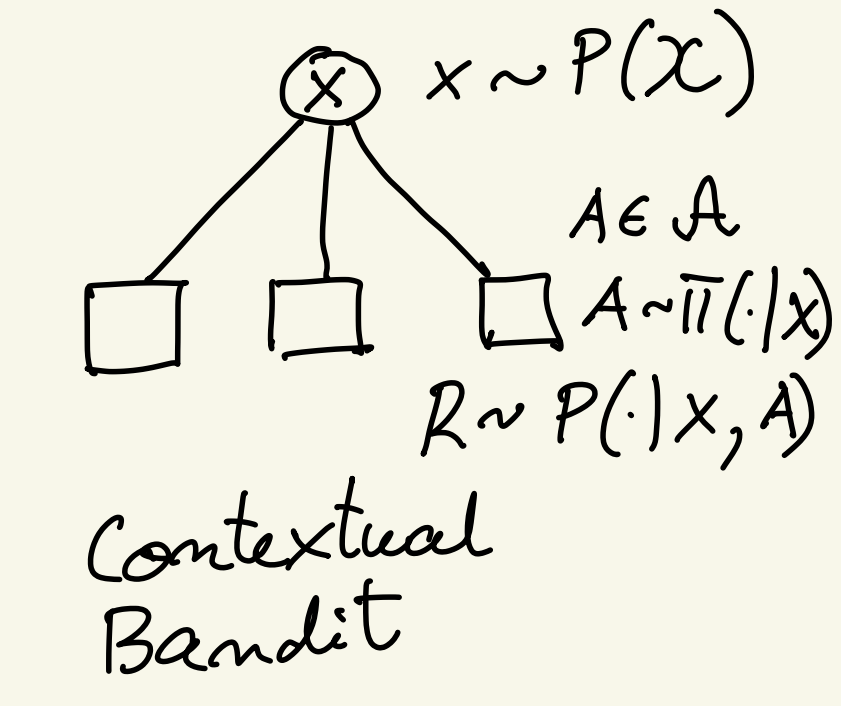
Problem space

	Single State	Associative
Instructive feedback	Averaging (Imitiation)	Supervised learning
Evaluative feedback	Bandits (Function optimization)	Contextual bandits

Contextual bandits

- We have some context, aka observation or state (discrete or continuous, often high-dimensional)

- The reward distribution depends on the context



Not just exploration!

- We have to *assign credit to different features of the context*!
- Usually we will use *function approximation* to estimate action-values $Q_w(x, a)$ (or to estimate the preference function, or policy)
- Algorithms we talked about all have equivalents in this problem!
- Eg epsilon-greedy, softmax
- Eg UCB -> LinUCB (assuming $Q_w(x, a) = w_a^T x$)
- Eg Thompson sampling assuming linear rewards