Introduction to Bandits

Chris Liaw

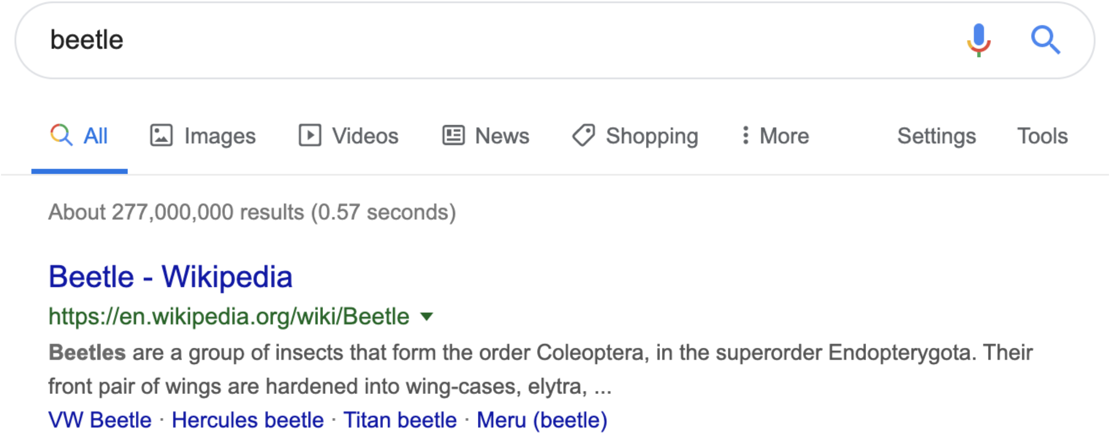
Machine Learning Reading Group

July 10, 2019

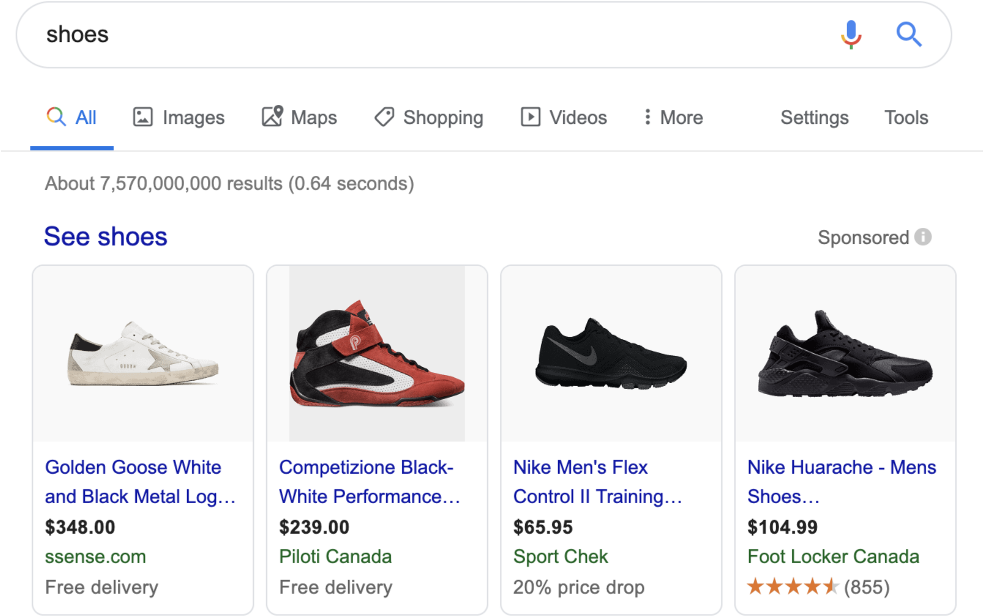
"*[T]he problem is a classic one; it was formulated during the war, and efforts to solve it so sapped the energies and minds of Allied analysts that the suggestion was made that the problem be dropped over Germany, as the ultimate instrument of intellectual sabotage*" - Peter Whittle (on the bandit problem)

Motivation and applications

Clinical trials (Thompson '33)



Search results



Online ads

Many others: network routing, recommender systems, etc.

Outline

- Intro to stochastic bandits
- Explore-then-commit
- Upper confidence bound algorithm
- Adversarial bandits & Exp3
- Application: Learning Diverse Rankings

Intro to stochastic bandits

K arms; unknown sequence of *stochastic* rewards $R_1, R_2, \ldots \in [0,1]^K$; $R_t \sim \nu$

For each round $t = 1,2, \ldots, T$ (assume horizon T is known; will say more later)

- Choose arm $A_t \in [K]$
- Obtain reward R_{t,A_t} and *only* see R_{t,A_t}

? ? ? ?

Problem was introduced by Robbins (1952).

? ? 0.2 ?

Pull arm 3

0.01 ? ? ?

Pull arm 1

? ? 0.6 ?

Pull arm 3

Intro to stochastic bandits

K arms; unknown sequence of *stochastic* rewards $R_1, R_2, \ldots \in [0,1]^K$; $R_t \sim \nu$

For each round $t = 1,2,\ldots,T$ (assume horizon T is known; will say more later)

- Choose arm $A_t \in [K]$
- Obtain reward R_{t,A_t} and *only* see R_{t,A_t}

Arm i has mean μ_i *which is unknown.*

Goal: Find a policy that minimizes the regret

$$Reg(T) = T \cdot \mu^* - E\left[\sum_{t\in[T]} R_{t,A_t}\right] \qquad \mu^* = \max_i \mu_i$$

Reward of best arm

Algorithm's reward

Ideally, we would like that $Reg(T) = o(T)$.

Exploration-Exploitation tradeoff

At each time step, we can either:

1. (Exploit) Pull the arm we think is the best one; or
2. (Explore) Pull an arm we think is suboptimal.

1. We do not know which is the best arm so if we keep exploiting, we may keep pulling a suboptimal arm which may incur large regret.
2. If we explore, we gather information about the arms, but we pull suboptimal arms so may incur large regret again!

Challenge is to tradeoff exploration and exploitation!

Explore-then-commit (ETC)

Perhaps the simplest algorithm that ***provably*** gets sublinear regret!

Let T_0 be a hyper-parameter and assume $T \geq K \cdot \boldsymbol{T_0}$.

1. Pull each of K arms $\boldsymbol{T_0}$ times.
2. Compute empirical average $\widehat{\mu_i}$ of each arm.
3. Pull arm with largest empirical average for remaining $T - K \cdot T_0$ rounds.

Theorem. Let $\Delta_i := \mu^* - \mu_i$ be suboptimality of arm i. Then

$$Reg(T) \leq \boldsymbol{T_0} \sum_{i \in [K]} \boldsymbol{\Delta_i} + (T - K \cdot \boldsymbol{T_0}) \cdot \sum_{i \in [K]} \boldsymbol{\Delta_i} \exp\left(-\boldsymbol{T_0} \cdot \frac{\boldsymbol{\Delta_i^2}}{C}\right)$$

"Cost of exploration"

Suboptimality of each additional step.

Note: The term $\boldsymbol{\Delta_i} \exp\left(-\boldsymbol{T_0} \cdot \frac{\boldsymbol{\Delta_i^2}}{4}\right)$ is small when $\boldsymbol{T_0}$ is large.

Explore-then-commit (ETC)

Theorem. Let $\Delta_i := \mu^* - \mu_i$ be suboptimality of arm i. Then

$$Reg(T) \leq \boldsymbol{T_0} \sum_{i \in [K]} \boldsymbol{\Delta_i} + (T - K \cdot \boldsymbol{T_0}) \cdot \sum_{i \in [K]} \boldsymbol{\Delta_i} \exp\left(-\boldsymbol{T_0} \cdot \frac{\boldsymbol{\Delta_i^2}}{C}\right)$$

- This illustrates exploration-exploitation tradeoff:
  - Explore too much (T_0 large) then first term is large.
  - Exploit too much (T_0 small) then second term is large.
- Can we tune exploration (i.e. T_0) to get sublinear regret?
- Yes! Choose $T_0 = T^{2/3}$. Can show that $Reg(T) = O(K \cdot T^{2/3})$.
- If $K = 2$ arms, can use a data-dependent T_0 to get $Reg(T) = O(T^{1/2})$ [Garivier, Kaufmann, Lattimore NeurIPS '16]

Explore-then-commit (ETC)

Theorem. Let $\Delta_i := \mu^* - \mu_i$ be suboptimality of arm i. Then

$$Reg(T) \le \boldsymbol{T_0} \sum_{i\in[K]} \boldsymbol{\Delta_i} + (T - K \cdot \boldsymbol{T_0}) \cdot \sum_{i\in[K]} \boldsymbol{\Delta_i} \exp\left(-\boldsymbol{T_0} \cdot \frac{\boldsymbol{\Delta_i^2}}{C}\right)$$

Sketch.

- Initially, we try each arm i for T_0 trials; this incurs regret $T_0 \cdot \Delta_i$
- Next, we exploit; we only pull arm i again if empirical average of arm i is at least that of best arm.
  - This happens with probability at most $\exp\left(-\boldsymbol{T_0} \cdot \frac{\boldsymbol{\Delta_i^2}}{C}\right)$.
- Summing the contribution from all arms gives the claimed regret.

Aside: Doubling Trick

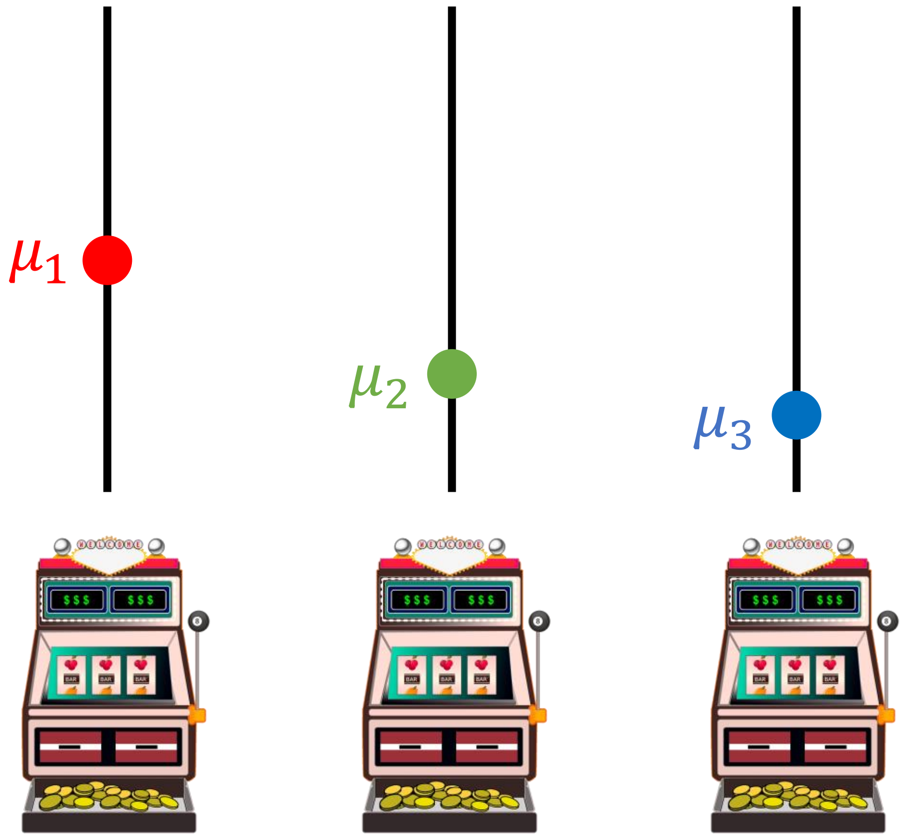
- Previously, we assumed that time horizon T is known beforehand.
- The doubling trick can be used to get around that.

- Suppose that some algorithm $\mathcal{A}$ has regret $o(T)$ *if it knew the time horizon beforehand*.
- At every power of 2 step (i.e. at step 2^k for some k), we reset $\mathcal{A}$ and assume time horizon is 2^k.
- Then this gives an algorithm with regret $o(t)$ for *all* t, i.e. an "anytime algorithm".

Upper confidence bound (UCB) algorithm

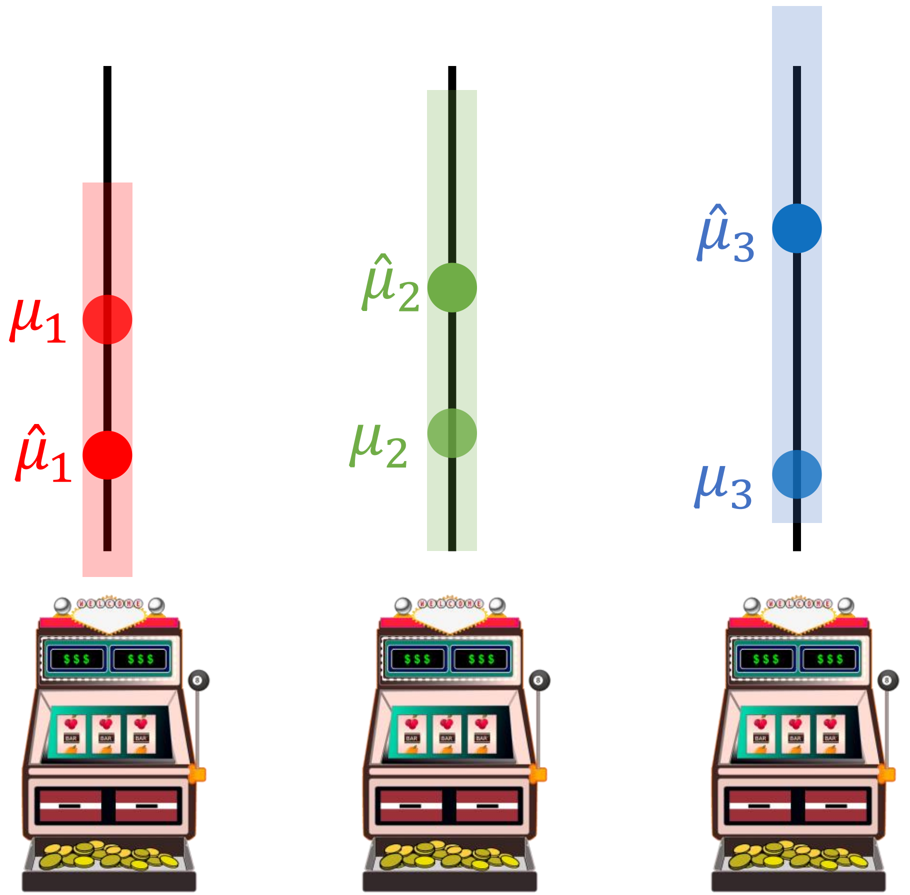
- Based on the idea of “optimism in the face of uncertainty.”
- Algorithm: compute the empirical mean of each arm and a confidence interval; use the **upper confidence bound** as a proxy for goodness of arm.
  - Note: confidence interval chosen so that true mean is very unlikely to be outside of confidence interval.

Upper confidence bound (UCB) algorithm



Start by pulling each arm once.

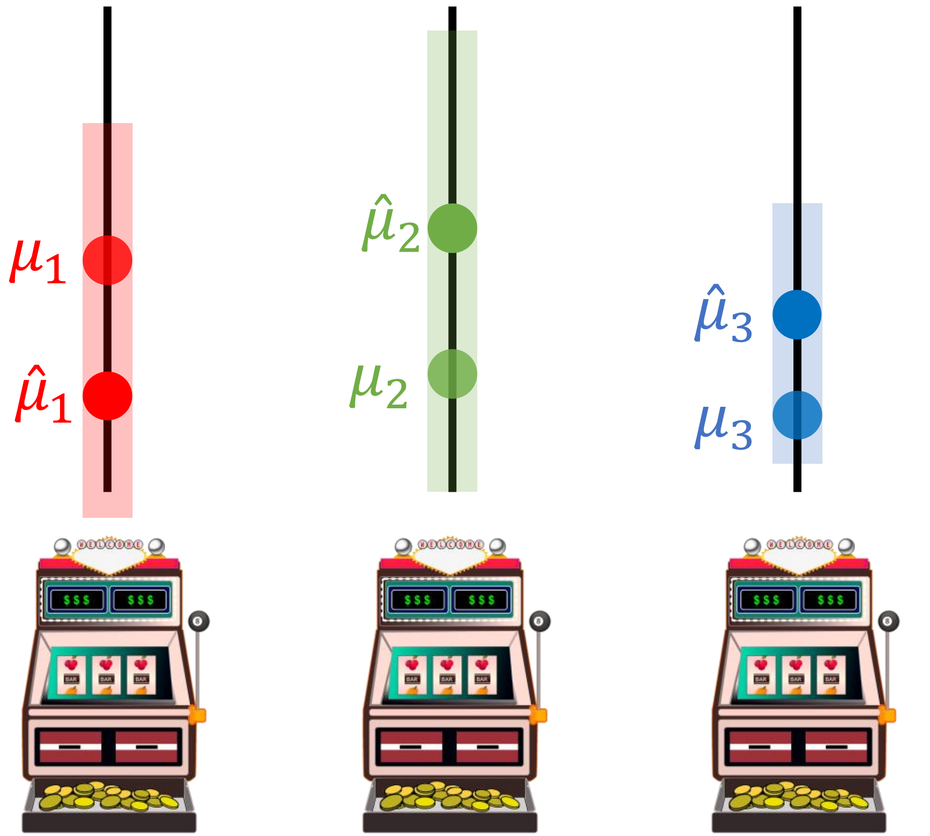
Upper confidence bound (UCB) algorithm



Start by pulling each arm once.

Arm 3 has the highest UCB, we pull that next.

Upper confidence bound (UCB) algorithm



Start by pulling each arm once.

Arm 3 has the highest UCB, we pull that next.

Now, arm 2 has the highest UCB; we pull arm 2.

Upper confidence bound (UCB) algorithm

Let $\delta \in (0,1)$ be a hyper-parameter.

- Pull each of K arms once.
- For $t = K+1, K+2, \dots, T$
  1. Let $N_i(t)$ be number of times arm i was pulled so far and $\hat{\mu}_i(t)$ be empirical average.
  2. Let $UCB_i(t) = \hat{\mu}_i(t) + \sqrt{\mathbf{2\log\left(\frac{1}{\delta}\right)/N_i(t)}}$
  3. Play arm in $\arg\max UCB_i(t)$.

Claim. Fix an arm i. Then with probability at least $1 - 2\delta$, we have

$$|\mu_i - \hat{\mu}_i(t)| \le \sqrt{\mathbf{2\log\left(\frac{1}{\delta}\right)/N_i(t)}}$$

Upper confidence bound (UCB) algorithm

Theorem. Let $\Delta_i := \mu^* - \mu_i$ be suboptimality of arm i. If we choose $\delta \sim 1/T^2$:

$$Reg(T) \leq C \sum_{i \in [K]} \Delta_i + \sum_{i: \Delta_i > 0} \frac{C \log(T)}{\Delta_i}$$

Always have to pay.

This turns out to mean the following:

If $\Delta_i > 0$, we only pull arm i roughly $\Delta_i^{-2} \log(T)$ times incurring regret Δ_i each time.

Upper confidence bound (UCB) algorithm

Theorem. Let $\Delta_i := \mu^* - \mu_i$ be suboptimality of arm i. If we choose $\delta \sim 1/T^2$:

$$Reg(T) \leq C \sum_{i \in [K]} \Delta_i + \sum_{i:\Delta_i > 0} \frac{C \log(T)}{\Delta_i}$$

Sketch.

- **Fact.** $Reg(T) = \sum_{i:\Delta_i>0} \Delta_i E[N_i(T)]$ ($N_i(T)$ counts number of times arm i was pulled up to time T)
- Want to bound $E[N_i(T)]$ whenever $\Delta_i > 0$.
- W.h.p. $UCB_i(t) = \hat{\mu}_i(t) + \sqrt{2 \log\left(\frac{1}{\delta}\right) / N_i(t)} \leq \mu_i + 2\sqrt{2 \log\left(\frac{1}{\delta}\right) / N_i(t)}$
- If $N_i(t) \geq \Omega\left(\log\left(\frac{1}{\delta}\right) \Delta_i^{-2}\right)$ then $UCB_i(t) < \mu^*$ so will pull $O\left(\log\left(\frac{1}{\delta}\right) \Delta_i^{-2}\right)$ w.h.p.
- To conclude, if $\Delta_i > 0$ then $\Delta_i E[N_i(T)] \lesssim O\left(\log\left(\frac{1}{\delta}\right) \Delta_i^{-1}\right)$.
- Choose $\boldsymbol{\delta \sim 1/T^2}$ to beat union bound.

Upper confidence bound (UCB) algorithm

Theorem. Let $\Delta_i := \mu^* - \mu_i$ be suboptimality of arm i. If we choose $\delta \sim 1/T^2$:

$$Reg(T) \leq C \sum_{i \in [K]} \Delta_i + \sum_{i : \Delta_i > 0} \frac{C \log(T)}{\Delta_i}$$

This is an instance-dependent bound but we can also get a instance-free bound.

Corollary. If we choose $\delta \sim 1/T^2$ then

$$Reg(T) \leq O\left(\sqrt{TK \cdot \log T}\right)$$

So regret is $O_K\left(\sqrt{T \cdot \log T}\right)$. (Recall that ETC has regret $O_K\left(T^{2/3}\right)$.)

It is possible to get regret $O\left(\sqrt{TK}\right)$ [Audibert, Bubeck '10]; this is optimal.

UCB can also be extended to heavier tails (e.g. [Bubeck, Cesa-Bianchi, Lugosi '13])

ϵ-greedy algorithm

Let $\epsilon_{K+1}, \epsilon_{K+2}, \ldots \in [0,1]$ be an exploration schedule.

- Pull each of K arms once.
- For $t = K+1, K+2, \ldots$
    1. With probability ϵ_t, pull a random arm; otherwise pull arm with highest empirical mean.

Theorem. For an appropriate choice of ϵ_t, can show

$$Reg(t) = O(t^{2/3}(K \log t)^{1/3}).$$

Choosing $\epsilon_t = t^{-1/3}(K \cdot \log t)^{1/3}$ will give the theorem (see Theorem 1.4 in book by Slivkins).

Adversarial bandits

Assume K experts and *rewards* $r_t \in [0,1]^K$
Initialize p_1 (e.g. uniform distribution over experts)
For time $t = 1, 2, \ldots$

1. Algorithm plays according to p_t; say chooses action j
2. Algorithm gains $\langle p_t, r_t \rangle$ (expected reward over randomness of action)
3. Algorithm receives $\boldsymbol{r_{t,j}}$ and updates p_t to get p_{t+1}.

The *only* difference with expert setting (where r_t is revealed).

Goal: minimize "pseudo"-regret over all reward vectors (same as experts)

$$Reg(T) = \max_{i \in [K]} \sum_t r_{t,i} - \sum_t \langle p_t, r_t \rangle$$

Adversarial bandits and Exp3

Assume K experts and *rewards* $r_t \in [0,1]^K$
Initialize p_1 (e.g. uniform distribution over experts)
For time $t = 1, 2, \ldots$
1. Algorithm plays according to p_t; say chooses action j
2. Algorithm gains $\langle p_t, r_t \rangle$ (expected reward over randomness of action)
3. Algorithm receives $\boldsymbol{r_{t,j}}$ and updates p_t to get p_{t+1}.

A nifty trick:
- Algorithm only receives $r_{t,j}$; ideally, we would like r_t
- Define $\tilde{r}_{t,j} = \frac{r_{t,j}}{p_{t,j}}$ if algorithm chose action j and $\tilde{r}_{t,j} = 0$ otherwise.
- Then $E[\tilde{r}_t] = r_t$, i.e. algorithm can get an unbiased estimate of r_t.
- One gets Exp3 algorithm by replacing r_t in MWU with $\tilde{r}_t$!

Exp3

MWU. Assume K experts and *rewards* $r_t \in [0,1]^K$; step size η
Initialize $R_0 = (0, \dots, 0)$
For time $t = 1, 2, \dots, T$

1. Set $p_{t,j} = \exp\left(\eta R_{t-1,j}\right) / Z_{t-1}$ where $Z_{t-1} = \sum_i \exp(\eta R_{t-1,i})$.
2. Follow expert j with prob. $p_{t,j}$. Expected reward is $\langle p_t, r_t \rangle$.
3. Algorithm observes $\boldsymbol{r_t}$.
4. Update: $R_{t,j} = R_{t-1,j} + \boldsymbol{r_{t,j}}$ for all j.

Exp3

Exp3. Assume K experts and *rewards* $r_t \in [0,1]^K$; step size η

Initialize $R_0 = (0, \dots, 0)$

For time $t = 1, 2, \dots, T$

1. Set $p_{t,j} = \exp\left(\eta R_{t-1,j}\right) / Z_{t-1}$ where $Z_{t-1} = \sum_i \exp(\eta R_{t-1,i})$.
2. Follow expert j with prob. $p_{t,j}$. Expected reward is $\langle p_t, r_t \rangle$.
3. Algorithm observes $\boldsymbol{r_{t,j}}$. **Set** $\tilde{\boldsymbol{r}}_{t,j} = \boldsymbol{r_{t,j}/p_{t,j}}$ **if follow expert** $\boldsymbol{j}$**; else** $\tilde{\boldsymbol{r}}_{t,j} = \boldsymbol{0}$**.**
4. Update: $R_{t,j} = R_{t-1,j} + \tilde{\boldsymbol{r}}_{t,j}$ for all j.

Exp3

Theorem. In the **experts** setting with K experts, **MWU** has regret $O(\sqrt{T \cdot \log K})$.

Theorem. In the **bandits** setting with K experts, **Exp3** has regret $O(\sqrt{TK \cdot \log K})$.

Proof for Exp3 is nearly identical to MWU!
(See [Bubeck, Cesa-Bianchi '12] or Lecture 17 in Nick Harvey's CPSC 531H course.)

In the bandits setting, can get $O\left(\sqrt{TK}\right)$ regret and this is optimal [Audibert, Bubeck '10]

Application: Learning Diverse Rankings

Paper is *Learning Diverse Rankings with Multi-Armed Bandits* by Radlinsky, Kleinberg, Joachims (ICML '08)

- Setting is web search
  - A user enters a search query
  - We want to ensure that a relevant document is near the top.

beetle

All · Images · Videos · News · Shopping · More · Settings · Tools

About 277,000,000 results (0.57 seconds)

Beetle - Wikipedia
https://en.wikipedia.org/wiki/Beetle ▾
Beetles are a group of insects that form the order Coleoptera, in the superorder Endopterygota. Their front pair of wings are hardened into wing-cases, elytra, ...
VW Beetle · Hercules beetle · Titan beetle · Meru (beetle)

2019 VW Beetle | Compact Car | Volkswagen Canada
https://www.vwmodels.ca/2019/beetle ▾
The 2019 Volkswagen **Beetle** is one of the most loved compact cars around the world. Discover what makes this iconic bug so unique. Get behind the wheel ...

What are beetles? - Insects in the City
https://citybugs.tamu.edu/factsheets/household/beetles-house/what-are-beetles/ ▾
Beetles are the most common type of insect. **Beetles** are everywhere. But **beetles** can be confused with other kinds of insects, especially some true bugs. So how ...

User may mean the insect or the car and both appear on the top few.

bandits

All · Images · Videos · News · Maps · More · Settings · Tools

About 169,000,000 results (0.74 seconds)

The Bandits - Fraser Valley, British Columbia
https://www.thebandits.ca/
The **Bandits** are a professional basketball team in Fraser Valley, British Columbia. They play within the Canadian Elite Basketball League and tip-off is Summer ...
Schedule · Roster · Tickets · News

Roster - The Bandits Professional Basketball Team
https://www.thebandits.ca/roster
The **Bandits** are a professional basketball team in Fraser Valley, British Columbia. They play within the Canadian Elite Basketball League and tip-off is Summer ...

Fraser Valley Bandits - Wikipedia
https://en.wikipedia.org/wiki/Fraser_Valley_Bandits
The Fraser Valley **Bandits** are a Canadian professional basketball team based in Abbotsford, British Columbia, that is announced to compete in the Canadian ...
History: Fraser Valley Bandits; (2019–) **Arena**: Abbotsford Centre
Leagues: CEBL **Location**: Abbotsford, British Columbia

Fraser Valley Bandits Tickets | Single Game Tickets & Schedule ...
https://www.ticketmaster.ca › Sports Tickets › Basketball
Tickets for Minor League games: buy Fraser Valley **Bandits** Minor League single game tickets at Ticketmaster.ca. Find game schedules and team promotions.

Thu., Jul. 11	Fraser Valley Bandits vs ...	Abbotsford Centre ...
Thu., Jul. 18	Fraser Valley Bandits vs ...	Abbotsford Centre ...
Sat., Jul. 20	Niagara River Lions vs ...	Meridian Centre, St ...

An example of a search which is not diverse at all.

Those searching for bandit algorithms would not click.

Application: Learning Diverse Rankings

- Setting is web search
  - A user enters a search query
  - We want to ensure that a relevant document is near the top.

- Model this as follows.
  - Let $\mathcal{D}$ be a set of documents (for one fixed query).
  - A user u_t comes with some "type" which is a prob. vector p_t indicating probability of clicking a specific document.
  - If user clicks, get reward of 1; if user leaves, get reward of 0.
  - Goal: Maximize number of user clicks.

- Note that offline problem is NP-hard (for one user, we need to solve max coverage problem); but we can get $(1 - 1/e)$-approximation.

Ranked Explore and Commit

Algorithm 1 Ranked Explore and Commit

input: Documents $(d_1, .., d_n)$, parameters ϵ, δ, k.
$x \leftarrow \lceil 2k^2/\epsilon^2 \log(2k/\delta) \rceil$
$(b_1, \ldots, b_k) \leftarrow k$ arbitrary documents.
for i=1 … k **do** *At every rank*
    $\forall j.\ p_j \leftarrow 0$
    for counter=1 … x **do** *Loop x times*
        for j=1 … n **do** *over every document d_j*
            $b_i \leftarrow d_j$
            display $\{b_1, \ldots, b_k\}$ to user; record clicks
            if user clicked on b_i **then** $p_j \leftarrow p_j + 1$
        end for
    end for
    $j^* \leftarrow \operatorname{argmax}_j p_j$ *Commit to best document at this rank*
    $b_i \leftarrow d_{j^*}$
end for

- Model users as static identities.

- Start at the first rank (top of we page).
- Try every possible document for that position a bunch of times.
- Whichever document has the most hits at that rank is chosen to be in that rank.
- Repeat this for every rank.

Ranked Explore and Commit

Theorem. With a suitable choice of parameters, payoff for ranked ETC after T rounds is at least $\left(1-\frac{1}{e}\right) \cdot OPT - O_{n,k}(T^{2/3})$.

Optimal payoff on offline setting.

If $OPT \geq \Omega(T)$ (i.e. a constant fraction of users want to click on some document) then ranked ETC is competitive with optimal offline algorithm.

Ranked Bandits Algorithm

Algorithm 2 Ranked Bandits Algorithm

initialize $\mathsf{MAB}_1(n), \ldots, \mathsf{MAB}_k(n)$ *Initialize MABs*
for t = 1 ... T **do**
  for i = 1 ... k **do** *Sequentially select documents*
    $\hat{b}_i(t) \leftarrow$ select-arm (MAB_i)
    if $\hat{b}_i(t) \in \{b_1(t), .., b_{i-1}(t)\}$ **then** *Replace repeats*
      $b_i(t) \leftarrow$ arbitrary unselected document
    else
      $b_i(t) \leftarrow \hat{b}_i(t)$
    end if
  end for
  display $\{b_1(t), \ldots, b_k(t)\}$ to user; record clicks
  for i = 1 ... k **do** *Determine feedback for* MAB_i
    if user clicked $b_i(t)$ **and** $\hat{b}_i(t) = b_i(t)$ **then**
      $f_{it} = 1$
    else
      $f_{it} = 0$
    end if
    update $(\mathsf{MAB}_i, arm = \hat{b}_i(t), reward = f_{it})$
  end for
end for

- Here, users can change over time.
- Instantiate a multi-armed bandit algorithm for each rank.
- For each rank, we ask algorithm for a document.
- Bandit corresponding to rank r gets reward 1 if page is clicked; else gets zero.
- Note that the MAB algorithm can be arbitrary.

Ranked Bandits Algorithm

Theorem. With a suitable choice of parameters, payoff for ranked bandits after T rounds is at least $\left(1 - \frac{1}{e}\right) \cdot OPT - k \cdot R(T)$, where $R(T)$ is regret of MAB algorithm.

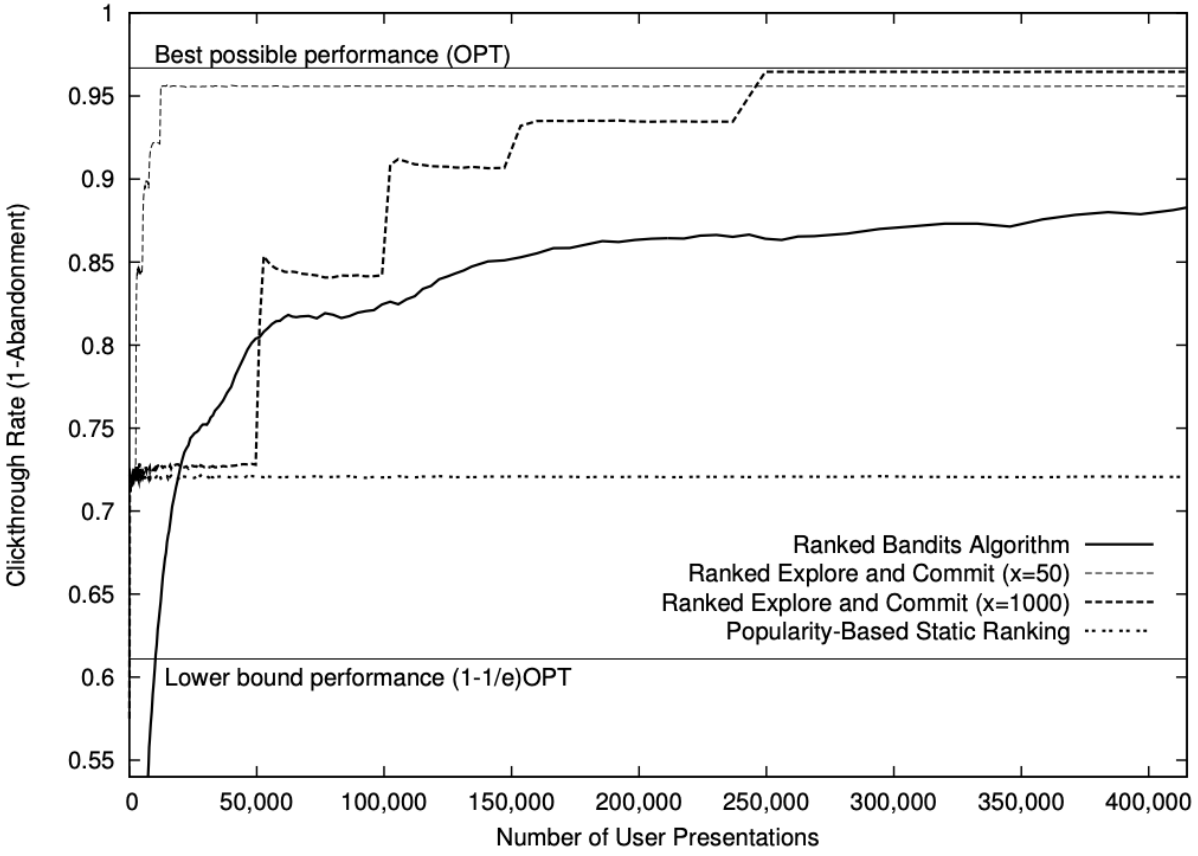
Number of slots (documents to show)

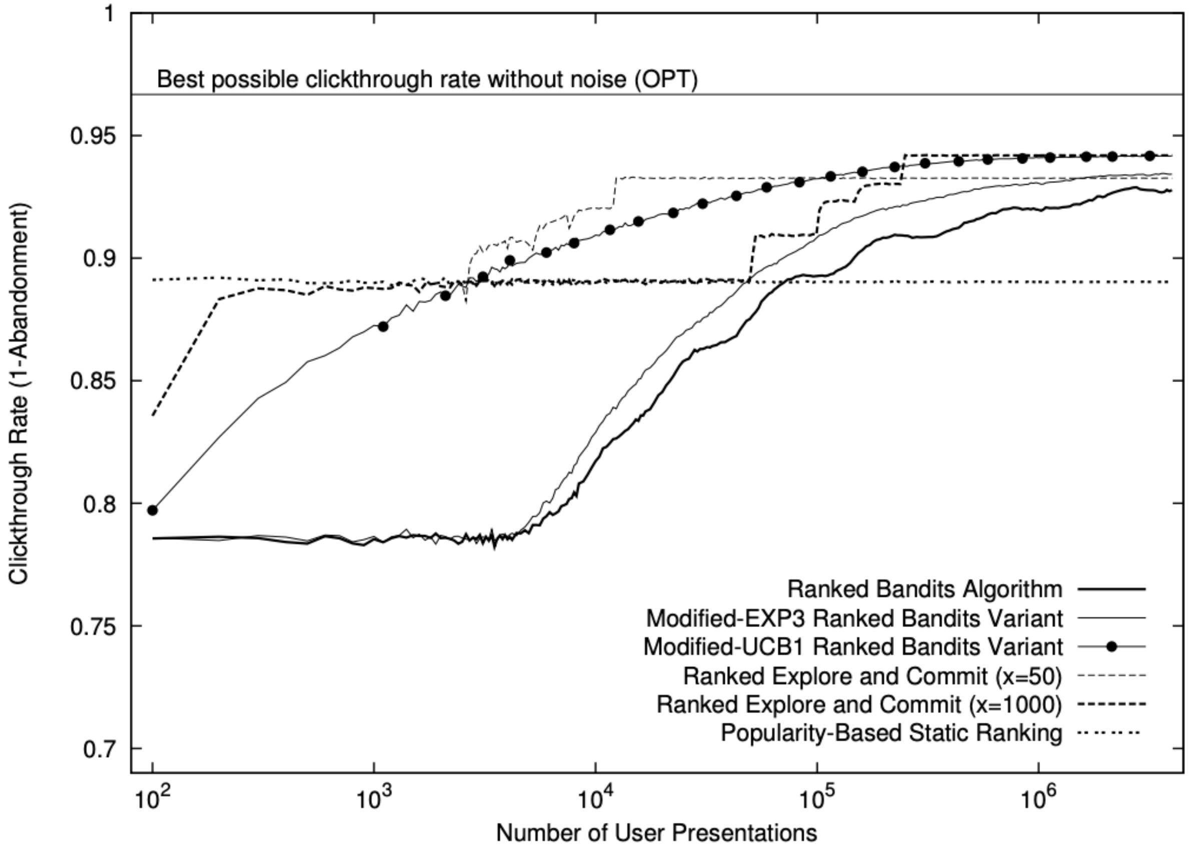
Optimal payoff in offline setting.

For e.g., if we use Exp3 then $R(T) = \tilde{O}_{n,k}\left(T^{1/2}\right)$.

If $OPT \geq \Omega(T)$ (i.e. a constant fraction of users want to click on some document) then ranked bandits is competitive with optimal offline algorithm.

Empirical Results





All their experiments were with synthetic data.

Recap

- We introduced stochastic bandits problem and saw two algorithms:
- **Explore-then-commit** which has an initial exploration stage and then commits for the rest of time
- Upper-confidence bound algorithm which maintains a confidence interval for each arm and uses the upper-confidence as a proxy.

- We introduced adversarial bandits and saw the Exp3 algorithm.

- Some references:
  - Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems by Bubeck and Cesa-Bianchi
  - Introduction to Online Convex Optimization by Hazan
  - Introduction to Online Optimization by Bubeck
  - Bandit Algorithms by Lattimore and Szepesvári
  - Introduction to Multi-Armed Bandits by Slivkins