The Multi-Arm Bandit Framework

A. LAZARIC (*SequeL Team @INRIA-Lille*)
ENS Cachan - Master 2 MVA

SequeL – INRIA Lille

The Exploration-Exploitation Dilemma

The Exploration-Exploitation Dilemma

Tools

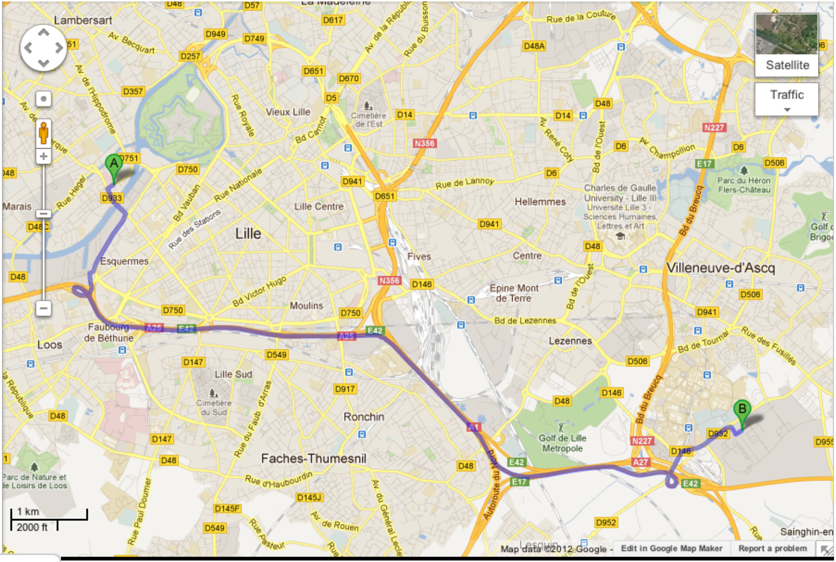
Stochastic Multi-Armed Bandit

Contextual Linear Bandit

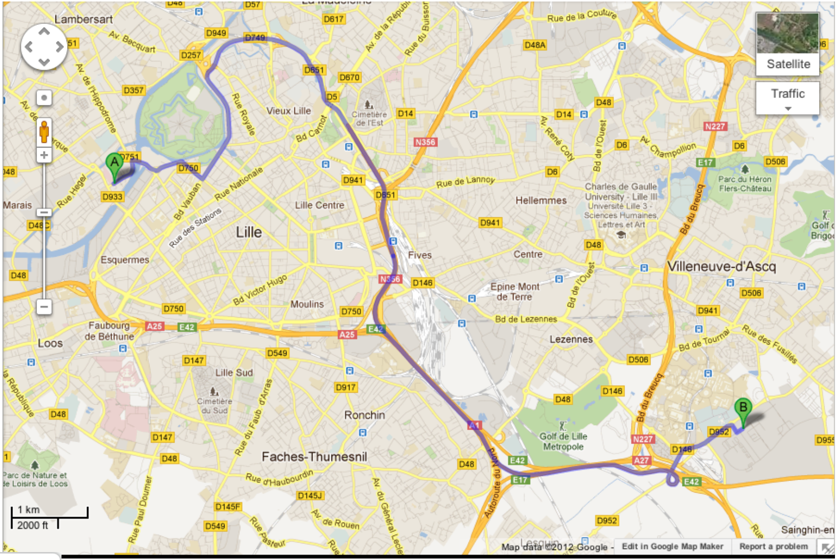
Adversarial Multi-Armed Bandit

Other Multi-Armed Bandit Problems

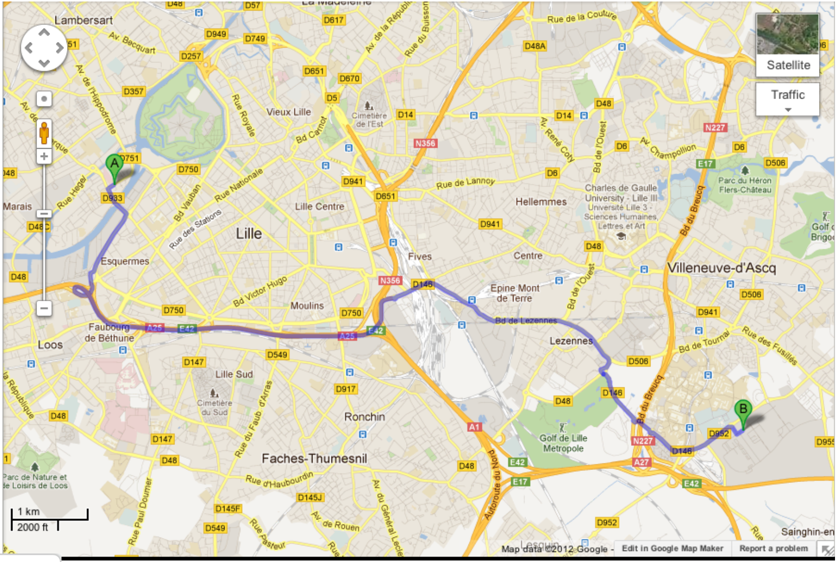
The Navigation Problem



The Navigation Problem



The Navigation Problem



The Navigation Problem

Question: which route should we take?

The Navigation Problem

Question: which route should we take?

Problem: each day we obtain a ***limited feedback***: traveling time of the *chosen route*

The Navigation Problem

Question: which route should we take?

Problem: each day we obtain a ***limited feedback***: traveling time of the *chosen route*

Results: if we do not repeatedly try different options we cannot learn.

The Navigation Problem

Question: which route should we take?

Problem: each day we obtain a ***limited feedback***: traveling time of the *chosen route*

Results: if we do not repeatedly try different options we cannot learn.

Solution: trade off between *optimization* and *learning*.

Learning the Optimal Policy

For $i = 1, \ldots, n$

1. Set $t = 0$
2. Set initial state x_0
3. **While** (x_t not terminal)

   3.1 Take action a_t ***according to a suitable exploration policy***

   3.2 Observe next state x_{t+1} and reward r_t

   3.3 Compute the temporal difference δ_t (e.g., Q-learning)

   3.4 Update the Q-function

$$\widehat{Q}(x_t, a_t) = \widehat{Q}(x_t, a_t) + \alpha(x_t, a_t)\delta_t$$

   3.5 Set $t = t + 1$

   EndWhile

EndFor

Learning the Optimal Policy

For $i = 1, \dots, n$

1. Set $t = 0$
2. Set initial state x_0
3. **While** (x_t not terminal)
   - 3.1 ***Take action*** $a_t = \arg\max_a Q(x_t, a)$
   - 3.2 Observe next state x_{t+1} and reward r_t
   - 3.3 Compute the temporal difference δ_t (e.g., Q-learning)
   - 3.4 Update the Q-function

$$\widehat{Q}(x_t, a_t) = \widehat{Q}(x_t, a_t) + \alpha(x_t, a_t)\delta_t$$

   - 3.5 Set $t = t + 1$

   EndWhile

EndFor

Learning the Optimal Policy

For $i = 1, \dots, n$

1. Set $t = 0$
2. Set initial state x_0
3. **While** (x_t not terminal)

    3.1 ***Take action*** $a_t = \arg\max_a Q(x_t, a)$
    3.2 Observe next state x_{t+1} and reward r_t
    3.3 Compute the temporal difference δ_t (e.g., Q-learning)
    3.4 Update the Q-function

$$\widehat{Q}(x_t, a_t) = \widehat{Q}(x_t, a_t) + \alpha(x_t, a_t)\delta_t$$

    3.5 Set $t = t + 1$

   EndWhile

EndFor

$\Rightarrow$ ***no convergence***

Learning the Optimal Policy

For $i = 1, \dots, n$

1. Set $t = 0$
2. Set initial state x_0
3. **While** (x_t not terminal)
    3.1 ***Take action*** $a_t \sim \mathcal{U}(A)$
    3.2 Observe next state x_{t+1} and reward r_t
    3.3 Compute the temporal difference δ_t (e.g., Q-learning)
    3.4 Update the Q-function

$$\widehat{Q}(x_t, a_t) = \widehat{Q}(x_t, a_t) + \alpha(x_t, a_t)\delta_t$$

    3.5 Set $t = t + 1$

    EndWhile

EndFor

Learning the Optimal Policy

For $i = 1, \ldots, n$

1. Set $t = 0$
2. Set initial state x_0
3. **While** (x_t not terminal)
   3.1 ***Take action*** $a_t \sim \mathcal{U}(A)$
   3.2 Observe next state x_{t+1} and reward r_t
   3.3 Compute the temporal difference δ_t (e.g., Q-learning)
   3.4 Update the Q-function
   $$\widehat{Q}(x_t, a_t) = \widehat{Q}(x_t, a_t) + \alpha(x_t, a_t)\delta_t$$
   3.5 Set $t = t + 1$

   EndWhile

EndFor

$\Rightarrow$ ***very poor rewards***

The Exploration-Exploitation Dilemma

Tools

Contextual Linear Bandit

Stochastic Multi-Armed Bandit

Adversarial Multi-Armed Bandit

Other Multi-Armed Bandit Problems

Concentration Inequalities

Proposition (Chernoff-Hoeffding Inequality)

Let $X_i \in [a_i, b_i]$ be n *independent* r.v. with mean $\mu_i = \mathbb{E}X_i$. Then

$$\mathbb{P}\Big[\Big|\sum_{i=1}^{n}(X_i - \mu_i)\Big| \geq \epsilon\Big] \leq 2\exp\Big(-\frac{2\epsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\Big).$$

Concentration Inequalities

Proof.

$$
\begin{aligned}
\mathbb{P}\Big(\sum_{i=1}^n X_i - \mu_i \geq \epsilon\Big) &= \mathbb{P}(e^{s\sum_{i=1}^n X_i - \mu_i} \geq e^{s\epsilon}) \\
&\leq e^{-s\epsilon}\mathbb{E}[e^{s\sum_{i=1}^n X_i - \mu_i}], \quad \text{Markov inequality} \\
&= e^{-s\epsilon}\prod_{i=1}^n \mathbb{E}[e^{s(X_i - \mu_i)}], \quad \text{independent random variables} \\
&\leq e^{-s\epsilon}\prod_{i=1}^n e^{s^2(b_i - a_i)^2/8}, \quad \text{Hoeffding inequality} \\
&= e^{-s\epsilon + s^2\sum_{i=1}^n (b_i - a_i)^2/8}
\end{aligned}
$$

If we choose $s = 4\epsilon/\sum_{i=1}^n (b_i - a_i)^2$, the result follows.
Similar arguments hold for $\mathbb{P}\big(\sum_{i=1}^n X_i - \mu_i \leq -\epsilon\big)$.

Concentration Inequalities

Finite sample guarantee:

$$\mathbb{P}\Bigg[\underbrace{\Big|\frac{1}{n}\sum_{t=1}^{n} X_t - \mathbb{E}[X_1]\Big|}_{\text{deviation}} > \underbrace{\epsilon}_{\text{accuracy}}\Bigg] \leq \underbrace{2\exp\Big(-\frac{2n\epsilon^2}{(b-a)^2}\Big)}_{\text{confidence}}$$

Concentration Inequalities

Finite sample guarantee:

$$\mathbb{P}\left[\left|\frac{1}{n}\sum_{t=1}^{n} X_t - \mathbb{E}[X_1]\right| > (b-a)\sqrt{\frac{\log 2/\delta}{2n}}\right] \leq \delta$$

Concentration Inequalities

Finite sample guarantee:

$$\mathbb{P}\left[\left|\frac{1}{n}\sum_{t=1}^{n} X_t - \mathbb{E}[X_1]\right| > \epsilon\right] \leq \delta$$

if $n \geq \frac{(b-a)^2 \log 2/\delta}{2\epsilon^2}$.

The Exploration-Exploitation Dilemma

Tools

Stochastic Multi-Armed Bandit

Contextual Linear Bandit

Adversarial Multi-Armed Bandit

Other Multi-Armed Bandit Problems

Reducing RL down to Multi-Armed Bandit

Definition (Markov decision process)

A ***Markov decision process*** *is defined as a tuple* $M = (X, A, p, r)$:

- ~~X *is the state space,*~~
- A *is the action space,*
- ~~$p(y|x, a)$ *is the transition probability*~~
- ~~$r(x, a, y)$ *is the reward of transition* (x, a, y)~~
  $\Rightarrow r(a)$ *is the reward of action* a

Notice

For coherence with the bandit literature we use the notation

- $i = 1, \ldots, K$ set of possible actions
- $t = 1, \ldots, n$ time
- I_t action selected at time t
- $X_{i,t}$ reward for action i at time t

Learning the Optimal Policy

Objective: learn the optimal policy π^* ***as efficiently as possible***

Learning the Optimal Policy

Objective: learn the optimal policy π^* ***as efficiently as possible***

For $t = 1, \ldots, n$

1. ~~Set $t = 0$~~
2. ~~Set initial state x_0~~
3. ~~**While** (x_t not terminal)~~
   3.1 Take action a_t
   3.2 Observe ~~next state x_{t+1} and~~ reward r_t
   3.3 ~~Set $t = t + 1$~~

   ~~**EndWhile**~~

EndFor

The Multi–armed Bandit Protocol

The learner has $i = 1, \ldots, K$ arms (actions)

At each round $t = 1, \ldots, n$

The Multi–armed Bandit Protocol

The learner has $i = 1, \dots, K$ arms (actions)

At each round $t = 1, \dots, n$

- At the same time

The Multi–armed Bandit Protocol

The learner has $i = 1, \ldots, K$ arms (actions)

At each round $t = 1, \ldots, n$

- At the same time
  - The environment chooses a vector of *rewards* $\{X_{i,t}\}_{i=1}^{K}$
  - The learner chooses an arm I_t

The Multi–armed Bandit Protocol

The learner has $i = 1, \ldots, K$ arms (actions)

At each round $t = 1, \ldots, n$

- At the same time
  - The environment chooses a vector of *rewards* $\{X_{i,t}\}_{i=1}^K$
  - The learner chooses an arm I_t
- The learner receives a reward $X_{I_t,t}$

The Multi–armed Bandit Protocol

The learner has $i = 1, \ldots, K$ arms (actions)

At each round $t = 1, \ldots, n$

- At the same time
  - The environment chooses a vector of *rewards* $\{X_{i,t}\}_{i=1}^{K}$
  - The learner chooses an arm I_t
- The learner receives a reward $X_{I_t,t}$
- The environment ***does not*** reveal the rewards of the other arms

The Multi–armed Bandit Game (cont'd)

The regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The Multi–armed Bandit Game (cont'd)

The regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The expectation summarizes any possible source of randomness (either in X or in the algorithm)

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
$\Rightarrow$ the learner should *gain information* by repeatedly pulling all the arms

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
$\Rightarrow$ the learner should *gain information* by repeatedly pulling all the arms

Problem 2: Whenever the learner pulls a ***bad arm***, it suffers some regret

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
$\Rightarrow$ the learner should *gain information* by repeatedly pulling all the arms

Problem 2: Whenever the learner pulls a ***bad arm***, it suffers some regret
$\Rightarrow$ the learner should *reduce the regret* by repeatedly pulling the best arm

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
$\Rightarrow$ the learner should *gain information* by repeatedly pulling all the arms

Problem 2: Whenever the learner pulls a ***bad arm***, it suffers some regret
$\Rightarrow$ the learner should *reduce the regret* by repeatedly pulling the best arm

Challenge: The learner should solve two opposite problems!

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
$\Rightarrow$ the learner should *gain information* by repeatedly pulling all the arms
$\Rightarrow$ ***exploration***

Problem 2: Whenever the learner pulls a ***bad arm***, it suffers some regret
$\Rightarrow$ the learner should *reduce the regret* by repeatedly pulling the best arm

Challenge: The learner should solve two opposite problems!

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
$\Rightarrow$ the learner should *gain information* by repeatedly pulling all the arms
$\Rightarrow$ ***exploration***

Problem 2: Whenever the learner pulls a ***bad arm***, it suffers some regret
$\Rightarrow$ the learner should *reduce the regret* by repeatedly pulling the best arm
$\Rightarrow$ ***exploitation***

Challenge: The learner should solve two opposite problems!

The Exploration–Exploitation Lemma

Problem 1: The environment ***does not*** reveal the rewards of the arms not pulled by the learner
⇒ the learner should *gain information* by repeatedly pulling all the arms
⇒ ***exploration***

Problem 2: Whenever the learner pulls a ***bad arm***, it suffers some regret
⇒ the learner should *reduce the regret* by repeatedly pulling the best arm
⇒ ***exploitation***

Challenge: The learner should solve the *exploration-exploitation* dilemma!

The Multi–armed Bandit Game (cont'd)

Examples

- Packet routing
- Clinical trials
- Web advertising
- Computer games
- Resource mining
- ...

The Stochastic Multi–armed Bandit Problem

Definition

The environment is stochastic

- *Each arm has a distribution ν_i bounded in $[0, 1]$ and characterized by an expected value μ_i*
- *The rewards are i.i.d. $X_{i,t} \sim \nu_i$ (as in the MDP model)*

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} (n\mu_i) - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} (n\mu_i) - \sum_{i=1}^{K} \mathbb{E}[T_{i,n}]\mu_i$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = n\mu_{i^*} - \sum_{i=1}^{K} \mathbb{E}[T_{i,n}]\mu_i$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = \sum_{i \neq i^*} \mathbb{E}[T_{i,n}](\mu_{i^*} - \mu_i)$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = \sum_{i \neq i^*} \mathbb{E}[T_{i,n}]\Delta_i$$

The Stochastic Multi–armed Bandit Problem (cont'd)

Notation

- Number of times arm i has been pulled after n rounds

$$T_{i,n} = \sum_{t=1}^{n} \mathbb{I}\{I_t = i\}$$

- Regret

$$R_n(\mathcal{A}) = \sum_{i \neq i^*} \mathbb{E}[T_{i,n}] \Delta_i$$

- Gap $\Delta_i = \mu_{i^*} - \mu_i$

The Stochastic Multi–armed Bandit Problem (cont'd)

$$R_n(\mathcal{A}) = \sum_{i \neq i^*} \mathbb{E}[T_{i,n}]\Delta_i$$

$\Rightarrow$ we only need to study the *expected number of pulls* of the *suboptimal* arms

The Stochastic Multi–armed Bandit Problem (cont'd)

Optimism in Face of Uncertainty Learning (OFUL)

Whenever we are *uncertain* about the outcome of an arm, we consider the *best possible world* and choose the *best arm*.

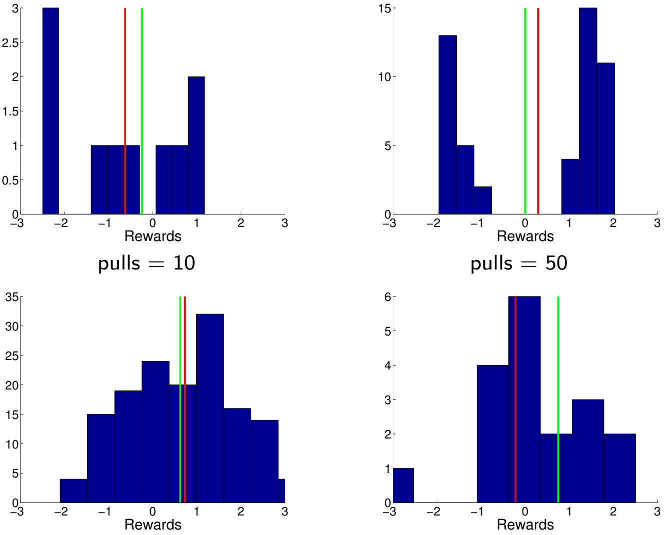
The Stochastic Multi–armed Bandit Problem (cont'd)

Optimism in Face of Uncertainty Learning (OFUL)

Whenever we are *uncertain* about the outcome of an arm, we consider the *best possible world* and choose the *best arm*.
Why it works:

- If the *best possible world* is correct ⇒ *no regret*
- If the *best possible world* is wrong ⇒ *the reduction in the uncertainty is maximized*

The Stochastic Multi–armed Bandit Problem (cont'd)



pulls = 10

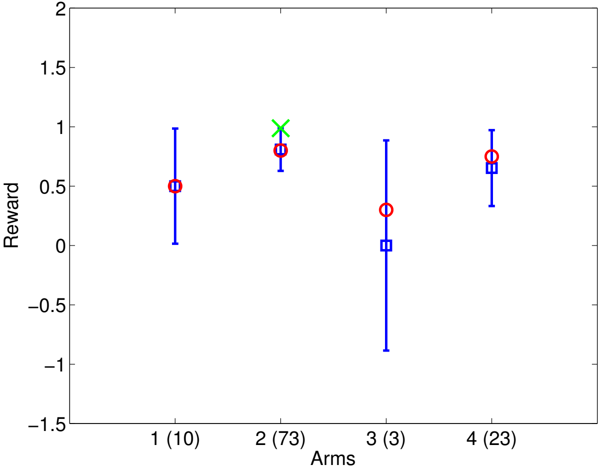
pulls = 50

pulls = 150

pulls = 20

The Upper–Confidence Bound (UCB) Algorithm

The idea



The Upper–Confidence Bound (UCB) Algorithm

Show time!

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

At each round $t = 1, \ldots, n$

- Compute the *score* of each arm i

$$B_i = (\textit{optimistic} \text{ score of arm } i)$$

- Pull arm

$$I_t = \arg\max_{i=1,\ldots,K} B_{i,s,t}$$

- Update the number of pulls $T_{I_t,t} = T_{I_t,t-1} + 1$ and the other statistics

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

The score (with parameters ρ and δ)

$$B_i = (\textit{optimistic} \text{ score of arm } i)$$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

The score (with parameters ρ and δ)

$B_{i,s,t}$ = (*optimistic* score of arm i if pulled s times up to round t)

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

The score (with parameters ρ and δ)

$B_{i,s,t} =$ (*optimistic* score of arm i if pulled s times up to round t)

Optimism in face of uncertainty:
Current knowledge: average rewards $\hat{\mu}_{i,s}$
Current uncertainty: number of pulls s

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

The score (with parameters ρ and δ)

$$B_{i,s,t} = \text{knowledge} \underbrace{+}_{\textit{optimism}} \text{uncertainty}$$

Optimism in face of uncertainty:
Current knowledge: average rewards $\hat{\mu}_{i,s}$
Current uncertainty: number of pulls s

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

The score (with parameters ρ and δ)

$$B_{i,s,t} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log 1/\delta}{2s}}$$

Optimism in face of uncertainty:
Current knowledge: average rewards $\hat{\mu}_{i,s}$
Current uncertainty: number of pulls s

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

At each round $t = 1, \ldots, n$

- Compute the *score* of each arm i

$$B_{i,t} = \hat{\mu}_{i,T_{i,t}} + \rho\sqrt{\frac{\log(t)}{2T_{i,t}}}$$

- Pull arm

$$I_t = \arg\max_{i=1,\ldots,K} B_{i,t}$$

- Update the number of pulls $T_{I_t,t} = T_{I_t,t-1} + 1$ and $\hat{\mu}_{i,T_{i,t}}$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

Theorem

Let $X_1, \ldots, X_n$ be i.i.d. samples from a distribution bounded in $[a, b]$, then for any $\delta \in (0, 1)$

$$\mathbb{P}\left[\left|\frac{1}{n}\sum_{t=1}^{n} X_t - \mathbb{E}[X_1]\right| > (b-a)\sqrt{\frac{\log 2/\delta}{2n}}\right] \leq \delta$$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

After s pulls, arm i

$$\mathbb{P}\left[\mathbb{E}[X_i] \leq \frac{1}{s}\sum_{t=1}^{s} X_{i,t} + \sqrt{\frac{\log 1/\delta}{2s}}\right] \geq 1-\delta$$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

After s pulls, arm i

$$\mathbb{P}\left[\mu_i \leq \hat{\mu}_{i,s} + \sqrt{\frac{\log 1/\delta}{2s}}\right] \geq 1 - \delta$$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

After s pulls, arm i

$$\mathbb{P}\left[\mu_i \leq \hat{\mu}_{i,s} + \sqrt{\frac{\log 1/\delta}{2s}}\right] \geq 1 - \delta$$

$\Rightarrow$ UCB uses an *upper confidence bound* on the expectation

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

Theorem

For any set of K arms with distributions bounded in $[0, b]$, if $\delta = 1/t$, then UCB(ρ) with $\rho > 1$, achieves a regret

$$R_n(\mathcal{A}) \leq \sum_{i \neq i^*} \left[\frac{4b^2}{\Delta_i} \rho \log(n) + \Delta_i \left(\frac{3}{2} + \frac{1}{2(\rho - 1)} \right) \right]$$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

Let $K = 2$ with $i^* = 1$

$$R_n(\mathcal{A}) \leq O\left(\frac{1}{\Delta}\rho \log(n)\right)$$

Remark 1: the *cumulative* regret slowly increases as $\log(n)$

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

Let $K = 2$ with $i^* = 1$

$$R_n(\mathcal{A}) \leq O\left(\frac{1}{\Delta}\rho \log(n)\right)$$

Remark 1: the *cumulative* regret slowly increases as $\log(n)$
Remark 2: the *smaller the gap* the *bigger the regret*... why?

The Upper–Confidence Bound (UCB) Algorithm (cont'd)

Show time (again)!

The Worst–case Performance

Remark: the regret bound is *distribution–dependent*

$$R_n(\mathcal{A}; \Delta) \leq O\left(\frac{1}{\Delta}\rho \log(n)\right)$$

The Worst–case Performance

Remark: the regret bound is *distribution–dependent*

$$R_n(\mathcal{A}; \Delta) \leq O\left(\frac{1}{\Delta}\rho \log(n)\right)$$

Meaning: the algorithm is able to *adapt to the specific problem* at hand!

The Worst–case Performance

Remark: the regret bound is *distribution–dependent*

$$R_n(\mathcal{A}; \Delta) \leq O\left(\frac{1}{\Delta}\rho \log(n)\right)$$

Meaning: the algorithm is able to *adapt to the specific problem* at hand!

Worst–case performance: what is the distribution which leads to the worst possible performance of UCB? what is the distribution–free performance of UCB?

$$R_n(\mathcal{A}) = \sup_{\Delta} R_n(\mathcal{A}; \Delta)$$

The Worst–case Performance

Problem: it seems like if $\Delta \rightarrow 0$ then the regret tends to infinity...

The Worst–case Performance

Problem: it seems like if $\Delta \to 0$ then the regret tends to infinity...
... nosense because the regret is defined as

$$R_n(\mathcal{A}; \Delta) = \mathbb{E}[T_{2,n}]\Delta$$

The Worst–case Performance

Problem: it seems like if $\Delta \to 0$ then the regret tends to infinity...
... nosense because the regret is defined as

$$R_n(\mathcal{A}; \Delta) = \mathbb{E}[T_{2,n}]\Delta$$

then if Δ_i is small, the regret is also small...

The Worst–case Performance

Problem: it seems like if $\Delta \to 0$ then the regret tends to infinity...
... nosense because the regret is defined as

$$R_n(\mathcal{A};\Delta) = \mathbb{E}[T_{2,n}]\Delta$$

then if Δ_i is small, the regret is also small...
In fact

$$R_n(\mathcal{A};\Delta) = \min\left\{ O\left(\frac{1}{\Delta}\rho\log(n)\right), \mathbb{E}[T_{2,n}]\Delta \right\}$$

The Worst–case Performance

Then

$$R_n(\mathcal{A}) = \sup_{\Delta} R_n(\mathcal{A};\Delta) = \sup_{\Delta} \min\left\{ O\left(\frac{1}{\Delta}\rho \log(n)\right), n\Delta \right\} \approx \sqrt{n}$$

for $\Delta = \sqrt{1/n}$

Tuning the confidence δ of UCB

Remark: UCB is an *anytime* algorithm ($\delta = 1/t$)

$$B_{i,s,t} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log t}{2s}}$$

Tuning the confidence δ of UCB

Remark: UCB is an *anytime* algorithm ($\delta = 1/t$)

$$B_{i,s,t} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log t}{2s}}$$

Remark: If the time horizon n is known then the optimal choice is $\delta = 1/n$

$$B_{i,s,t} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log n}{2s}}$$

Tuning the confidence δ of UCB (cont'd)

Intuition: UCB should pull the suboptimal arms

- *Enough*: so as to understand which arm is the best
- *Not too much*: so as to keep the regret as small as possible

Tuning the confidence δ of UCB (cont'd)

Intuition: UCB should pull the suboptimal arms

- *Enough*: so as to understand which arm is the best
- *Not too much*: so as to keep the regret as small as possible

The confidence $1-\delta$ has the following impact (similar for ρ)

- *Big* $1-\delta$: high level of *exploration*
- *Small* $1-\delta$: high level of *exploitation*

Tuning the confidence δ of UCB (cont'd)

Intuition: UCB should pull the suboptimal arms

- *Enough*: so as to understand which arm is the best
- *Not too much*: so as to keep the regret as small as possible

The confidence $1-\delta$ has the following impact (similar for ρ)

- *Big* $1-\delta$: high level of *exploration*
- *Small* $1-\delta$: high level of *exploitation*

Solution: depending on the time horizon, we can tune how to trade-off between exploration and exploitation

UCB Proof

Let's dig into the (1 page and half!!) proof.

Define the (high-probability) event *[statistics]*

$$\mathcal{E} = \left\{ \forall i, s \;\; \left| \hat{\mu}_{i,s} - \mu_i \right| \leq \sqrt{\frac{\log 1/\delta}{2s}} \right\}$$

By Chernoff-Hoeffding $\mathbb{P}[\mathcal{E}] \geq 1 - nK\delta$.

UCB Proof

Let's dig into the (1 page and half!!) proof.

Define the (high-probability) event *[statistics]*

$$\mathcal{E} = \left\{ \forall i, s \ \left| \hat{\mu}_{i,s} - \mu_i \right| \le \sqrt{\frac{\log 1/\delta}{2s}} \right\}$$

By Chernoff-Hoeffding $\mathbb{P}[\mathcal{E}] \ge 1 - nK\delta$.
At time t we pull arm i *[algorithm]*

$$B_{i,T_{i,t-1}} \ge B_{i^*,T_{i^*,t-1}}$$

UCB Proof

Let's dig into the (1 page and half!!) proof.

Define the (high-probability) event *[statistics]*

$$\mathcal{E} = \left\{ \forall i, s \ \ \left| \hat{\mu}_{i,s} - \mu_i \right| \leq \sqrt{\frac{\log 1/\delta}{2s}} \right\}$$

By Chernoff-Hoeffding $\mathbb{P}[\mathcal{E}] \geq 1 - nK\delta$.
At time t we pull arm i *[algorithm]*

$$\hat{\mu}_{i,T_{i,t-1}} + \sqrt{\frac{\log 1/\delta}{2T_{i,t-1}}} \geq \hat{\mu}_{i^*,T_{i^*,t-1}} + \sqrt{\frac{\log 1/\delta}{2T_{i^*,t-1}}}$$

UCB Proof

Let's dig into the (1 page and half!!) proof.

Define the (high-probability) event *[statistics]*

$$\mathcal{E} = \left\{ \forall i, s \ \ \left| \hat{\mu}_{i,s} - \mu_i \right| \leq \sqrt{\frac{\log 1/\delta}{2s}} \right\}$$

By Chernoff-Hoeffding $\mathbb{P}[\mathcal{E}] \geq 1 - nK\delta$.
At time t we pull arm i *[algorithm]*

$$\hat{\mu}_{i,T_{i,t-1}} + \sqrt{\frac{\log 1/\delta}{2T_{i,t-1}}} \geq \hat{\mu}_{i^*,T_{i^*,t-1}} + \sqrt{\frac{\log 1/\delta}{2T_{i^*,t-1}}}$$

On the event $\mathcal{E}$ we have *[math]*

$$\mu_i + 2\sqrt{\frac{\log 1/\delta}{2T_{i,t-1}}} \geq \mu_{i^*}$$

UCB Proof (cont'd)

Assume t is the last time i is pulled, then $T_{i,n} = T_{i,t-1} + 1$, thus

$$\mu_i + 2\sqrt{\frac{\log 1/\delta}{2(T_{i,n} - 1)}} \geq \mu_{i^*}$$

UCB Proof (cont'd)

Assume t is the last time i is pulled, then $T_{i,n} = T_{i,t-1} + 1$, thus

$$\mu_i + 2\sqrt{\frac{\log 1/\delta}{2(T_{i,n} - 1)}} \geq \mu_{i^*}$$

Reordering *[math]*

$$T_{i,n} \leq \frac{\log 1/\delta}{2\Delta_i^2} + 1$$

under event $\mathcal{E}$ and thus with probability $1 - nK\delta$.

UCB Proof (cont'd)

Assume t is the last time i is pulled, then $T_{i,n} = T_{i,t-1} + 1$, thus

$$\mu_i + 2\sqrt{\frac{\log 1/\delta}{2(T_{i,n} - 1)}} \geq \mu_{i^*}$$

Reordering *[math]*

$$T_{i,n} \leq \frac{\log 1/\delta}{2\Delta_i^2} + 1$$

under event $\mathcal{E}$ and thus with probability $1 - nK\delta$.
Moving to the expectation *[statistics]*

$$\mathbb{E}[T_{i,n}] = \mathbb{E}[T_{i,n}\mathbb{I}\mathcal{E}] + \mathbb{E}[T_{i,n}\mathbb{I}\mathcal{E}^C]$$

UCB Proof (cont'd)

Assume t is the last time i is pulled, then $T_{i,n} = T_{i,t-1} + 1$, thus

$$\mu_i + 2\sqrt{\frac{\log 1/\delta}{2(T_{i,n} - 1)}} \geq \mu_{i^*}$$

Reordering *[math]*

$$T_{i,n} \leq \frac{\log 1/\delta}{2\Delta_i^2} + 1$$

under event $\mathcal{E}$ and thus with probability $1 - nK\delta$.
Moving to the expectation *[statistics]*

$$\mathbb{E}[T_{i,n}] \leq \frac{\log 1/\delta}{2\Delta_i^2} + 1 + n(nK\delta)$$

UCB Proof (cont'd)

Assume t is the last time i is pulled, then $T_{i,n} = T_{i,t-1} + 1$, thus

$$\mu_i + 2\sqrt{\frac{\log 1/\delta}{2(T_{i,n} - 1)}} \geq \mu_{i^*}$$

Reordering *[math]*

$$T_{i,n} \leq \frac{\log 1/\delta}{2\Delta_i^2} + 1$$

under event $\mathcal{E}$ and thus with probability $1 - nK\delta$.
Moving to the expectation *[statistics]*

$$\mathbb{E}[T_{i,n}] \leq \frac{\log 1/\delta}{2\Delta_i^2} + 1 + n(nK\delta)$$

Trading-off the two terms $\delta = 1/n^2$, we obtain

$$\hat{\mu}_{i,T_{i,t-1}} + \sqrt{\frac{2\log n}{2T_{i,t-1}}}$$

UCB Proof (cont'd)

Trading-off the two terms $\delta = 1/n^2$, we obtain

$$\hat{\mu}_{i,T_{i,t-1}} + \sqrt{\frac{2\log n}{2T_{i,t-1}}}$$

and

$$\mathbb{E}[T_{i,n}] \leq \frac{\log n}{\Delta_i^2} + 1 + K$$

Tuning the confidence δ of UCB (cont'd)

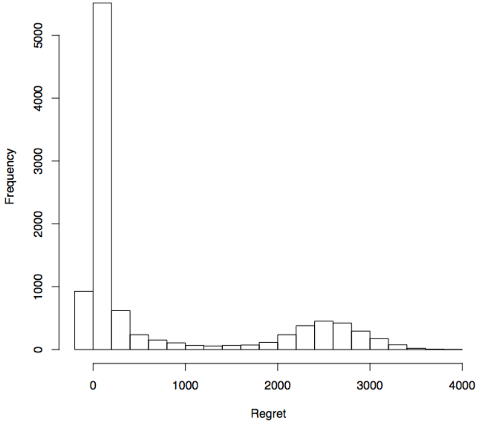
Multi–armed Bandit: the same for $\delta = 1/t$ and $\delta = 1/n$...

Tuning the confidence δ of UCB (cont'd)

Multi–armed Bandit: the same for $\delta = 1/t$ and $\delta = 1/n$...
... **almost** (i.e., in expectation)

Tuning the confidence δ of UCB (cont'd)

The value–at–risk of the regret for UCB-anytime



Tuning the ρ of UCB (cont'd)

UCB values (for the $\delta = 1/n$ algorithm)

$$B_{i,s} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log n}{2s}}$$

Tuning the ρ of UCB (cont'd)

UCB values (for the $\delta = 1/n$ algorithm)

$$B_{i,s} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log n}{2s}}$$

Theory

- $\rho < 0.5$, polynomial regret w.r.t. n
- $\rho > 0.5$, logarithmic regret w.r.t. n

Tuning the ρ of UCB (cont'd)

UCB values (for the $\delta = 1/n$ algorithm)

$$B_{i,s} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log n}{2s}}$$

Theory

- $\rho < 0.5$, polynomial regret w.r.t. n
- $\rho > 0.5$, logarithmic regret w.r.t. n

Practice: $\rho = 0.2$ is often the best choice

Tuning the ρ of UCB (cont'd)

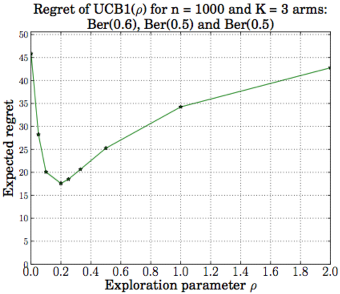
UCB values (for the $\delta = 1/n$ algorithm)

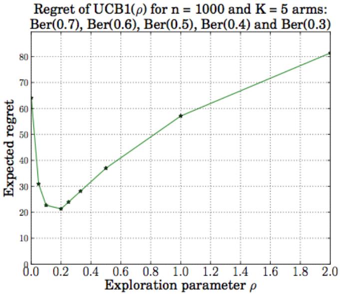
$$B_{i,s} = \hat{\mu}_{i,s} + \rho\sqrt{\frac{\log n}{2s}}$$

Theory

- $\rho < 0.5$, polynomial regret w.r.t. n
- $\rho > 0.5$, logarithmic regret w.r.t. n

Practice: $\rho = 0.2$ is often the best choice





Improvements: UCB-V

Idea: use *empirical Bernstein bounds* for more accurate c.i.

Improvements: UCB-V

Idea: use *empirical Bernstein bounds* for more accurate c.i.

Algorithm

- Compute the *score* of each arm i

$$B_{i,t} = \hat{\mu}_{i,T_{i,t}} + \rho\sqrt{\frac{\log(t)}{2T_{i,t}}}$$

- Pull arm

$$I_t = \arg\max_{i=1,\dots,K} B_{i,t}$$

- Update the number of pulls $T_{I_t,t}$, $\hat{\mu}_{i,T_{i,t}}$

Improvements: UCB-V

Idea: use *empirical Bernstein bounds* for more accurate c.i.

Algorithm

- Compute the *score* of each arm i

$$B_{i,t} = \hat{\mu}_{i,T_{i,t}} + \sqrt{\frac{2\hat{\sigma}^2_{i,T_{i,t}} \log t}{T_{i,t}}} + \frac{8 \log t}{3 T_{i,t}}$$

- Pull arm

$$I_t = \arg \max_{i=1,\dots,K} B_{i,t}$$

- Update the number of pulls $T_{I_t,t}$, $\hat{\mu}_{i,T_{i,t}}$ and $\hat{\sigma}^2_{i,T_{i,t}}$

Improvements: UCB-V

Idea: use *empirical Bernstein bounds* for more accurate c.i.

Algorithm

- Compute the *score* of each arm i

$$B_{i,t} = \hat{\mu}_{i,T_{i,t}} + \sqrt{\frac{2\hat{\sigma}^2_{i,T_{i,t}} \log t}{T_{i,t}}} + \frac{8 \log t}{3 T_{i,t}}$$

- Pull arm

$$I_t = \arg \max_{i=1,\dots,K} B_{i,t}$$

- Update the number of pulls $T_{I_t,t}$, $\hat{\mu}_{i,T_{i,t}}$ and $\hat{\sigma}^2_{i,T_{i,t}}$

Regret

$$R_n \leq O\Big(\frac{1}{\Delta} \log n\Big)$$

Improvements: UCB-V

Idea: use *empirical Bernstein bounds* for more accurate c.i.

Algorithm

- Compute the *score* of each arm i

$$B_{i,t} = \hat{\mu}_{i,T_{i,t}} + \sqrt{\frac{2\hat{\sigma}^2_{i,T_{i,t}} \log t}{T_{i,t}}} + \frac{8 \log t}{3 T_{i,t}}$$

- Pull arm

$$I_t = \arg \max_{i=1,\dots,K} B_{i,t}$$

- Update the number of pulls $T_{I_t,t}$, $\hat{\mu}_{i,T_{i,t}}$ and $\hat{\sigma}^2_{i,T_{i,t}}$

Regret

$$R_n \leq O\Big(\frac{\sigma^2}{\Delta} \log n\Big)$$

Improvements: KL-UCB

Idea: use even tighter c.i. based on *Kullback–Leibler divergence*

$$d(p, q) = p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$$

Improvements: KL-UCB

Idea: use even tighter c.i. based on *Kullback–Leibler divergence*

$$d(p,q) = p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$$

Algorithm: Compute the *score* of each arm i (convex optimization)

$$B_{i,t} = \max \Big\{ q \in [0,1] : T_{i,t} d\big(\hat{\mu}_{i,T_{i,t}}, q\big) \leq \log(t) + c \log(\log(t)) \Big\}$$

Improvements: KL-UCB

Idea: use even tighter c.i. based on *Kullback–Leibler divergence*

$$d(p,q) = p\log\frac{p}{q} + (1-p)\log\frac{1-p}{1-q}$$

Algorithm: Compute the *score* of each arm i (convex optimization)

$$B_{i,t} = \max\Big\{q \in [0,1] : T_{i,t}\, d\big(\hat{\mu}_{i,T_{i,t}}, q\big) \le \log(t) + c\log(\log(t))\Big\}$$

Regret: pulls to suboptimal arms

$$\mathbb{E}\big[T_{i,n}\big] \le (1+\epsilon)\frac{\log(n)}{d(\mu_i,\mu^*)} + C_1\log(\log(n)) + \frac{C_2(\epsilon)}{n^{\beta(\epsilon)}}$$

where $d(\mu_i,\mu^*) > 2\Delta_i^2$

Improvements: Thompson strategy

Idea: Use a Bayesian approach to estimate the means $\{\mu_i\}_i$

Improvements: Thompson strategy

Idea: Use a Bayesian approach to estimate the means $\{\mu_i\}_i$

Algorithm: Assuming Bernoulli arms and a *Beta* prior on the mean

- Compute
$$\mathcal{D}_{i,t} = \text{Beta}(S_{i,t} + 1, F_{i,t} + 1)$$
- Draw a mean sample as
$$\widetilde{\mu}_{i,t} \sim \mathcal{D}_{i,t}$$
- Pull arm
$$I_t = \arg\max \widetilde{\mu}_{i,t}$$
- If $X_{I_t,t} = 1$ update $S_{I_t,t+1} = S_{I_t,t} + 1$, else update $F_{I_t,t+1} = F_{I_t,t} + 1$

Regret:

$$\lim_{n\to\infty} \frac{R_n}{\log(n)} = \sum_{i=1}^{K} \frac{\Delta_i}{d(\mu_i, \mu^*)}$$

The Lower Bound

Theorem

For any stochastic bandit $\{\nu_i\}$, any algorithm $\mathcal{A}$ has a regret

$$\lim_{n\to\infty} \frac{R_n}{\log n} \geq \frac{\Delta_i}{\inf_\nu KL(\nu_i, \nu)}$$

The Lower Bound

Theorem

For any stochastic bandit $\{\nu_i\}$, any algorithm $\mathcal{A}$ has a regret

$$\lim_{n\to\infty} \frac{R_n}{\log n} \geq \frac{\Delta_i}{\inf_\nu KL(\nu_i, \nu)}$$

Problem: this is just asymptotic

The Lower Bound

Theorem

For any stochastic bandit $\{\nu_i\}$, any algorithm $\mathcal{A}$ has a regret

$$\lim_{n\to\infty} \frac{R_n}{\log n} \geq \frac{\Delta_i}{\inf_\nu KL(\nu_i, \nu)}$$

Problem: this is just asymptotic

Open Question: what is the finite-time lower bound?

The Exploration-Exploitation Dilemma

The Contextual Linear Bandit Problem

Motivating Examples

- Different users may have different preferences
- The set of available news may change over time
- We want to minimise the regret w.r.t. the best news for each user

The Contextual Linear Bandit Problem

The problem: at each time $t = 1, \ldots, n$

- User u_t arrives and a set of news $\mathcal{A}_t$ is provided
- The user u_t together with a news $a \in \mathcal{A}_t$ are described by a feature vector $x_{t,a}$
- The learner chooses a news a_t and receives a reward r_{t,a_t}

The optimal news: at each time $t = 1, \ldots, n$, the optimal news is

$$a_t^* = \arg\max_{a \in \mathcal{A}_t} \mathbb{E}[r_{t,a}]$$

The regret:

$$R_n = \mathbb{E}\Big[\sum_{t=1}^{n} r_{t,a_t^*}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} r_{t,a_t}\Big]$$

The Contextual Linear Bandit Problem

The linear assumption: the reward is a linear combination between the context and an unknown parameter vector

$$\mathbb{E}[r_{t,a}|x_{t,a}] = x_{t,a}^\top \theta_a$$

The Contextual Linear Bandit Problem

The linear regression estimate:

- $\mathcal{T}_a = \{t : a_t = a\}$
- Construct the design matrix of all the contexts observed when action a has been taken $D_a \in \mathbb{R}^{|\mathcal{T}_a| \times d}$
- Construct the reward vector of all the rewards observed when action a has been taken $c_a \in \mathbb{R}^{|\mathcal{T}_a|}$
- Estimate θ_a as

$$\hat{\theta}_a = (D_a^\top D_a + I)^{-1} D_a^\top c_a$$

The Contextual Linear Bandit Problem

Optimism in face of uncertainty: the LinUCB algorithm

- Chernoff-Hoeffding in this case becomes

$$\left|x_{t,a}^\top \hat{\theta}_a - \mathbb{E}[r_{t,a}|x_{t,a}]\right| \leq \alpha \sqrt{x_{t,a}^\top (D_a^\top D_a + I)^{-1} x_{t,a}}$$

- and the UCB strategy is

$$a_t = \arg\max_{a \in \mathcal{A}_t} x_{t,a}^\top \hat{\theta}_a + \alpha \sqrt{x_{t,a}^\top (D_a^\top D_a + I)^{-1} x_{t,a}}$$

The Contextual Linear Bandit Problem

The evaluation problem

- Online evaluation: too expensive
- Offline evaluation: how to use the logged data?

The Contextual Linear Bandit Problem

Evaluation from logged data

- Assumption 1: contexts and rewards are i.i.d. from a stationary distribution

$$(x_1, \ldots, x_K, r_1, \ldots, r_K) \sim D$$

- Assumption 2: the logging strategy is random

The Contextual Linear Bandit Problem

Evaluation from logged data: given a bandit strategy π, a desired number of samples T, and a (infinite) stream of data

Algorithm 3 Policy_Evaluator.

Inputs: $T > 0$; policy π; stream of events
$h_0 \leftarrow \emptyset$ {An initially empty history}
$R_0 \leftarrow 0$ {An initially zero total payoff}
for $t = 1, 2, 3, \ldots, T$ **do**
  repeat
    Get next event $(\mathbf{x}_1, ..., \mathbf{x}_K, a, r_a)$
  until $\pi(h_{t-1}, (\mathbf{x}_1, ..., \mathbf{x}_K)) = a$
  $h_t \leftarrow$ CONCATENATE$(h_{t-1}, (\mathbf{x}_1, ..., \mathbf{x}_K, a, r_a))$
  $R_t \leftarrow R_{t-1} + r_a$
end for
Output: R_T / T

The Exploration-Exploitation Dilemma

The Non–Stochastic Multi–armed Bandit Problem

Definition

The environment is *adversarial*

- Arms have no fixed distribution
- The rewards $X_{i,t}$ are arbitrarily chosen by the environment

The Non–Stochastic Multi–armed Bandit Problem (cont'd)

The (non–stochastic bandit) regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,N} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The Non–Stochastic Multi–armed Bandit Problem (cont'd)

The (non–stochastic bandit) regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,N} \sum_{t=1}^{n} X_{i,t} - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The Exponentially Weighted Average Forecaster

Initialize the weights $w_{i,0} = 1$

- Compute $(W_{t-1} = \sum_{i=1}^{N} w_{i,t-1})$

$$\hat{p}_{i,t} = \frac{w_{i,t-1}}{W_{t-1}}$$

- Choose the arm at random

$$I_t \sim \hat{\mathbf{p}}_t$$

- Observe the rewards $\{X_{i,t}\}$
- Receive a reward $X_{I_t,t}$
- Update

$$w_{i,t} = w_{i,t-1} \exp\left(+ \eta X_{i_t,t}\right)$$

The Non–Stochastic Multi–armed Bandit Problem (cont'd)

Problem: we only observe the reward of the specific arm chosen at time t!! (i.e., only $X_{I_t,t}$ is observed)

The Exponentially Weighted Average Forecaster

Initialize the weights $w_{i,0} = 1$

- Compute $(W_{t-1} = \sum_{i=1}^{N} w_{i,t-1})$

$$\hat{p}_{i,t} = \frac{w_{i,t-1}}{W_{t-1}}$$

- Choose the arm at random

$$I_t \sim \hat{\mathbf{p}}_t$$

- ~~Observe the rewards $\{X_{i,t}\}$~~
- Receive a reward $X_{I_t,t}$
- Update

$$w_{i,t} = w_{i,t-1} \exp\left(\eta X_{i_t,t}\right) \Rightarrow \text{this update is not possible}$$

The Non–Stochastic Multi–armed Bandit Problem (cont'd)

We use the *importance weight* trick

$$\hat{X}_{i,t} = \begin{cases} \frac{X_{i,t}}{\hat{p}_{i,t}} & \text{if } i = I_t \\ 0 & \text{otherwise} \end{cases}$$

The Non–Stochastic Multi–armed Bandit Problem (cont'd)

We use the *importance weight* trick

$$\hat{X}_{i,t} = \begin{cases} \frac{X_{i,t}}{\hat{p}_{i,t}} & \text{if } i = I_t \\ 0 & \text{otherwise} \end{cases}$$

Why it is a good idea:

$$\mathbb{E}\big[\hat{X}_{i,t}\big] = \frac{X_{i,t}}{\hat{p}_{i,t}}\hat{p}_{i,t} + 0(1 - \hat{p}_{i,t}) = X_{i,t}$$

$\hat{X}_{i,t}$ is an *unbiased* estimator of $X_{i,t}$

The Exp3 Algorithm

Exp3: Exponential-weight algorithm for Exploration and Exploitation

Initialize the weights $w_{i,0} = 1$

- Compute $(W_{t-1} = \sum_{i=1}^{N} w_{i,t-1})$

$$\hat{p}_{i,t} = \frac{w_{i,t-1}}{W_{t-1}}$$

- Choose the arm at random

$$I_t \sim \hat{\mathbf{p}}_t$$

- Receive a reward $X_{I_t,t}$
- Update

$$w_{i,t} = w_{i,t-1} \exp\left(\eta \hat{X}_{i_t,t}\right)$$

The Exp3 Algorithm

Question: is this enough? is this algorithm actually exploring enough?

The Exp3 Algorithm

Question: is this enough? is this algorithm actually exploring enough?
Answer: more or less...

- Exp3 has a small regret *in expectation*
- Exp3 might have large deviations with *high probability* (ie, from time to time it may *concentrate* $\hat{\mathbf{p}}_t$ *on the wrong arm* for too long and then incur a large regret)

The Exp3 Algorithm

Fix: add some extra uniform exploration

Initialize the weights $w_{i,0} = 1$

- Compute $\left(W_{t-1} = \sum_{i=1}^{N} w_{i,t-1}\right)$

$$\hat{p}_{i,t} = (1-\gamma)\frac{w_{i,t-1}}{W_{t-1}} + \frac{\gamma}{K}$$

- Choose the arm at random

$$I_t \sim \hat{\mathbf{p}}_t$$

- Receive a reward $X_{I_t,t}$
- Update

$$w_{i,t} = w_{i,t-1} \exp\left(\eta \hat{X}_{i_t,t}\right)$$

The Exp3 Algorithm

Theorem

If Exp3 is run with $\gamma = \eta$, then it achieves a regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,N} \sum_{t=1}^{n} X_{i,t} - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big] \leq (e-1)\gamma G_{\max} + \frac{N \log N}{\gamma}$$

with $G_{\max} = \max_{i=1,\dots,N} \sum_{t=1}^{n} X_{i,t}$.

The Exp3 Algorithm

Theorem

If Exp3 is run with

$$\gamma = \eta = \sqrt{\frac{N \log N}{(e-1)n}}$$

then it achieves a regret

$$R_n(\mathcal{A}) \leq O(\sqrt{nN \log N})$$

The Exp3 Algorithm

Comparison with online learning

$$R_n(Exp3) \leq O(\sqrt{nN \log N})$$

$$R_n(EWA) \leq O(\sqrt{n \log N})$$

The Exp3 Algorithm

Comparison with online learning

$$R_n(Exp3) \leq O(\sqrt{nN \log N})$$

$$R_n(EWA) \leq O(\sqrt{n \log N})$$

Intuition: in online learning at each round we obtain N feedbacks, while in bandits we receive 1 feedback.

The Improved-Exp3 Algorithm

Initialize the weights $w_{i,0} = 1$

- Compute $\left(W_{t-1} = \sum_{i=1}^{N} w_{i,t-1}\right)$

$$\hat{p}_{i,t} = (1-\gamma)\frac{w_{i,t-1}}{W_{t-1}} + \frac{\gamma}{K}$$

- Choose the arm at random

$$I_t \sim \hat{\mathbf{p}}_t$$

- Receive a reward $X_{I_t,t}$
- Compute

$$\widetilde{X}_{i,t} = \hat{X}_{i,t} + \frac{\beta}{\hat{p}_{i,t}}$$

- Update

$$w_{i,t} = w_{i,t-1} \exp\left(\eta \widetilde{X}_{i_t,t}\right)$$

The Improved-Exp3 Algorithm

Theorem

If Improved-Exp3 is run with parameters in the ranges

$$\gamma \leq \frac{1}{2}; \quad 0 \leq \eta \leq \frac{\gamma}{2N}; \quad \sqrt{\frac{1}{nN}\log\frac{N}{\delta}} \leq \beta \leq 1$$

then it achieves a regret

$$R_n^{HP}(\mathcal{A}) \leq n\big(\gamma + \eta(1+\beta)N\big) + \frac{\log N}{\eta} + 2nN\beta$$

with probability at least $1-\delta$*.*

The Improved-Exp3 Algorithm

Theorem

If Improved-Exp3 is run with parameters in the ranges

$$\beta = \sqrt{\frac{1}{nN}\log\frac{N}{\delta}}; \quad \gamma = \frac{4N\beta}{3+\beta}; \quad \eta = \frac{\gamma}{2N}$$

then it achieves a regret

$$R_n^{HP}(\mathcal{A}) \leq \frac{11}{2}\sqrt{nN\log(N/\delta)} + \frac{\log N}{2}$$

with probability at least $1-\delta$.

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*	*C*
1	*30*, *-30*	*-10*, *10*	*20*, *-20*
2	*10*, *-10*	*-20*, *20*	*-20*, *20*

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*	*C*
1	*30*, *-30*	*-10*, *10*	*20*, *-20*
2	*10*, *-10*	*-20*, *20*	*-20*, *20*

Nash equilibrium:
A set of strategies is a Nash equilibrium if *no player* can do better by *unilaterally changing* his strategy.

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*	*C*
1	*30*, *-30*	*-10*, *10*	*20*, *-20*
2	*10*, *-10*	*-20*, *20*	*-20*, *20*

Nash equilibrium:
Red: take action *1* with *prob. 1*
Blue: take action *B* with *prob. 1*

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*	*C*
1	*30*, *-30*	*-10*, *10*	*20*, *-20*
2	*10*, *-10*	*-20*, *20*	*-20*, *20*

Nash equilibrium:
Value of the game: $V = -10$ (reward of Red at the equilibrium)

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	B
1	-2, 2	3, -3
2	3, -3	-4, 4

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*
1	*-2*, *2*	*3*, *-3*
2	*3*, *-3*	*-4*, *4*

Nash equilibrium:
A set of strategies is a Nash equilibrium if *no player* can do better by *unilaterally changing* his strategy.

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*
1	*-2*, *2*	*3*, *-3*
2	*3*, *-3*	*-4*, *4*

Nash equilibrium:
Red: take action *1* with *prob. 7/12* and action *2* with *prob. 5/12*
Blue: take action *A* with *prob. 7/12* and action *B* with *prob. 5/7*

Repeated Two–Player Zero–Sum Games

A two–player zero–sum game

	A	*B*
1	*-2*, *2*	*3*, *-3*
2	*3*, *-3*	*-4*, *4*

Nash equilibrium:
Value of the game: $V = 1/12$ (reward of Red at the equilibrium)

Repeated Two–Player Zero–Sum Games

At each round t

- Row player computes a mixed strategy $\hat{\mathbf{p}}_t = (\hat{p}_{1,t}, \ldots, \hat{p}_{N,t})$
- Column player computes a mixed strategy $\hat{\mathbf{q}}_t = (\hat{q}_{1,t}, \ldots, \hat{q}_{M,t})$

Repeated Two–Player Zero–Sum Games

At each round t

- Row player computes a mixed strategy $\hat{\mathbf{p}}_t = (\hat{p}_{1,t}, \ldots, \hat{p}_{N,t})$
- Column player computes a mixed strategy $\hat{\mathbf{q}}_t = (\hat{q}_{1,t}, \ldots, \hat{q}_{M,t})$
- Row player selects action $I_t \in \{1, \ldots, N\}$
- Column player selects action $J_t \in \{1, \ldots, M\}$

Repeated Two–Player Zero–Sum Games

At each round t

- Row player computes a mixed strategy $\hat{\mathbf{p}}_t = (\hat{p}_{1,t}, \ldots, \hat{p}_{N,t})$
- Column player computes a mixed strategy $\hat{\mathbf{q}}_t = (\hat{q}_{1,t}, \ldots, \hat{q}_{M,t})$
- Row player selects action $I_t \in \{1, \ldots, N\}$
- Column player selects action $J_t \in \{1, \ldots, M\}$
- Row player suffers $\ell(I_t, J_t)$
- Column player suffers $-\ell(I_t, J_t)$

Repeated Two–Player Zero–Sum Games

At each round t

- Row player computes a mixed strategy $\hat{\mathbf{p}}_t = (\hat{p}_{1,t}, \ldots, \hat{p}_{N,t})$
- Column player computes a mixed strategy $\hat{\mathbf{q}}_t = (\hat{q}_{1,t}, \ldots, \hat{q}_{M,t})$
- Row player selects action $I_t \in \{1, \ldots, N\}$
- Column player selects action $J_t \in \{1, \ldots, M\}$
- Row player suffers $\ell(I_t, J_t)$
- Column player suffers $-\ell(I_t, J_t)$

Value of the game

$$V = \max_{\mathbf{q}} \min_{\mathbf{p}} \overline{\ell}(\mathbf{p}, \mathbf{q})$$

with

$$\overline{\ell}(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{N} \sum_{j=1}^{M} p_i q_j \ell(i, j)$$

Repeated Two–Player Zero–Sum Games

Question: what if the two players are both bandit algorithms (e.g., Exp3)?

Repeated Two–Player Zero–Sum Games

Question: what if the two players are both bandit algorithms (e.g., Exp3)?
Row player: a bandit algorithm is able to minimize

$$R_n(\text{row}) = \sum_{t=1}^{n} \ell_{I_t,J_t} - \min_{i=1,\dots,N} \sum_{t=1}^{n} \ell_{i,J_t}$$

Repeated Two–Player Zero–Sum Games

Question: what if the two players are both bandit algorithms (e.g., Exp3)?

Row player: a bandit algorithm is able to minimize

$$R_n(\text{row}) = \sum_{t=1}^{n} \ell_{I_t,J_t} - \min_{i=1,\dots,N} \sum_{t=1}^{n} \ell_{i,J_t}$$

Col player: a bandit algorithm is able to minimize

$$R_n(\text{col}) = \sum_{t=1}^{n} \ell_{I_t,J_t} - \min_{j=1,\dots,M} \sum_{t=1}^{n} \ell_{I_t,j}$$

Repeated Two–Player Zero–Sum Games

Theorem

If both the row and column players play according to an Hannan-consistent strategy, then

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n} \ell(I_t, J_t) = V$$

Repeated Two–Player Zero–Sum Games

Theorem

The empirical distribution of plays

$$\hat{p}_{i,n} = \frac{1}{n}\sum_{t=1}^{n}\mathbb{I}\{I_t = i\} \quad \hat{q}_{j,n} = \frac{1}{n}\sum_{t=1}^{n}\mathbb{I}\{J_t = j\}$$

induces a product distribution $\hat{\mathbf{p}}_n \times \hat{\mathbf{q}}_n$ *which converges to the set of Nash equilibria* $\mathbf{p} \times \mathbf{q}$.

Repeated Two–Player Zero–Sum Games

Proof idea.

Since $\overline{\ell}(\mathbf{p}, J_t)$ is linear, over the simplex, the minimum is at one of the corners *[math]*

$$\min_{i=1,\dots,N} \frac{1}{N} \sum_{t=1}^{n} \ell(i, J_t) = \min_{\mathbf{p}} \frac{1}{n} \sum_{t=1}^{n} \overline{\ell}(\mathbf{p}, J_t)$$

Repeated Two–Player Zero–Sum Games

Proof idea.

Since $\bar{\ell}(\mathbf{p}, J_t)$ is linear, over the simplex, the minimum is at one of the corners *[math]*

$$\min_{i=1,\ldots,N} \frac{1}{N} \sum_{t=1}^{n} \ell(i, J_t) = \min_{\mathbf{p}} \frac{1}{n} \sum_{t=1}^{n} \bar{\ell}(\mathbf{p}, J_t)$$

We consider the empirical probability of the row player *[def]*

$$\hat{q}_{j,n} = \frac{1}{n} \sum_{t=1}^{n} \mathbb{I} J_t = j$$

Repeated Two–Player Zero–Sum Games

Proof idea.

Since $\bar{\ell}(\mathbf{p}, J_t)$ is linear, over the simplex, the minimum is at one of the corners *[math]*

$$\min_{i=1,\dots,N} \frac{1}{N} \sum_{t=1}^{n} \ell(i, J_t) = \min_{\mathbf{p}} \frac{1}{n} \sum_{t=1}^{n} \bar{\ell}(\mathbf{p}, J_t)$$

We consider the empirical probability of the row player *[def]*

$$\hat{q}_{j,n} = \frac{1}{n} \sum_{t=1}^{n} \mathbb{I} J_t = j$$

Elaborating on it *[math]*

$$\begin{aligned}
\min_{\mathbf{p}} \frac{1}{n} \sum_{t=1}^{n} \bar{\ell}(\mathbf{p}, J_t) &= \min_{\mathbf{p}} \sum_{j=1}^{M} \hat{q}_{j,n} \bar{\ell}(\mathbf{p}, j) \\
&= \min_{\mathbf{p}} \bar{\ell}(\mathbf{p}, \hat{\mathbf{q}}_n) \\
&\leq \max_{\mathbf{q}} \min_{\mathbf{p}} \bar{\ell}(\mathbf{p}, \mathbf{q}) = V
\end{aligned}$$

Repeated Two–Player Zero–Sum Games

Proof idea.
By definition of Hannan's consistent strategy *[def]*

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n} \ell(I_t, J_t) = \min_{i=1,\dots,N} \frac{1}{n}\sum_{t=1}^{n} \ell(i, J_t)$$

Repeated Two–Player Zero–Sum Games

Proof idea.
By definition of Hannan's consistent strategy *[def]*

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n}\ell(I_t, J_t) = \min_{i=1,\ldots,N}\frac{1}{n}\sum_{t=1}^{n}\ell(i, J_t)$$

Then

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n}\ell(I_t, J_t) \le V$$

Repeated Two–Player Zero–Sum Games

Proof idea.
By definition of Hannan's consistent strategy *[def]*

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n} \ell(I_t, J_t) = \min_{i=1,\dots,N} \frac{1}{n}\sum_{t=1}^{n} \ell(i, J_t)$$

Then

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n} \ell(I_t, J_t) \leq V$$

If we do the same for the other player *[zero–sum game]*

$$\limsup_{n\to\infty} \frac{1}{n}\sum_{t=1}^{n} \ell(I_t, J_t) \geq V$$

Repeated Two–Player Zero–Sum Games

Question: how fast do they converge to the Nash equilibrium?

Repeated Two–Player Zero–Sum Games

Question: how fast do they converge to the Nash equilibrium?
Answer: it depends on the specific algorithm. For EWA(η), we now that

$$\sum_{t=1}^{n} \ell(I_t, J_t) - \min_{i=1,\dots,N} \sum_{t=1}^{n} \ell(i, J_t) \leq \frac{\log N}{\eta} + \frac{n\eta}{8} + \sqrt{\frac{n}{2} \log \frac{1}{\delta}}$$

Repeated Two–Player Zero–Sum Games

Generality of the results

- Players do not know the payoff matrix

Repeated Two–Player Zero–Sum Games

Generality of the results

- Players do not know the payoff matrix
- Players do not observe the loss of the other player

Repeated Two–Player Zero–Sum Games

Generality of the results

- Players do not know the payoff matrix
- Players do not observe the loss of the other player
- Players do not even observe the action of the other player

Internal Regret and Correlated Equilibria

External (expected) regret

$$
\begin{aligned}
R_n &= \sum_{t=1}^{n} \bar{\ell}(\hat{\mathbf{p}}_t, y_t) - \min_{i=1,\ldots,N} \sum_{t=1}^{n} \ell(i, y_t) \\
&= \max_{i=1,\ldots,N} \sum_{t=1}^{n} \sum_{j=1}^{N} \hat{p}_{j,t} \big(\ell(j, y_t) - \ell(i, y_t) \big)
\end{aligned}
$$

Internal Regret and Correlated Equilibria

External (expected) regret

$$R_n = \sum_{t=1}^{n} \bar{\ell}(\hat{\mathbf{p}}_t, y_t) - \min_{i=1,\dots,N} \sum_{t=1}^{n} \ell(i, y_t)$$
$$= \max_{i=1,\dots,N} \sum_{t=1}^{n} \sum_{j=1}^{N} \hat{p}_{j,t} \big(\ell(j, y_t) - \ell(i, y_t) \big)$$

Internal (expected) regret

$$R_n^I = \max_{i,j=1,\dots,N} \sum_{t=1}^{n} \hat{p}_{j,t} \big(\ell(i, y_t) - \ell(j, y_t) \big)$$

Internal Regret and Correlated Equilibria

Internal (expected) regret

$$R_n^I = \max_{i,j=1,\dots,N} \sum_{t=1}^{n} \hat{p}_{j,t} \big(\ell(i, y_t) - \ell(j, y_t) \big)$$

Intuition: an algorithm has *small internal regret* if, for each pair of experts (i, j), the learner does not regret of not having followed expert j each time it followed expert i.

Internal Regret and Correlated Equilibria

Theorem

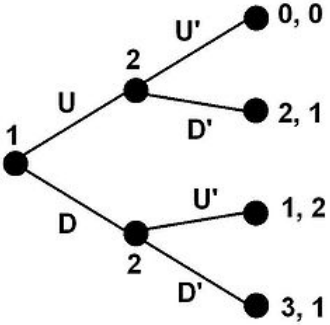
Given a K–person game with a set of correlated equilibria $\mathcal{C}$. If all the players are internal–regret minimizers, then the distance between the empirical distribution of plays and the set of correlated equilibria $\mathcal{C}$ converges to 0.

Nash Equilibria in Extensive Form Games

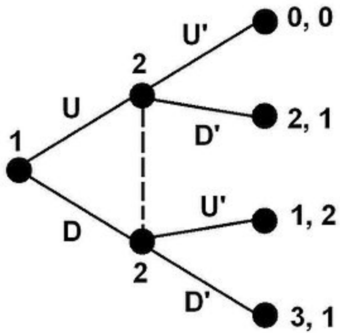
A powerful model for *sequential* games

- Checkers / Chess / Go
- Poker
- Bargaining
- Monitoring
- Patrolling
- ...

Nash Equilibria in Extensive Form Games



Nash Equilibria in Extensive Form Games



Nash Equilibria in Extensive Form Games

A bit of notation

- Set of actions A
- Set of information sets I (roughly equivalent to states)
- Strategy $\sigma_i : I \to \Delta(A)$
- Strategy profile $\sigma = (\sigma_i, \sigma_{-i})$
- History h is a sequence of actions
- $\pi^\sigma(h)$ probability of generating history h
- $\pi^\sigma(I) = \sum_{h \in I} \pi^\sigma(h)$ probability of reaching I
- $\pi^\sigma_{-i}(I)$ counterfactual probability of reaching I "without" i
- Z set of terminal histories

Nash Equilibria in Extensive Form Games

- Counterfactural value

$$v_i(\sigma, h) = \sum_{z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

- Counterfactual regret w.r.t. action a

$$r(h, a) = v_i(\sigma_{I \to a}, h) - v_i(\sigma, h)$$

- Counterfactual regret of information set I w.r.t. action a

$$r(I, a) = \sum_{h \in I} r(h, a)$$

- Cumulative counterfactual regret

$$R_i^T(I, a) = \sum_{t=1}^{T} r_i^t(I, a)$$

Nash Equilibria in Extensive Form Games

Counterfactual regret minimization: at each information set compute

$$\sigma_i^{T+1}(I, a) = \frac{R_i^{T,+}(I, a)}{\sum_{b \in A} R_i^{T,+}(I, b)}$$

Nash Equilibria in Extensive Form Games

Theorem

If player k selects actions according to the counterfactual regret minimization algorithm, then it achieves a regret

$$R_{k,T} \leq \#\, states \sqrt{\frac{\#\, actions}{T}}$$

Nash Equilibria in Extensive Form Games

Theorem

If player k selects actions according to the counterfactual regret minimization algorithm, then it achieves a regret

$$R_{k,T} \leq \#\ states\sqrt{\frac{\#\ actions}{T}}$$

Theorem

In a two–player zero–sum extensive form game, counterfactual regret minimization algorithms achieves an 2ϵ-Nash equilibrium, with

$$\epsilon \leq \#\ states\sqrt{\frac{\#\ actions}{T}}$$

The Exploration-Exploitation Dilemma

Tools

Stochastic Multi-Armed Bandit

Contextual Linear Bandit

Adversarial Multi-Armed Bandit

Other Multi-Armed Bandit Problems

The Best Arm Identification Problem

Motivating Examples

- Find the best shortest path in a limited number of days
- Maximize the confidence about the best treatment after a finite number of patients
- Discover the best advertisements after a training phase
- ...

The Best Arm Identification Problem

Objective: given a fixed budget n, return the best arm $i^* = \arg\max_i \mu_i$ at the end of the experiment

The Best Arm Identification Problem

Objective: given a fixed budget n, return the best arm $i^* = \arg\max_i \mu_i$ at the end of the experiment
Measure of performance: the probability of error

$$\mathbb{P}[J_n \neq i^*] \leq \sum_{i=1}^{N} \exp\big(- T_{i,n}\Delta_i^2\big)$$

The Best Arm Identification Problem

Objective: given a fixed budget n, return the best arm $i^* = \arg\max_i \mu_i$ at the end of the experiment
Measure of performance: the probability of error

$$\mathbb{P}[J_n \neq i^*] \leq \sum_{i=1}^{N} \exp\big(- T_{i,n}\Delta_i^2\big)$$

Algorithm idea: mimic the behavior of the optimal strategy

$$T_{i,n} = \frac{\frac{1}{\Delta_i^2}}{\sum_{j=1}^{N} \frac{1}{\Delta_j^2}} n$$

The Best Arm Identification Problem

The Successive Reject Algorithm

- Divide the budget in $N-1$ phases. Define $(\overline{\log}(N) = 0.5 + \sum_{i=2}^{N} 1/i)$

$$n_k = \frac{1}{\overline{\log} K} \frac{n-N}{N+1-k}$$

The Best Arm Identification Problem

The Successive Reject Algorithm

- Divide the budget in $N-1$ phases. Define $(\overline{\log}(N) = 0.5 + \sum_{i=2}^{N} 1/i)$

$$n_k = \frac{1}{\overline{\log} K} \frac{n-N}{N+1-k}$$

- Set of active arms A_k at phase k ($A_1 = \{1, \ldots, N\}$)

The Best Arm Identification Problem

The Successive Reject Algorithm

- Divide the budget in $N-1$ phases. Define $(\overline{\log}(N) = 0.5 + \sum_{i=2}^{N} 1/i)$

$$n_k = \frac{1}{\overline{\log} K} \frac{n - N}{N + 1 - k}$$

- Set of active arms A_k at phase k ($A_1 = \{1, \ldots, N\}$)
- For each phase $k = 1, \ldots, N-1$
  - For each arm $i \in A_k$, pull arm i for $n_k - n_{k-1}$ rounds

The Best Arm Identification Problem

The Successive Reject Algorithm

- Divide the budget in $N-1$ phases. Define $(\overline{\log}(N) = 0.5 + \sum_{i=2}^{N} 1/i)$

$$n_k = \frac{1}{\overline{\log} K} \frac{n - N}{N + 1 - k}$$

- Set of active arms A_k at phase k ($A_1 = \{1, \dots, N\}$)
- For each phase $k = 1, \dots, N-1$
  - For each arm $i \in A_k$, pull arm i for $n_k - n_{k-1}$ rounds
  - Remove the worst arm

$$A_{k+1} = A_k \setminus \arg\min_{i \in A_k} \hat{\mu}_{i,n_k}$$

The Best Arm Identification Problem

The Successive Reject Algorithm

- Divide the budget in $N-1$ phases. Define $(\overline{\log}(N) = 0.5 + \sum_{i=2}^{N} 1/i)$

$$n_k = \frac{1}{\overline{\log} K} \frac{n-N}{N+1-k}$$

- Set of active arms A_k at phase k ($A_1 = \{1, \ldots, N\}$)
- For each phase $k = 1, \ldots, N-1$
  - For each arm $i \in A_k$, pull arm i for $n_k - n_{k-1}$ rounds
  - Remove the worst arm

$$A_{k+1} = A_k \setminus \arg\min_{i \in A_k} \hat{\mu}_{i,n_k}$$

- Return the only remaining arm $J_n = A_N$

The Best Arm Identification Problem

The Successive Reject Algorithm

Theorem

The successive reject algorithm have a probability of doing a mistake of

$$\mathbb{P}[J_n \neq i^*] \leq \frac{K(K-1)}{2} \exp\Big(- \frac{n-N}{\overline{\log} N H_2} \Big)$$

with $H_2 = \max_{i=1,\ldots,N} i \Delta_{(i)}^{-2}$.

The Best Arm Identification Problem

The UCB-E Algorithm

- Define an exploration parameter a
- Compute

$$B_{i,s} = \hat{\mu}_{i,s} + \sqrt{\frac{a}{s}}$$

The Best Arm Identification Problem

The UCB-E Algorithm

- Define an exploration parameter a
- Compute

$$B_{i,s} = \hat{\mu}_{i,s} + \sqrt{\frac{a}{s}}$$

- Select

$$I_t = \arg\max_{B_{i,s}}$$

The Best Arm Identification Problem

The UCB-E Algorithm

- Define an exploration parameter a
- Compute

$$B_{i,s} = \hat{\mu}_{i,s} + \sqrt{\frac{a}{s}}$$

- Select

$$I_t = \arg\max_{B_{i,s}}$$

- At the end return

$$J_n = \arg\max_i \hat{\mu}_{i,T_{i,n}}$$

The Best Arm Identification Problem

The UCB-E Algorithm

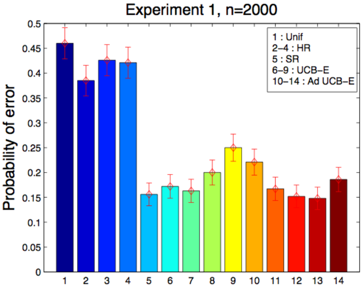
Theorem

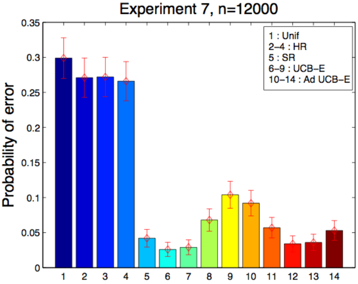
The UCB-E algorithm with $a = \frac{25}{36}\frac{n-N}{H_1}$ *has a probability of doing a mistake of*

$$\mathbb{P}[J_n \neq i^*] \leq 2nN \exp\Big(-\frac{2a}{25}\Big)$$

with $H_1 = \sum_{i=1}^N 1/\Delta_i^2$.

The Best Arm Identification Problem





The Active Bandit Problem

Motivating Examples

- N production lines
- The test of the performance of a line is expensive
- We want an accurate estimation of the performance of each production line

The Active Bandit Problem

Objective: given a fixed budget n, return the an estimate of the means $\hat{\mu}_{i,t}$ which is as accurate as possible for all the arms

The Active Bandit Problem

Objective: given a fixed budget n, return the an estimate of the means $\hat{\mu}_{i,t}$ which is as accurate as possible for all the arms

Notice: Given an arm has a mean μ_i and a variance σ_i^2, if it is pulled $T_{i,n}$ times, then

$$L_{i,n} = \mathbb{E}\big[(\hat{\mu}_{i,T_{i,n}} - \mu_i)^2\big] = \frac{\sigma_i^2}{T_{i,n}}$$

The Active Bandit Problem

Objective: given a fixed budget n, return the an estimate of the means $\hat{\mu}_{i,t}$ which is as accurate as possible for all the arms

Notice: Given an arm has a mean μ_i and a variance σ_i^2, if it is pulled $T_{i,n}$ times, then

$$L_{i,n} = \mathbb{E}\big[(\hat{\mu}_{i,T_{i,n}} - \mu_i)^2\big] = \frac{\sigma_i^2}{T_{i,n}}$$

$$L_n = \max_i L_{i,n}$$

The Active Bandit Problem

Problem: what are the number of pulls $(T_{1,n}, \ldots, T_{N,n})$ (such that $\sum T_{i,n} = n$) which minimizes the loss?

$$(T^*_{1,n}, \ldots, T^*_{N,n}) = \arg \min_{(T_{1,n}, \ldots, T_{N,n})} L_n$$

The Active Bandit Problem

Problem: what are the number of pulls $(T_{1,n}, \ldots, T_{N,n})$ (such that $\sum T_{i,n} = n$) which minimizes the loss?

$$(T^*_{1,n}, \ldots, T^*_{N,n}) = \arg\min_{(T_{1,n},\ldots,T_{N,n})} L_n$$

Answer

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{j=1}^N \sigma_j^2} n$$

The Active Bandit Problem

Problem: what are the number of pulls $(T_{1,n},\ldots,T_{N,n})$ (such that $\sum T_{i,n}=n$) which minimizes the loss?

$$(T^*_{1,n},\ldots,T^*_{N,n}) = \arg\min_{(T_{1,n},\ldots,T_{N,n})} L_n$$

Answer

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{j=1}^N \sigma_j^2} n$$

$$L^*_n = \frac{\sum_{i=1}^N \sigma_i^2}{n} = \frac{\Sigma}{n}$$

The Active Bandit Problem

Objective: given a fixed budget n, return the an estimate of the means $\hat{\mu}_{i,t}$ which is as accurate as possible for all the arms

The Active Bandit Problem

Objective: given a fixed budget n, return the an estimate of the means $\hat{\mu}_{i,t}$ which is as accurate as possible for all the arms

Measure of performance: the regret on the quadratic error

$$R_n(\mathcal{A}) = \max_i L_n(\mathcal{A}) - \frac{\sum_{i=1}^N \sigma_i^2}{n}$$

The Active Bandit Problem

Objective: given a fixed budget n, return the an estimate of the means $\hat{\mu}_{i,t}$ which is as accurate as possible for all the arms
Measure of performance: the regret on the quadratic error

$$R_n(\mathcal{A}) = \max_i L_n(\mathcal{A}) - \frac{\sum_{i=1}^N \sigma_i^2}{n}$$

Algorithm idea: mimic the behavior of the optimal strategy

$$T_{i,n} = \frac{\sigma_i^2}{\sum_{j=1}^N \sigma_j^2} n = \lambda_i n$$

The Active Bandit Problem

An UCB–based strategy

At each time step $t = 1, \dots, n$

- Estimate

$$\hat{\sigma}^2_{i,T_{i,t-1}} = \frac{1}{T_{i,t-1}} \sum_{s=1}^{T_{i,t-1}} X^2_{s,i} - \hat{\mu}^2_{i,T_{i,t-1}}$$

- Compute

$$B_{i,t} = \frac{1}{T_{i,t-1}} \left(\hat{\sigma}^2_{i,T_{i,t-1}} + 5\sqrt{\frac{\log 1/\delta}{2 T_{i,t-1}}} \right)$$

- Pull arm

$$I_t = \arg\max B_{i,t}$$

The Active Bandit Problem

Theorem

The UCB–based algorithm achieves a regret

$$R_n(\mathcal{A}) \leq \frac{98 \log(n)}{n^{3/2} \lambda_{\min}^{5/2}} + O\left(\frac{\log n}{n^2}\right)$$

The Active Bandit Problem

Theorem

The UCB–based algorithm achieves a regret

$$R_n(\mathcal{A}) \leq \frac{98 \log(n)}{n^{3/2} \lambda_{\min}^{5/2}} + O\left(\frac{\log n}{n^2}\right)$$

The Exploration-Exploitation Dilemma

Tools

Stochastic Multi-Armed Bandit

Contextual Linear Bandit

Adversarial Multi-Armed Bandit

Other Multi-Armed Bandit Problems **Bonus: Reinforcement Learning**

Learning the Optimal Policy

For $i = 1, \dots, n$

1. Set $t = 0$
2. Set initial state x_0
3. **While** (x_t not terminal)
   - 3.1 Take action a_t ***according to a suitable exploration policy***
   - 3.2 Observe next state x_{t+1} and reward r_t
   - 3.3 Compute the temporal difference δ_t (e.g., Q-learning)
   - 3.4 Update the Q-function

$$\widehat{Q}(x_t, a_t) = \widehat{Q}(x_t, a_t) + \alpha(x_t, a_t)\delta_t$$

   - 3.5 Set $t = t + 1$

   EndWhile

EndFor

Learning the Optimal Policy

The regret in MAB

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

Learning the Optimal Policy

The regret in MAB

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

$$\Rightarrow R_n(\mathcal{A}) = \max_{\pi} \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, \pi(x_t))\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, a_t)\Big]$$

Learning the Optimal Policy

The regret in MAB

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

$$\Rightarrow R_n(\mathcal{A}) = \max_{\pi} \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, \pi(x_t))\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, a_t)\Big]$$

⇒ ***not correct***: actions influence the state as well!

Learning the Optimal Policy

The regret in MAB

$$R_n(\mathcal{A}) = \max_{i=1,\dots,K} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

$$\Rightarrow R_n(\mathcal{A}) = \max_{\pi} \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, \pi(x_t))\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, a_t)\Big]$$

$\Rightarrow$ ***not correct***: actions influence the state as well!

The regret in RL

$$R_n(\mathcal{A}) = \max_{\pi} \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t^*, \pi(x_t^*))\Big] - \mathbb{E}\Big[\sum_{t=1}^{n} r(x_t, a_t)\Big],$$

$$x_t^* \sim p\big(\cdot\,|x_{t-1}^*, \pi^*(x_{t-1}^*)\big)$$

Learning the Optimal Policy

Idea: can we adapt UCB (that already works in MAB, contextual bandit) here?

Learning the Optimal Policy

Idea: can we adapt UCB (that already works in MAB, contextual bandit) here? ***Yes!***

The Setting

- Infinite horizon average reward
- Average reward

$$\rho(\pi; M) = \lim_{n\to\infty} \mathbb{E}\Big[\frac{1}{n}\sum_{t=1}^{n} r(x_t, \pi(x_t))\Big]$$

- Optimal policy $\pi^* = \arg\max_\pi \rho(\pi; M)$
- Regret

$$R_n = n\rho(\pi^*; M) - \sum_{t=1}^{n} r(x_t, a_t)$$

The UCRL2 Algorithm

Initialize episode k

1. Current time t_k
2. Let $N_k(x,a) = \big|\{\tau < t_k : x_t = x, a_t = a\}\big|$
3. Let $R_k(x,a) = \sum_{t=1}^{t_k} r_t \mathbb{I}\{x_t = x, a_t = a\}$
4. Let $P_k(x,a,x') = \big|\{\tau < t_k : x_t = x, a_t = a, x_{t+1} = x'\}\big|$
5. Compute $\hat{r}_k(x,a) = R_k(x,a)/N_k(x,a)$, $\hat{p}_k(x,a,x') = P_k(x,a,x')/N_k(x,a)$

Compute optimistic policy π_k

1. Let

$$\mathcal{M}_k = \Big\{\widetilde{M} : |\tilde{r}(x,a) - \hat{r}_k(x,a)| < 1/\sqrt{N_k(x,a)};$$
$$|\tilde{p}(x,a,x') - \hat{p}_k(x,a,x')| < 1/\sqrt{N_k(x,a)}\Big\}$$

2. Compute $\pi_k = \arg\max_\pi \max_{\tilde{M}\in\mathcal{M}_k} \rho(\pi;\tilde{M})$

Execute π_k until at least one state-action space counter is doubled

The Regret

Theorem

UCRL2 run in an MDP with diameter D, X states and A actions suffers a regret

$$R_n = O(DX\sqrt{An})$$

where $D = \max_{x,x'} \min_\pi \mathbb{E}[T_\pi(x, x')]$.

Bibliography I

Reinforcement Learning

Alessandro Lazaric
alessandro.lazaric@inria.fr
sequel.lille.inria.fr