Multi-Armed Bandits: An Introduction

RL Course by David Silver - Lecture 9: Exploration and Exploitation

Sutton Chapter 2

Pascal Poupart CS885 Reinforcement Learning, Lecture 8, 9

RoadMap

- Introduction
- Exploration vs. Exploitation
- Stochastic Bandits (greedy, UCB)
- Bayesian Bandits (Bayesian UCB, Thompson Sampling)
- Information State
- Contextual Bandits

Origin

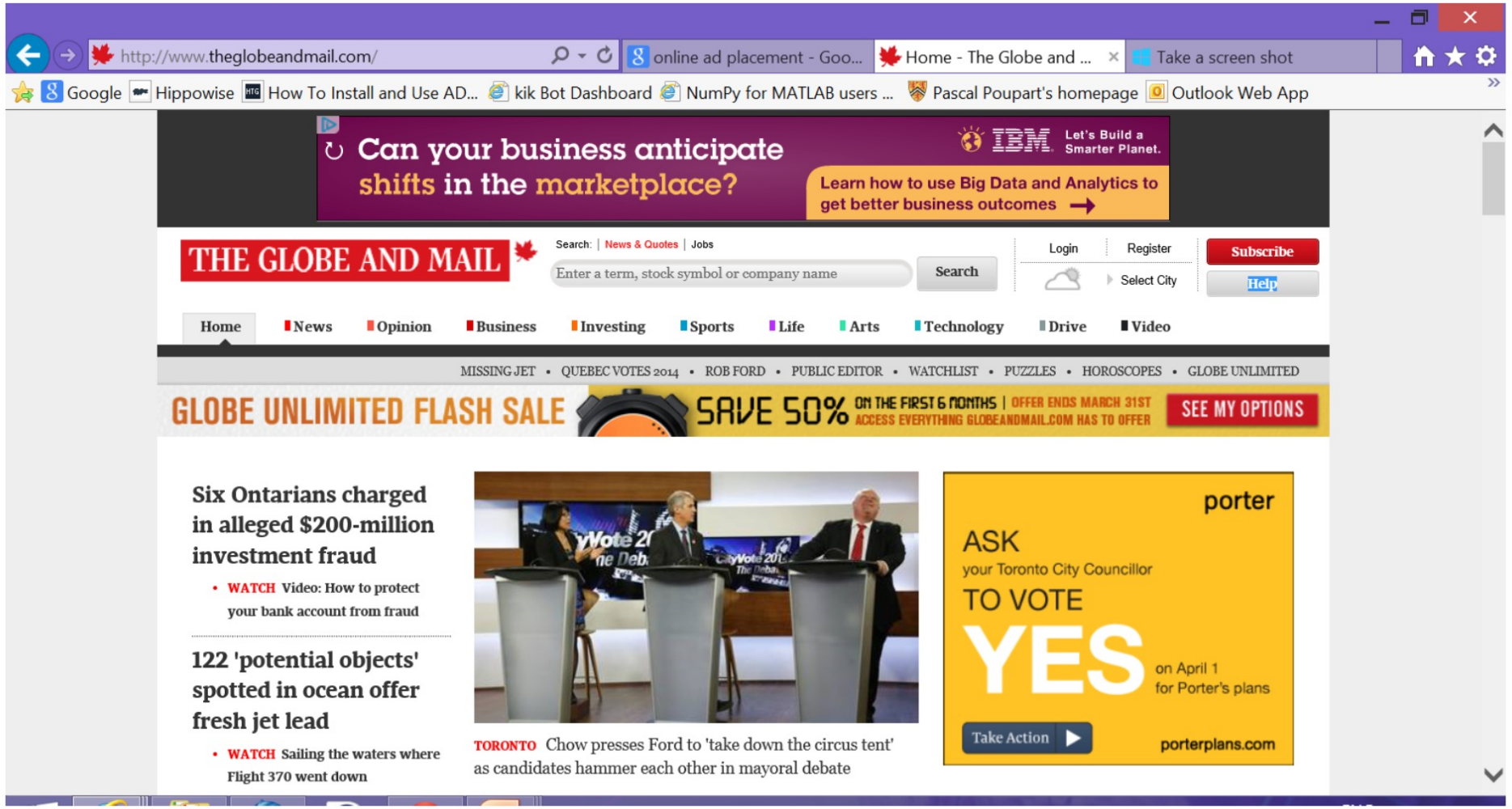
- The term bandit comes from gambling where slot machines can be thought as one-armed bandits.

- Problem: which slot machine should we play at each turn when their payoffs are not necessarily the same and initially unknown?

Examples

- Design of experiments (Clinical Trials)
- Online ad placement
- Web page personalization
- Games
- Networks (packet routing)

Online Ad Optimization



Online Ad Optimization

- Problem: which ad should be presented?
- Answer: present ad with highest payoff

$$payoff = clickThroughRate \times payment$$

  - Click through rate: probability that user clicks on ad
  - Payment: $$ paid by advertiser
    - Amount determined by an auction

Online Ad Optimization

- Assume payment is 1 unit for all ads
- Need to estimate click through rate

- Formulate as a bandit problem:
  - Arms: the set of possible ads
  - Rewards: 0 (no click) or 1 (click)

- In what order should ads be presented to maximize revenue?
  - How should we balance exploitation and exploration?

Simple yet difficult problem

- Simple: description of the problem is short
- Difficult: no known tractable optimal solution

Exploration vs. Exploitation Dilemma

- Online decision-making involves a fundamental choice:

  Exploitation Make the best decision given current information

  Exploration Gather more information

- The best long-term strategy may involve short-term sacrifices
- Gather enough information to make the best overall decisions

Examples

- Restaurant Selection

  Exploitation Go to your favourite restaurant
  Exploration Try a new restaurant

- Online Banner Advertisements

  Exploitation Show the most successful advert
  Exploration Show a different advert

- Oil Drilling

  Exploitation Drill at the best known location
  Exploration Drill at a new location

- Game Playing

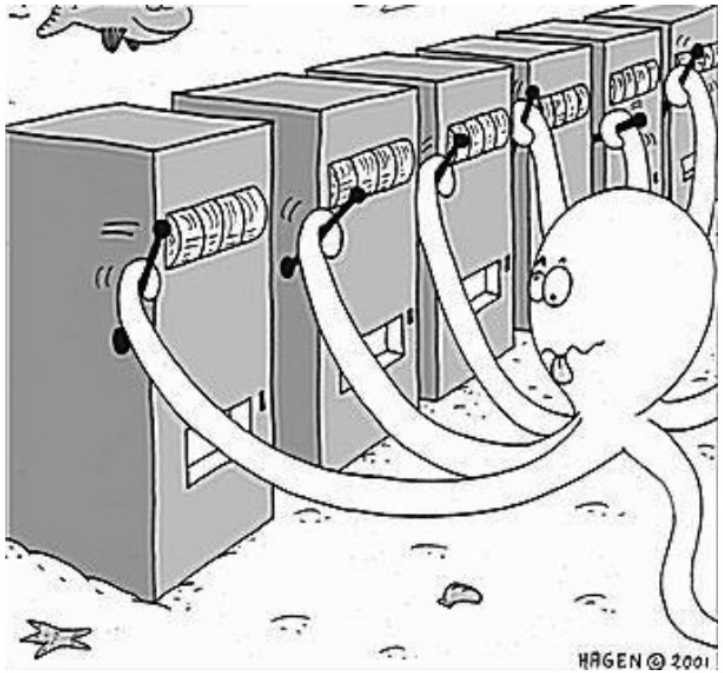
  Exploitation Play the move you believe is best
  Exploration Play an experimental move

Principles

- Naive Exploration
  - Add noise to greedy policy (e.g. ϵ-greedy)
- Optimistic Initialisation
  - Assume the best until proven otherwise
- Optimism in the Face of Uncertainty
  - Prefer actions with uncertain values
- Probability Matching
  - Select actions according to probability they are best
- Information State Search
  - Lookahead search incorporating value of information

Multi-Armed Bandits (Formal)

- A multi-armed bandit is a tuple $\langle \mathcal{A}, \mathcal{R} \rangle$
- $\mathcal{A}$ is a known set of m actions (or "arms")
- $\mathcal{R}^a(r) = \mathbb{P}[r|a]$ is an unknown probability distribution over rewards
- At each step t the agent selects an action $a_t \in \mathcal{A}$
- The environment generates a reward $r_t \sim \mathcal{R}^{a_t}$
- The goal is to maximise cumulative reward $\sum_{\tau=1}^{t} r_\tau$



Regret

- The *action-value* is the mean reward for action a,

$$Q(a) = \mathbb{E}[r|a]$$

- The *optimal value* V^* is

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$$

- The *regret* is the opportunity loss for one step

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

- The *total regret* is the total opportunity loss

$$L_t = \mathbb{E}\left[\sum_{\tau=1}^{t} V^* - Q(a_\tau)\right]$$

- Maximise cumulative reward $\equiv$ minimise total regret

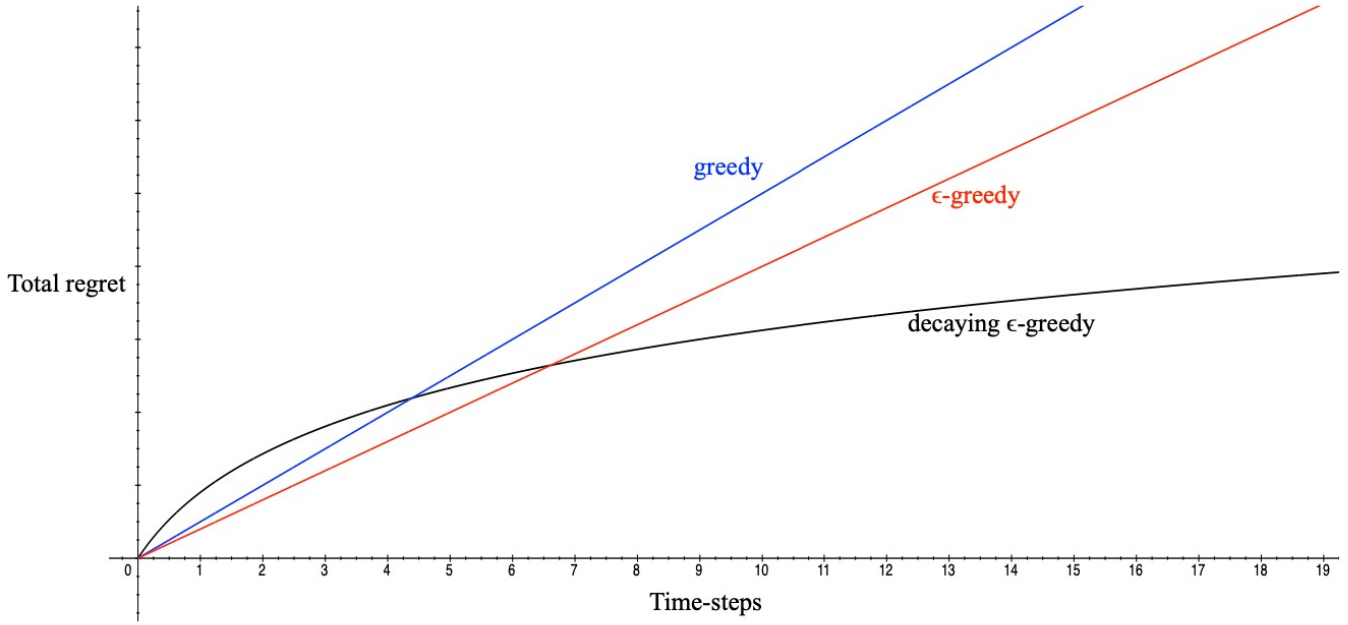
Counting Regret

- The *count* $N_t(a)$ is expected number of selections for action a
- The *gap* Δ_a is the difference in value between action a and optimal action a^*, $\Delta_a = V^* - Q(a)$
- Regret is a function of gaps and the counts

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

- A good algorithm ensures small counts for large gaps
- Problem: gaps are not known!

Linear vs. Sub-Linear 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?

Greedy

- We consider algorithms that estimate $\hat{Q}_t(a) \approx Q(a)$
- Estimate the value of each action by Monte-Carlo evaluation

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{t=1}^{T} r_t \mathbf{1}(a_t = a)$$

- The *greedy* algorithm selects action with highest value

$$a_t^* = \underset{a \in \mathcal{A}}{\text{argmax}}\ \hat{Q}_t(a)$$

- Greedy can lock onto a suboptimal action forever
- $\Rightarrow$ Greedy has linear total regret

A Simple Trick

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

$$\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\
&= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\
&= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big],
\end{aligned}$$

$$NewEstimate \leftarrow OldEstimate \; + \; StepSize \; \Big[Target - OldEstimate\Big]$$

Optimistic Initialization

- Simple and practical idea: initialise $Q(a)$ to high value
- Update action value by incremental Monte-Carlo evaluation
- Starting with $N(a) > 0$

$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)}(r_t - \hat{Q}_{t-1})$$

- Encourages systematic exploration early on
- But can still lock onto suboptimal action
- $\Rightarrow$ greedy + optimistic initialisation has linear total regret

ϵ - Greedy

- The ϵ-*greedy* algorithm continues to explore forever
  - With probability $1 - \epsilon$ select $a = \underset{a \in \mathcal{A}}{\text{argmax}}\ \hat{Q}(a)$
  - With probability ϵ select a random action
- Constant ϵ ensures minimum regret

$$l_t \geq \frac{\epsilon}{\mathcal{A}} \sum_{a \in \mathcal{A}} \Delta_a$$

- $\Rightarrow$ ϵ-greedy has linear total regret

Decay ϵ_t - Greedy

- Pick a decay schedule for $\epsilon_1, \epsilon_2, ...$
- Consider the following schedule

$$c > 0$$
$$d = \min_{a|\Delta_a > 0} \Delta_i$$
$$\epsilon_t = \min \left\{ 1, \frac{c|\mathcal{A}|}{d^2 t} \right\}$$

- Decaying ϵ_t-greedy has *logarithmic* asymptotic total regret!
- Unfortunately, schedule requires advance knowledge of gaps
- Goal: find an algorithm with sublinear regret for any multi-armed bandit (without knowledge of $\mathcal{R}$)

Lower Bound

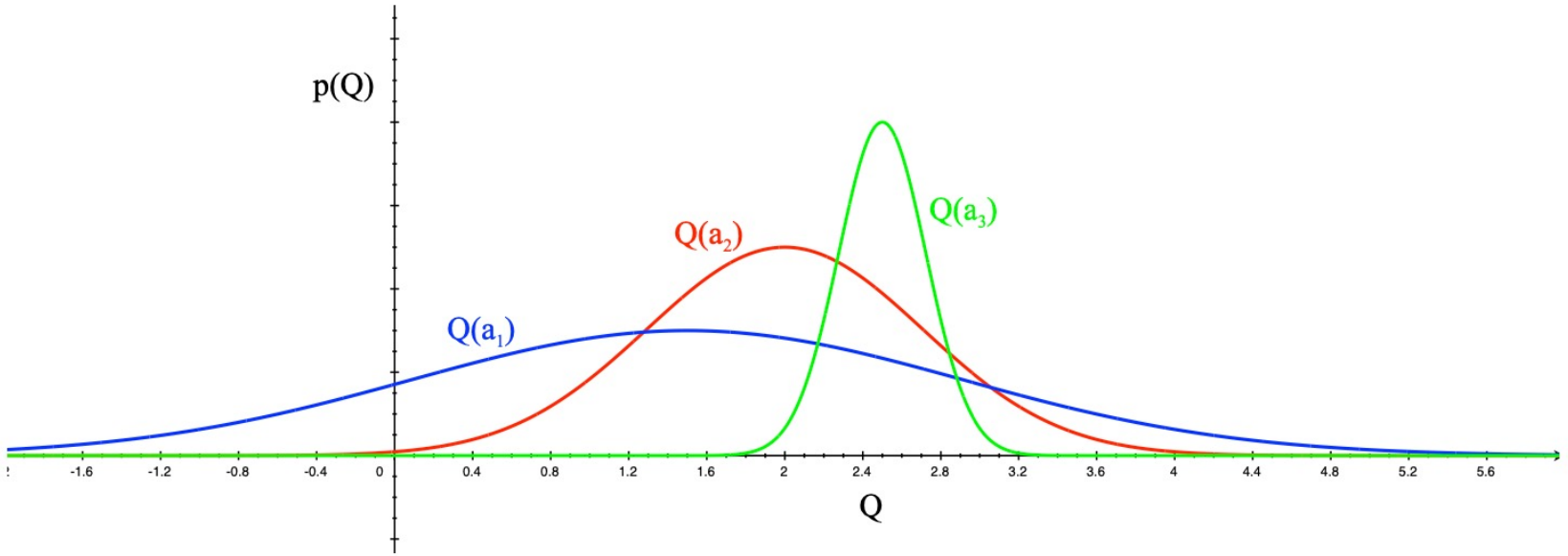
- The performance of any algorithm is determined by similarity between optimal arm and other arms
- Hard problems have similar-looking arms with different means
- This is described formally by the gap Δ_a and the similarity in distributions $KL(\mathcal{R}^a||\mathcal{R}^a*)$

Theorem (Lai and Robbins)

Asymptotic total regret is at least logarithmic in number of steps

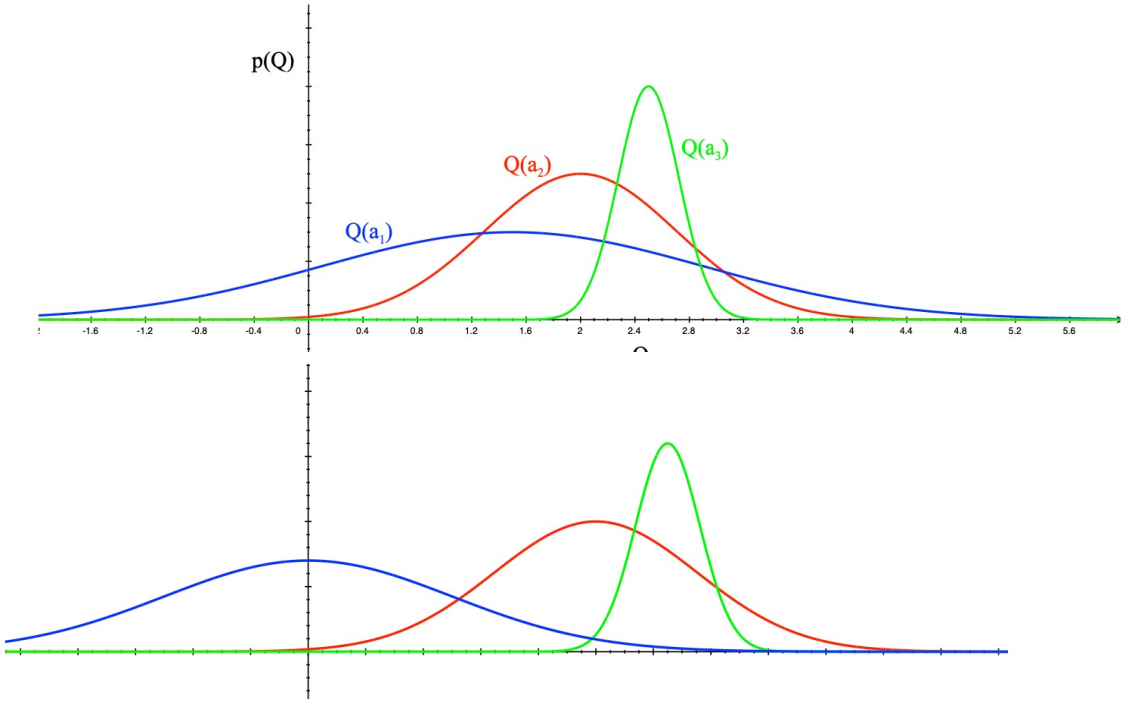
$$\lim_{t\to\infty} L_t \geq \log t \sum_{a|\Delta_a>0} \frac{\Delta_a}{KL(\mathcal{R}^a||\mathcal{R}^{a^*})}$$

Optimism in the Face of Uncertainty



- Which action should we pick?
- The more uncertain we are about an action-value
- The more important it is to explore that action
- It could turn out to be the best action

Optimism in the Face of Uncertainty



- After picking blue action
- We are less uncertain about the value
- And more likely to pick another action
- Until we home in on best action

Upper Confidence Bound

- Estimate an upper confidence $\hat{U}_t(a)$ for each action value
- Such that $Q(a) \leq \hat{Q}_t(a) + \hat{U}_t(a)$ with high probability
- This depends on the number of times $N(a)$ has been selected
  - Small $N_t(a) \Rightarrow$ large $\hat{U}_t(a)$ (estimated value is uncertain)
  - Large $N_t(a) \Rightarrow$ small $\hat{U}_t(a)$ (estimated value is accurate)
- Select action maximising Upper Confidence Bound (UCB)

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ \hat{Q}_t(a) + \hat{U}_t(a)$$

Hoeffding's 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

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

- We will apply Hoeffding's Inequality to rewards of the bandit
- conditioned on selecting action a

$$\mathbb{P}\left[Q(a) > \hat{Q}_t(a) + U_t(a)\right] \leq e^{-2N_t(a)U_t(a)^2}$$

Calculating Upper Confidence Bound

- Pick a probability p that true value exceeds UCB
- Now solve for $U_t(a)$

$$e^{-2N_t(a)U_t(a)^2} = p$$

$$U_t(a) = \sqrt{\frac{-\log p}{2N_t(a)}}$$

- Reduce p as we observe more rewards, e.g. $p = t^{-4}$
- Ensures we select optimal action as $t \to \infty$

$$U_t(a) = \sqrt{\frac{2 \log t}{N_t(a)}}$$

UCB1

- This leads to the UCB1 algorithm

$$a_t = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\ Q(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$$

Theorem

The UCB algorithm achieves logarithmic asymptotic total regret

$$\lim_{t \to \infty} L_t \leq 8 \log t \sum_{a | \Delta_a > 0} \Delta_a$$

Other UCB Bounds

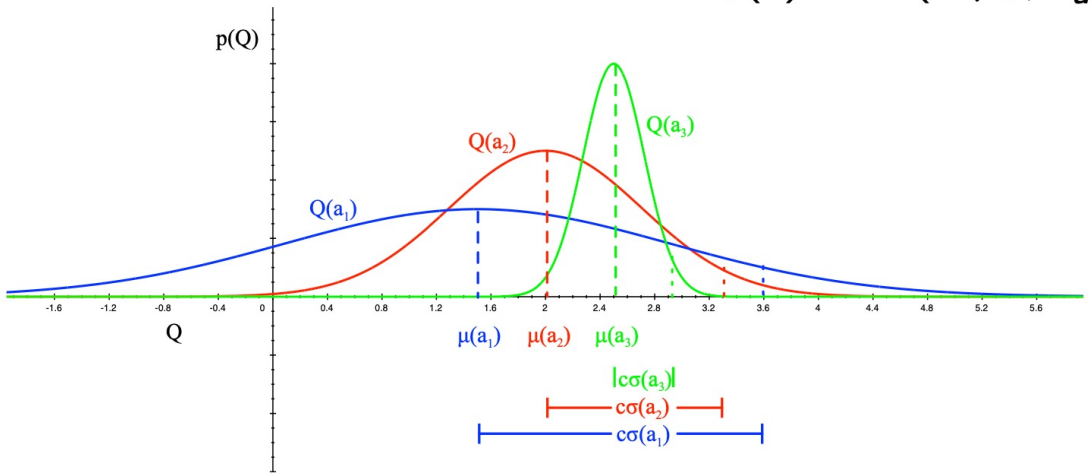
- UCB can be applied to other inequalities
  - Bernstein's inequality
  - Empirical Bernstein's inequality
  - Chernoff inequality
  - Azuma's inequality
  - …

Bayesian Bandits

- So far we have made no assumptions about the reward distribution $\mathcal{R}$
  - Except bounds on rewards
- Bayesian bandits exploit prior knowledge of rewards, $p[\mathcal{R}]$
- They compute posterior distribution of rewards $p[\mathcal{R} \mid h_t]$
  - where $h_t = a_1, r_1, ..., a_{t-1}, r_{t-1}$ is the history
- Use posterior to guide exploration
  - Upper confidence bounds (Bayesian UCB)
  - Probability matching (Thompson sampling)
- Better performance if prior knowledge is accurate

Bayesian UCB

- Assume reward distribution is Gaussian, $\mathcal{R}_a(r) = \mathcal{N}(r; \mu_a, \sigma_a^2)$



- Compute Gaussian posterior over μ_a and σ_a^2 (by Bayes law)

$$p\left[\mu_a, \sigma_a^2 \mid h_t\right] \propto p\left[\mu_a, \sigma_a^2\right] \prod_{t \mid a_t = a} \mathcal{N}(r_t; \mu_a, \sigma_a^2)$$

- Pick action that maximises standard deviation of $Q(a)$

$$a_t = \text{argmax}\ \mu_a + c\sigma_a / \sqrt{N(a)}$$

Probability Matching

- Probability matching selects action a according to probability that a is the optimal action

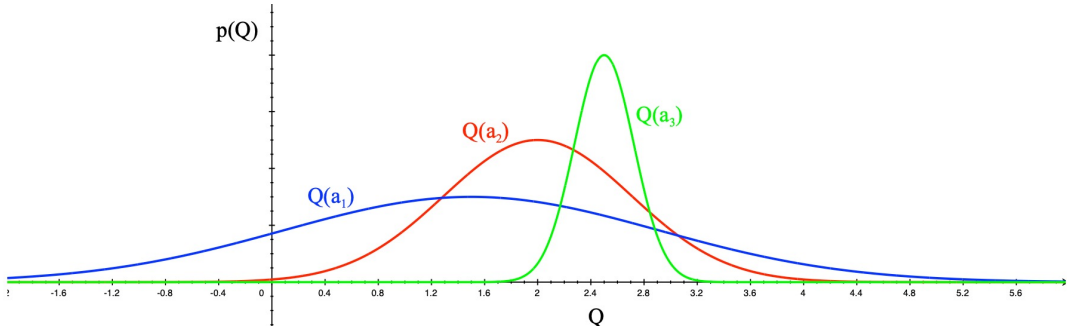
$$\pi(a \mid h_t) = \mathbb{P}\left[Q(a) > Q(a'), \forall a' \neq a \mid h_t\right]$$

- Probability matching is optimistic in the face of uncertainty
  - Uncertain actions have higher probability of being max
- Can be difficult to compute analytically from posterior

Thompson Sampling

- Thompson sampling implements probability matching

$$\pi(a \mid h_t) = \mathbb{P}\left[Q(a) > Q(a'), \forall a' \neq a \mid h_t\right]$$
$$= \mathbb{E}_{\mathcal{R}|h_t}\left[\mathbf{1}(a = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q(a))\right]$$



- Use Bayes law to compute posterior distribution $p\left[\mathcal{R} \mid h_t\right]$
- Sample a reward distribution $\mathcal{R}$ from posterior
- Compute action-value function $Q(a) = \mathbb{E}\left[\mathcal{R}_a\right]$
- Select action maximising value on sample, $a_t = \underset{a \in \mathcal{A}}{\text{argmax}}\, Q(a)$
- Thompson sampling achieves Lai and Robbins lower bound!

Value of Information

- Exploration is useful because it gains information
- Can we quantify the value of information?
  - How much reward a decision-maker would be prepared to pay in order to have that information, prior to making a decision
  - Long-term reward after getting information - immediate reward
- Information gain is higher in uncertain situations
- Therefore it makes sense to explore uncertain situations more
- If we know value of information, we can trade-off exploration and exploitation *optimally*

Information State Space

- We have viewed bandits as *one-step* decision-making problems
- Can also view as *sequential* decision-making problems
- At each step there is an *information state* $\tilde{s}$
  - $\tilde{s}$ is a statistic of the history, $\tilde{s}_t = f(h_t)$
  - summarising all information accumulated so far
- Each action a causes a transition to a new information state $\tilde{s}'$ (by adding information), with probability $\tilde{\mathcal{P}}^a_{\tilde{s},\tilde{s}'}$
- This defines MDP $\tilde{\mathcal{M}}$ in augmented information state space

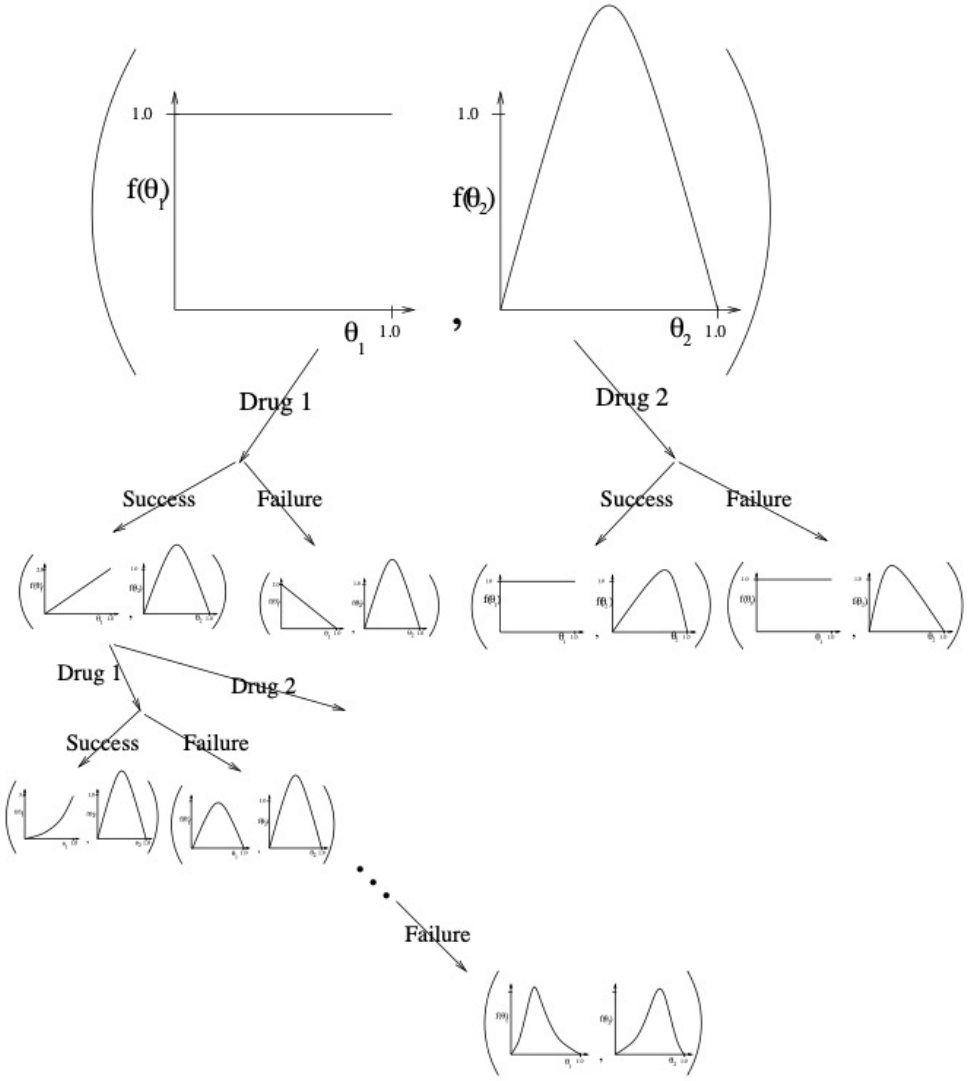
$$\tilde{\mathcal{M}} = \langle \tilde{\mathcal{S}}, \mathcal{A}, \tilde{\mathcal{P}}, \mathcal{R}, \gamma \rangle$$

Example: Bernoulli Bandits

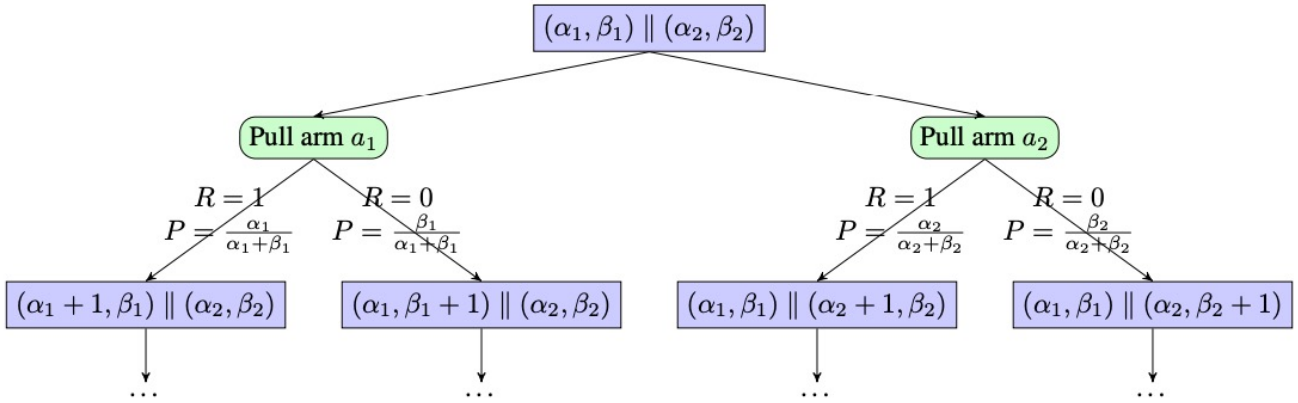
- Consider a Bernoulli bandit, such that $\mathcal{R}^a = \mathcal{B}(\mu_a)$
- e.g. Win or lose a game with probability μ_a
- Want to find which arm has the highest μ_a
- The information state is $\tilde{s} = \langle \alpha, \beta \rangle$
  - α_a counts the pulls of arm a where reward was 0
  - β_a counts the pulls of arm a where reward was 1

Bayes-Adaptive Bernoulli Bandits

- Start with $Beta(\alpha_a, \beta_a)$ prior over reward function $\mathcal{R}^a$
- Each time a is selected, update posterior for $\mathcal{R}^a$
  - $Beta(\alpha_a + 1, \beta_a)$ if $r = 0$
  - $Beta(\alpha_a, \beta_a + 1)$ if $r = 1$
- This defines transition function $\tilde{\mathcal{P}}$ for the Bayes-adaptive MDP
- Information state $\langle \alpha, \beta \rangle$ corresponds to reward model $Beta(\alpha, \beta)$
- Each state transition corresponds to a Bayesian model update



Bayes-Adaptive Bernoulli Bandits



- We now have an infinite MDP over information states
- This MDP can be solved by reinforcement learning
- Model-free reinforcement learning
  - e.g. Q-learning (Duff, 1994)
- Bayes-adaptive MDP can be solved by dynamic programming
- The solution is known as the *Gittins index*
- Exact solution to Bayes-adaptive MDP is typically intractable
  - Information state space is too large
- Recent idea: apply simulation-based search (Guez et al. 2012)

Contextual Bandits: Example

- A contextual bandit is a tuple $\langle \mathcal{A}, \mathcal{S}, \mathcal{R} \rangle$
- $\mathcal{A}$ is a known set of actions (or "arms")
- $\mathcal{S} = \mathbb{P}[s]$ is an unknown distribution over states (or "contexts")
- $\mathcal{R}_s^a(r) = \mathbb{P}[r|s, a]$ is an unknown probability distribution over rewards
- At each step t
  - Environment generates state $s_t \sim \mathcal{S}$
  - Agent selects action $a_t \in \mathcal{A}$
  - Environment generates reward $r_t \sim \mathcal{R}_{s_t}^{a_t}$
- Goal is to maximise cumulative reward $\sum_{\tau=1}^{t} r_\tau$

Contextual Bandits

- In many applications, the context provides additional information to select an action
  - E.g., personalized advertising, user interfaces
  - Context: user demographics (location, age, gender)

- Actions can also be characterized by features that influence their payoff
  - E.g., ads, webpages
  - Action features: topics, keywords, etc.

Contextual Bandits

- Contextual bandits: multi-armed bandits with states (corresponding to contexts) and action features
- Formally:
  - S: set of states where each state s is defined by a vector of features $\boldsymbol{x}^s = (x_1^s, x_2^s, \ldots, x_k^s)$
  - A: set of actions where each action a is associated with a vector of features $\boldsymbol{x}^a = (x_1^a, x_2^a, \ldots, x_l^a)$
  - Space of rewards (often $\mathbb{R}$)
- No transition function since the states at each step are independent
- Goal find policy $\pi: \boldsymbol{x}^s \rightarrow a$ that maximizes expected rewards $E(r|s,a) = E(r|\boldsymbol{x}^s, \boldsymbol{x}^a)$

Approximate Reward Function

- Common approach:
  - learn approximate average reward function $\tilde{R}(s,a) = \tilde{R}(\boldsymbol{x})$ (where $\boldsymbol{x} = (\boldsymbol{x}^s, \boldsymbol{x}^a)$) by regression

- Linear approximation: $\tilde{R}_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$

- Non-linear approximation: $\tilde{R}_{\boldsymbol{w}}(\boldsymbol{x}) = neuralNet(\boldsymbol{x}; \boldsymbol{w})$

Bayesian Linear Regression

- Consider a Gaussian prior:

$$pdf(\boldsymbol{w}) = N(\boldsymbol{w}|\boldsymbol{0}, \lambda^2 \boldsymbol{I}) \propto exp\left(-\frac{\boldsymbol{w}^T\boldsymbol{w}}{2\lambda^2}\right)$$

- Consider also a Gaussian likelihood:

$$pdf(r|\boldsymbol{x}, \boldsymbol{w}) = N(r|\boldsymbol{w}^T\boldsymbol{x}, \sigma^2) \propto exp\left(-\frac{(r - \boldsymbol{w}^T\boldsymbol{x})^2}{2\sigma^2}\right)$$

- The posterior is also Gaussian:

$$pdf(\boldsymbol{w}|r, \boldsymbol{x}) \propto pdf(\boldsymbol{w}) \Pr(r|\boldsymbol{x}, \boldsymbol{w})$$
$$\propto exp\left(-\frac{\boldsymbol{w}^T\boldsymbol{w}}{2\lambda^2}\right) exp\left(-\frac{(r - \boldsymbol{w}^T\boldsymbol{x})^2}{2\sigma^2}\right)$$
$$= N(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where $\boldsymbol{\mu} = \sigma^{-2}\boldsymbol{\Sigma}\boldsymbol{x}r$ and $\boldsymbol{\Sigma} = (\sigma^{-2}\boldsymbol{x}\boldsymbol{x}^T + \lambda^{-2}\boldsymbol{I})^{-1}$

Predictive

- Consider a state-action pair $(\boldsymbol{x}^s, \boldsymbol{x}^a) = \boldsymbol{x}$ for which we would like to predict the reward r

- Predictive posterior:

$$pdf(r|\boldsymbol{x}) = \int_{\boldsymbol{w}} N(r|\boldsymbol{w}^T\boldsymbol{x}, \sigma^2)\, N(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\, d\boldsymbol{w}$$
$$= N(r|\sigma^2\boldsymbol{x}^T\boldsymbol{\mu}, \boldsymbol{x}^T\boldsymbol{\Sigma}\boldsymbol{x})$$

- UCB: $\Pr\left(r < \sigma^2\boldsymbol{x}^T\boldsymbol{\mu} + c\sqrt{\boldsymbol{x}^T\boldsymbol{\Sigma}\boldsymbol{x}}\right) > 1 - \delta$

  where $c = 1 + \sqrt{\ln(2/\delta)\,/2}$

- Thomson sampling: $\tilde{r} \sim N(r|\sigma^2\boldsymbol{x}^T\boldsymbol{\mu}, \boldsymbol{x}^T\boldsymbol{\Sigma}\boldsymbol{x})$

UCB Linear Gaussian

UCB(h)

$V \leftarrow 0$, $pdf(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = N(\boldsymbol{w}|\boldsymbol{0}, \lambda^2 \boldsymbol{I})$

Repeat until $n = h$

Receive state $\boldsymbol{x}^s$

For each action $\boldsymbol{x}^a$ where $\boldsymbol{x} = (\boldsymbol{x}^s, \boldsymbol{x}^a)$ do

$confidenceBound(a) = \sigma^2 \boldsymbol{x}^T \boldsymbol{\mu} + c\sqrt{\boldsymbol{x}^T \boldsymbol{\Sigma} \boldsymbol{x}}$

$a^* \leftarrow \text{argmax}_a \, confidenceBound(a)$

Execute a^* and receive r

$V \leftarrow V + r$

update $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ based on $\boldsymbol{x} = (\boldsymbol{x}^s, \boldsymbol{x}^{a^*})$ and r

Return V

Thompson Sampling Linear Gaussian

ThompsonSampling(h)

$V \leftarrow 0$; $pdf(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = N(\boldsymbol{w}|\boldsymbol{0}, \lambda^2 \boldsymbol{I})$

For $n = 1$ to h

Receive state $\boldsymbol{x}^s$

For each action $\boldsymbol{x}^a$ where $\boldsymbol{x} = (\boldsymbol{x}^s, \boldsymbol{x}^a)$ do

Sample $R_1(a), \ldots, R_k(a) \sim N(r|\sigma^2 \boldsymbol{x}^T \boldsymbol{\mu}, \boldsymbol{x}^T \boldsymbol{\Sigma} \boldsymbol{x})$

$\hat{R}(a) \leftarrow \frac{1}{k} \sum_{i=1}^{k} R_i(a)$

$a^* \leftarrow \text{argmax}_a \hat{R}(a)$

Execute a^* and receive r

$V \leftarrow V + r$

update $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ based on $\boldsymbol{x} = (\boldsymbol{x}^s, \boldsymbol{x}^{a^*})$ and r

Return V

Real Application of C-MAB

- Contextual bandits are now commonly used for
  - Personalized advertising
  - Personalized web content
    - MSN news: 26% improvement in click through rate after adoption of contextual bandits (https://www.microsoft.com/en-us/research/blog/real-world-interactive-learning-cusp-enabling-new-class-applications/)