Recent Advancements in Multi–armed Bandits

From Clinical Trials to Web Advertising

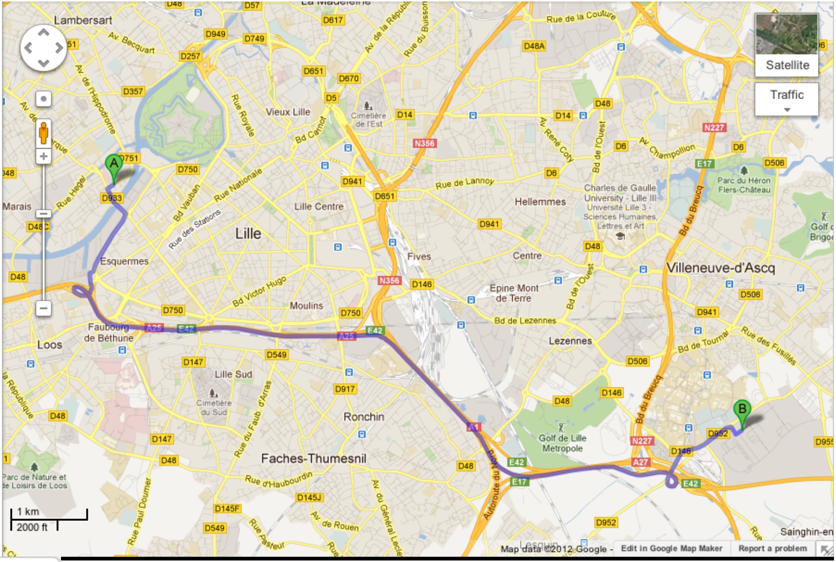
A. LAZARIC (*INRIA-Lille*)

DEI, Politecnico di Milano

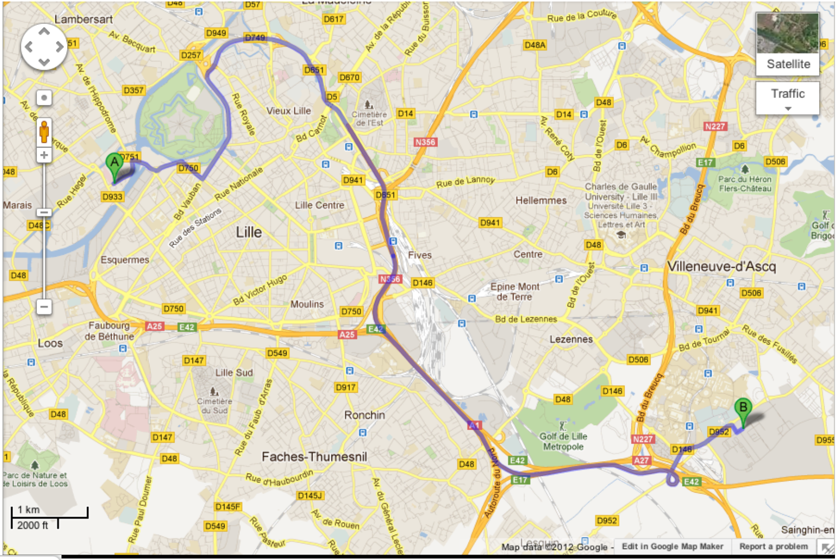
SequeL – INRIA Lille

April 13, 2012

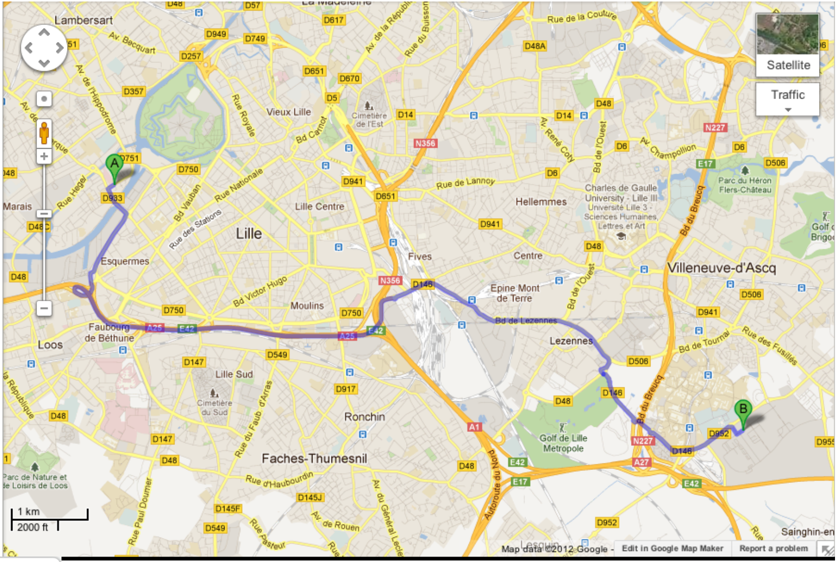
A Motivating Example



A Motivating Example



A Motivating Example



A Motivating Example

Question: which route should we take?

A Motivating Example

Question: which route should we take?

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

A Motivating Example

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.

A Motivating Example

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*.

Outline

Outline

The Multi–armed Bandit Game

The learner has $i = 1, \ldots, N$ arms (links, treatments, routes, ...)

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

The Multi–armed Bandit Game

The learner has $i = 1, \ldots, N$ arms (links, treatments, routes, ...)

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

- The learner chooses an arm I_t

The Multi–armed Bandit Game

The learner has $i = 1, \dots, N$ arms (links, treatments, routes, ...)

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

- The learner chooses an arm I_t
- The learner receives a reward $X_{I_t,t} \sim \nu_i$ (with mean μ_i)

The Multi–armed Bandit Game

The learner has $i = 1, \dots, N$ arms (links, treatments, routes, ...)

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

- The learner chooses an arm I_t
- The learner receives a reward $X_{I_t,t} \sim \nu_i$ (with mean μ_i)
- The learner ***does not*** know the reward of the other arms

The Multi–armed Bandit Problem (cont'd)

The regret

$$R_n(\mathcal{A}) = \text{perf. best possible arm} - \text{perf. algorithm } \mathcal{A}$$

The Multi–armed Bandit Problem (cont'd)

The 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 Multi–armed Bandit Problem (cont'd)

The regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,N} \underbrace{\mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big]}_{\text{cumul. exp. reward of arm } i} - \mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]$$

The Multi–armed Bandit Problem (cont'd)

The regret

$$R_n(\mathcal{A}) = \max_{i=1,\dots,N} \mathbb{E}\Big[\sum_{t=1}^{n} X_{i,t}\Big] - \underbrace{\mathbb{E}\Big[\sum_{t=1}^{n} X_{I_t,t}\Big]}_{\text{cumul. exp. reward of } \mathcal{A}}$$

The Multi–armed Bandit Problem (cont'd)

The regret in the stochastic bandit

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

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

The Multi–armed Bandit Problem (cont'd)

The regret in the stochastic bandit

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

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

- 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 Multi–armed Bandit Problem (cont'd)

The regret in the stochastic bandit

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

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

- Regret

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

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

The Multi–armed Bandit Problem (cont'd)

The regret in the stochastic bandit

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

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

- Regret

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

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

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

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
⇒ 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 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
$\Rightarrow$ 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 the *exploration-exploitation* dilemma!

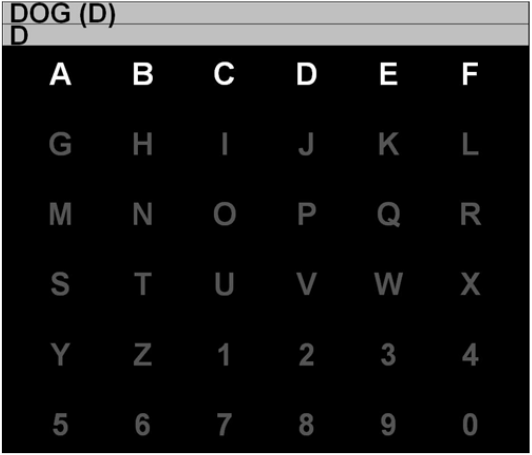
The Multi–armed Bandit Game (cont'd)

Examples

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

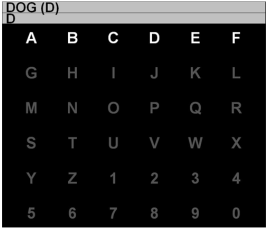
Outline

A Motivating Example



Joint work with V. Gabillon, M. Ghavamzadeh, S. Bubeck

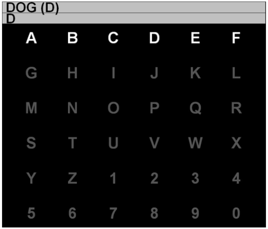
A Motivating Example



Repeat *n* times

- Choose either *row* or *column*
- *Flash* a specific row or column
- The user *thinks* either {right, wrong}

A Motivating Example



Repeat *n* times

- Choose either *row* or *column*
- *Flash* a specific row or column
- The user *thinks* either {right, wrong}

Return a *letter*

The Best Arm Identification Problem

Other Examples

- Find the *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

The setting

- M bandits (e.g., row or column)
- N arms each (e.g., rows from 1 to 6)
- Arm (m, i) has an expected value $\mu_{m,i}$
- Best arm for each bandit m

$$i^*_m = \arg\max_i \mu_{m,i}$$

The Best Arm Identification Problem

The problem

- n rounds (e.g., queries to the user)

The Best Arm Identification Problem

The problem

- n rounds (e.g., queries to the user)
- Explore bandit–arm pairs

The Best Arm Identification Problem

The problem

- n rounds (e.g., queries to the user)
- Explore bandit–arm pairs
- Return the estimated best arm J_m for each bandit m

The Best Arm Identification Problem

The problem

- n rounds (e.g., queries to the user)
- Explore bandit–arm pairs
- Return the estimated best arm J_m for each bandit m

The performance (probability of mistake)

$$\mathbb{P}[\exists m,\ J_m \neq i^*_m]$$

The Best Arm Identification Problem

Question: what about a simple uniform exploration?

The Best Arm Identification Problem

Question: what about a simple uniform exploration?

Answer: not bad, but it seems like we could do better...

The Best Arm Identification Problem

Question: what about a simple uniform exploration?

Answer: not bad, but it seems like we could do better...

Remark: if we pull each bandit–arm pair $T_{m,i}$ times $(\sum_m \sum_i T_{m,i} = n)$

$$\mathbb{P}[\exists m,\ J_m \neq i^*_m] \leq \sum_{m=1}^{M} \sum_{i=1}^{N} \exp\big(- T_{m,i} \Delta^2_{m,i} \big)$$

The Best Arm Identification Problem

Question: what about a simple uniform exploration?

Answer: not bad, but it seems like we could do better...

Remark: if we pull each bandit–arm pair $T_{m,i}$ times $(\sum_m \sum_i T_{m,i} = n)$

$$\mathbb{P}[\exists m,\ J_m \neq i^*_m] \leq \sum_{m=1}^{M} \sum_{i=1}^{N} \exp\big(- T_{m,i} \Delta^2_{m,i} \big)$$

If we know $\Delta_{m,i}$, then

$$T^*_{m,i} = \frac{\frac{1}{\Delta^2_{m,i}}}{\sum_{m'=1}^{M} \sum_{j=1}^{N} \frac{1}{\Delta^2_{m',j}}} n$$

The Best Arm Identification Problem

$$T^*_{m,i} = \frac{\frac{1}{\Delta^2_{m,i}}}{\sum_{m'=1}^{M} \sum_{j=1}^{N} \frac{1}{\Delta^2_{m',j}}} n$$

Intuition:

- large gap $\Rightarrow$ few pulls to the arm
- small gap $\Rightarrow$ many pulls to the arm

The Best Arm Identification Problem

$$T^*_{m,i} = \frac{\frac{1}{\Delta^2_{m,i}}}{\sum_{m'=1}^{M} \sum_{j=1}^{N} \frac{1}{\Delta^2_{m',j}}} n$$

Intuition:

- large gap $\Rightarrow$ few pulls to the arm
- small gap $\Rightarrow$ many pulls to the arm

Problem: we know only estimates $\widehat{\Delta}_{m,i}(t)$ with some uncertainty

A Broken Algorithm

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

- Compute

$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t)$$

A Broken Algorithm

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

- Compute

$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t)$$

- Draw arm

$$I(t) = \arg\max_{m,i} B_{m,i}(t)$$

A Broken Algorithm

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

- Compute
$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t)$$
- Draw arm
$$I(t) = \arg\max_{m,i} B_{m,i}(t)$$
- Observe $X_{I(t)}$
- Update $T_{I(t)}$ and $\widehat{\Delta}_{I(t)}(t)$

A Broken Algorithm

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

- Compute

$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t)$$

- Draw arm

$$I(t) = \arg\max_{m,i} B_{m,i}(t)$$

- Observe $X_{I(t)}$
- Update $T_{I(t)}$ and $\widehat{\Delta}_{I(t)}(t)$

Estimated gap

$$\widehat{\Delta}_{m,i}(t) = \hat{\mu}_{m,\hat{i}^*_m}(t) - \hat{\mu}_{m,i}(t)$$

A Broken Algorithm

Problem: This algorithm ***cannot*** work

A Broken Algorithm

Problem: This algorithm ***cannot*** work
Why? It only relies on estimates $\widehat{\Delta}_{m,i}(t)$ which could be *very* inaccurate

A Broken Algorithm

Problem: This algorithm ***cannot*** work

Why? It only relies on estimates $\widehat{\Delta}_{m,i}(t)$ which could be *very* inaccurate

How much? It depends on the number of times that arm has been pulled $T_{i,t}$

The Gap-E Algorithm

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

- Compute

$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t) + \sqrt{\frac{a}{T_{m,i}(t)}}$$

The Gap-E Algorithm

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

- Compute
$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t) + \sqrt{\frac{a}{T_{m,i}(t)}}$$

- Draw arm
$$I(t) = \arg\max_{m,i} B_{m,i}(t)$$

The Gap-E Algorithm

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

- Compute
$$B_{m,i}(t) = -\widehat{\Delta}_{m,i}(t) + \sqrt{\frac{a}{T_{m,i}(t)}}$$
- Draw arm
$$I(t) = \arg\max_{m,i} B_{m,i}(t)$$
- Observe $X_{I(t)}$
- Update $T_{I(t)}$ and $\widehat{\Delta}_{I(t)}(t)$

The Gap-E Algorithm

Theorem

The Gap-E algorithm with $a = \frac{4}{9}\frac{n-NM}{H}$ *has a probability of doing a mistake of*

$$\mathbb{P}[\exists m,\ J_m \neq i^*_m] \leq 2nMN \exp\Big(-\frac{1}{144}\frac{n-NM}{H}\Big)$$

with complexity $H = \sum_m \sum_i 1/\Delta^2_{m,i} = \sum_m \sum_i H_{m,i}$.

The Gap-E Algorithm

Theorem

The Gap-E algorithm with $a = \frac{4}{9}\frac{n-NM}{H}$ has a probability of doing a mistake of

$$\mathbb{P}[\exists m,\ J_m \neq i_m^*] \leq 2nMN \exp\Big(-\frac{1}{144}\frac{n-NM}{H}\Big)$$

with complexity $H = \sum_m \sum_i 1/\Delta_{m,i}^2 = \sum_m \sum_i H_{m,i}$.

Problem: the optimal parameter of GapE depends on the complexity H

The Gap-E Algorithm

Theorem

The Gap-E algorithm with $a = \frac{4}{9}\frac{n-NM}{H}$ *has a probability of doing a mistake of*

$$\mathbb{P}[\exists m,\ J_m \neq i_m^*] \leq 2nMN \exp\Big(-\frac{1}{144}\frac{n-NM}{H}\Big)$$

with complexity $H = \sum_m \sum_i 1/\Delta_{m,i}^2 = \sum_m \sum_i H_{m,i}$.

Problem: the optimal parameter of GapE depends on the complexity H
Solution: estimate H online (but we loose the theoretical guarantees...)

The Gap-E Algorithm

Gap-E

$$\mathbb{P}[\exists m,\ J_m \neq i^*_m] = O\Big(\exp\big(-\frac{n}{H}\big)\Big)$$

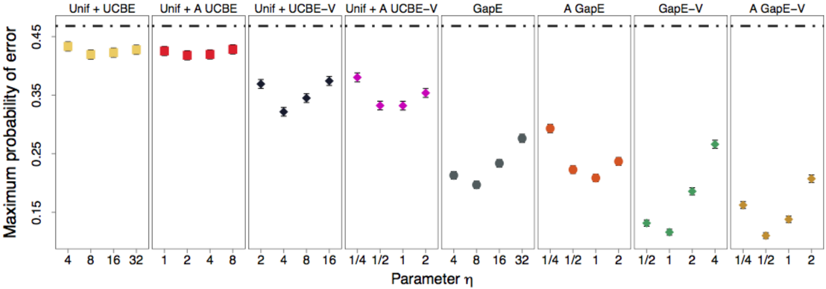
Uniform

$$\mathbb{P}[\exists m,\ J_m \neq i^*_m] = O\Big(\exp-\big(\frac{n}{\max_{m,i} H_{m,i}}\big)\Big)$$

Recall

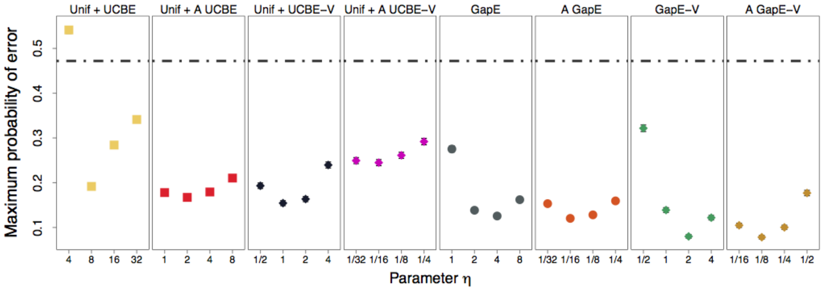
$$H = \sum_m \sum_i H_{m,i} = \sum_m \sum_i \frac{1}{\Delta^2_{m,i}}$$

The Best Arm Identification Problem



$M = 4, N = 4, n = 1400$ (mix of Bernoulli and Radamacher)

The Best Arm Identification Problem



$M = 10, N = 4, n = 1500$ (mix of Bernoulli and Radamacher)

Outline

A Motivating Example

Joint work with A. Carpentier, M. Ghavamzadeh, R. Munos, P. Auer

A Motivating Example

Given a budget of n tests

- Test one production line and measure the performance

A Motivating Example

Given a budget of n tests

- Test one production line and measure the performance
- Compute the average performance

A Motivating Example

Given a budget of n tests

- Test one production line and measure the performance
- Compute the average performance

Return an *estimated performance* for each production line *as accurate as possible*

The Active Bandit Problem

The setting

- N arms (e.g., production line)
- Arm i has an expected value μ_i and a variance σ_i^2

The Active Bandit Problem

The problem

- n rounds (e.g., tests)

The Active Bandit Problem

The problem

- n rounds (e.g., tests)
- Explore arms

The Active Bandit Problem

The problem

- n rounds (e.g., tests)
- Explore arms
- Return the empirical average $\hat{\mu}_{i,n}$ for each arm

The Active Bandit Problem

The problem

- n rounds (e.g., tests)
- Explore arms
- Return the empirical average $\hat{\mu}_{i,n}$ for each arm

The performance (the accuracy of the worst arm)

$$L_n(\mathcal{A}) = \max_i \mathbb{E}\big[(\hat{\mu}_{i,n} - \mu_i)^2\big]$$

The Active Bandit Problem

Question: what is a good strategy?

The Active Bandit Problem

Question: what is a good strategy?

Answer: let's go back to Statistics 101. If arm i is pulled $T_{i,n}$ times

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

The Active Bandit Problem

Question: what is a good strategy?

Answer: let's go back to Statistics 101. If arm i is pulled $T_{i,n}$ times

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

If we know σ_i^2, then (given that $\sum_i T_{i,n} = n$)

$$\{T^*_{i,n}\}_{i=1}^N = \arg\min_{T_{1,n},\dots,T_{N,n}} L_n = \arg\min_{T_{1,n},\dots,T_{N,n}} \max_i \frac{\sigma_i^2}{T_{i,n}}$$

The Active Bandit Problem

Question: what is a good strategy?

Answer: let's go back to Statistics 101. If arm i is pulled $T_{i,n}$ times

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

If we know σ_i^2, then (given that $\sum_i T_{i,n} = n$)

$$\{T^*_{i,n}\}_{i=1}^N = \arg\min_{T_{1,n},\dots,T_{N,n}} L_n = \arg\min_{T_{1,n},\dots,T_{N,n}} \max_i \frac{\sigma_i^2}{T_{i,n}}$$

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{i'} \sigma_{i'}^2} n$$

The Active Bandit Problem

Given the optimal (static) allocation

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{i'} \sigma_{i'}^2} n = \lambda_{i,n} n$$

The Active Bandit Problem

Given the optimal (static) allocation

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{i'} \sigma_{i'}^2} n = \lambda_{i,n} n$$

The best (smallest) loss is

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

The Active Bandit Problem

Given the optimal (static) allocation

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{i'} \sigma_{i'}^2} n = \lambda_{i,n} n$$

The best (smallest) loss is

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

We can define the *regret* of an algorithm $\mathcal{A}$ as

$$R_n = L_n(\mathcal{A}) - L^*_n$$

The Active Bandit Problem

Intuition: allocate the budget of n rounds *proportionally* to the variance of the arm

$$T^*_{i,n} = \frac{\sigma_i^2}{\sum_{i'} \sigma_{i'}^2} n = \lambda_{i,n} n$$

A UCB–based Algorithm

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

- Compute

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

A UCB–based Algorithm

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

- Compute

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

- Compute

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

A UCB–based Algorithm

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

- Compute

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

- Compute

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

- Pull arm

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

A UCB–based Algorithm

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

- Compute

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

- Compute

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

- Pull arm

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

- Observe $X_{I(t),t}$
- Update $T_{I(t),t}$ and $\hat{\sigma}^2_i$

A UCB–based Algorithm

Theorem

The UCB–based algorithm achieves a regret

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

with $\lambda_{\min} = \min_i \lambda_i$.

A UCB–based Algorithm

Theorem

The UCB–based algorithm achieves a regret

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

with $\lambda_{\min} = \min_i \lambda_i$.

A UCBV–based Algorithm

Theorem

The UCBV–based algorithm achieves a regret

$$R_n(\mathcal{A}) \leq O\Big(\frac{\log n}{n^{3/2}\lambda_{\min}}\Big)$$

A UCBV–based Algorithm

Theorem

The UCBV–based algorithm achieves a regret

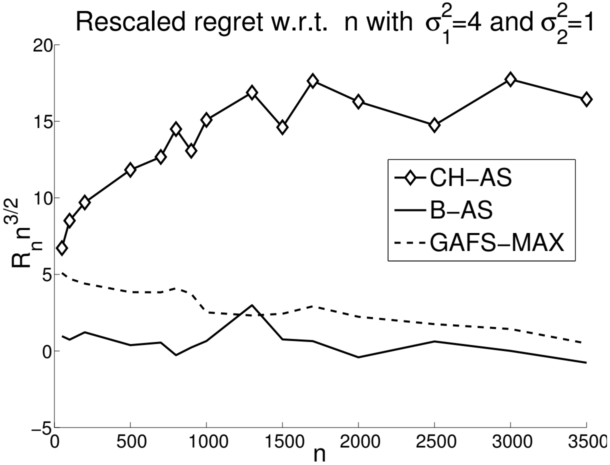
$$R_n(\mathcal{A}) \leq O\Big(\frac{\log n}{n^{3/2}\lambda_{\min}}\Big)$$

for Gaussian distribution (unimodal distributions??)

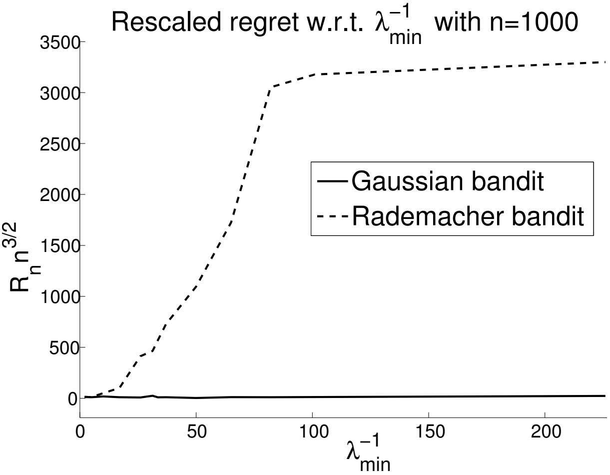
$$R_n(\mathcal{A}) \leq O\Big(\frac{\log n}{n^{3/2}}\Big)$$

no $\lambda_{\min}$*!!*

A UCB–based Algorithm



A UCB–based Algorithm



Outline

A Motivating Example

Google AdWords

Joint work with F. Trovò, N. Gatti

A Motivating Example

Repeat *n* times

- Collect the bids of the advertisers
- Allocate advertisements to slots
- Ask for payments if ads are clicked

A Motivating Example

Repeat *n* times

- Collect the bids of the advertisers
- Allocate advertisements to slots
- Ask for payments if ads are clicked

Maximize the *revenue* over time

Learning in Sponsored Search Auctions

The setting

- M advertisers

Learning in Sponsored Search Auctions

The setting

- M advertisers
- Each advertiser has a quality ρ_i (i.e., probability of click) and a value v_i

Learning in Sponsored Search Auctions

The setting

- M advertisers
- Each advertiser has a quality ρ_i (i.e., probability of click) and a value v_i
- K slots

Learning in Sponsored Search Auctions

The setting

- M advertisers
- Each advertiser has a quality ρ_i (i.e., probability of click) and a value v_i
- K slots
- Γ_k probability that a user will *look* at slot k

Learning in Sponsored Search Auctions

The setting

- M advertisers
- Each advertiser has a quality ρ_i (i.e., probability of click) and a value v_i
- K slots
- Γ_k probability that a user will *look* at slot k
- $\Rightarrow$ click–through–rate (CRT) of an ad i displayed at slot k

$$\rho_i \Gamma_k$$

Learning in Sponsored Search Auctions

The problem

- n rounds (e.g., same query from different users)

Learning in Sponsored Search Auctions

The problem

- n rounds (e.g., same query from different users)
- Estimate the quality of each ad

Learning in Sponsored Search Auctions

The problem

- n rounds (e.g., same query from different users)
- Estimate the quality of each ad
- Allocate ads to slots so as to maximize the revenue

Learning in Sponsored Search Auctions

Question: how can we solve this problem?

Learning in Sponsored Search Auctions

Question: how can we solve this problem?

Answer: *easy!* it is just a standard bandit problem!

$$\text{expected value of ad } i = \rho_i v_i$$

Learning in Sponsored Search Auctions

Question: how can we solve this problem?

Answer: *easy!* it is just a standard bandit problem!

$$\text{expected value of ad } i = \rho_i v_i$$

Use a bandit algorithm learning the ad with the *highest expected value*

Learning in Sponsored Search Auctions

Question: how can we solve this problem?

Answer: *easy!* it is just a standard bandit problem!

$$\text{expected value of ad } i = \rho_i v_i$$

Use a bandit algorithm learning the ad with the *highest expected value*

Problem: ***This cannot work with strategic advertisers!*** (see game theory/mechanism design)

Learning in Sponsored Search Auctions

Question: if we know the qualities ρ_i, what should we do?

Learning in Sponsored Search Auctions

Question: if we know the qualities ρ_i, what should we do?
Answer: use an (affine) VCG auction
Receive the bids b_i

Learning in Sponsored Search Auctions

Question: if we know the qualities ρ_i, what should we do?
Answer: use an (affine) VCG auction
Receive the bids b_i

- Sort the advertisers in (decreasing) order $\rho_i b_i$

Learning in Sponsored Search Auctions

Question: if we know the qualities ρ_i, what should we do?
Answer: use an (affine) VCG auction
Receive the bids b_i

- Sort the advertisers in (decreasing) order $\rho_i b_i$
- Allocate to slot k the (k)-th ad

Learning in Sponsored Search Auctions

Question: if we know the qualities ρ_i, what should we do?
Answer: use an (affine) VCG auction
Receive the bids b_i

- Sort the advertisers in (decreasing) order $\rho_i b_i$
- Allocate to slot k the (k)-th ad
- Ask payments

$$p_k = \sum_{l=k+1}^{K} (\Gamma_{l-1} - \Gamma_l) \max_{\rho_i b_i; l}$$

Learning in Sponsored Search Auctions

Question: if we know the qualities ρ_i, what should we do?
Answer: use an (affine) VCG auction
Receive the bids b_i

- Sort the advertisers in (decreasing) order $\rho_i b_i$
- Allocate to slot k the (k)-th ad
- Ask payments

$$p_k = \sum_{l=k+1}^{K} (\Gamma_{l-1} - \Gamma_l) \max_{\rho_i b_i; l}$$

Game theory: this mechanism *forces* all the advertisers to bid $b_i = v_i$ (i.e., incentive compatibility)

Learning in Sponsored Search Auctions

Intuition:

- learning ρ_i over n rounds
- preserve the *incentive compatibility*

Learning in Sponsored Search Auctions

Intuition:

- learning ρ_i over n rounds
- preserve the *incentive compatibility*

$\Rightarrow$ we want a *learning* mechanism which learns in a incentive compatible way

Learning in Sponsored Search Auctions

Intuition:

- learning ρ_i over n rounds
- preserve the *incentive compatibility*

$\Rightarrow$ we want a *learning* mechanism which learns in a incentive compatible way
$\Rightarrow$ a bandit problem with *strategic* constraints

The Explor-Exploit Algorithm

At round $t = 1, \ldots, \tau$ (*pure exploration*)

- Assign ads to slots in an arbitrary way
- Observe the clicks
- No payments

The Explor-Exploit Algorithm

At round $t = 1, \ldots, \tau$ (*pure exploration*)

- Assign ads to slots in an arbitrary way
- Observe the clicks
- No payments

Compute the estimated qualities $\hat{\rho}_i$

The Explor-Exploit Algorithm

At round $t = 1, \dots, \tau$ (*pure exploration*)

- Assign ads to slots in an arbitrary way
- Observe the clicks
- No payments

Compute the estimated qualities $\hat{\rho}_i$

At round $t = \tau + 1, \dots, n$ (*pure exploitation*)

- Use a VCG auction with the estimated qualities $\hat{\rho}_i$

The Explor-Exploit Algorithm

Theorem

The Explor-Exploit algorithm with $\tau \approx n^{2/3}$

$$R_n = \underbrace{n \sum_{k=1}^{K} p_k}_{\textit{revenue of VCG}} - \underbrace{\sum_{t=1}^{n} \sum_{k=1}^{K} p_{k,t}}_{\textit{revenue of Exp-Exp}} \leq \widetilde{O}(n^{2/3} K^{2/3} N^{1/3})$$

The Explor-Exploit Algorithm

Theorem

The Explor-Exploit algorithm with $\tau \approx n^{2/3}$

$$R_n = \underbrace{n\sum_{k=1}^{K} p_k}_{\text{revenue of VCG}} - \underbrace{\sum_{t=1}^{n}\sum_{k=1}^{K} p_{k,t}}_{\text{revenue of Exp-Exp}} \leq \widetilde{O}(n^{2/3}K^{2/3}N^{1/3})$$

$$\Rightarrow \frac{R_n}{n} = \widetilde{O}(n^{-1/3}K^{2/3}N^{1/3})$$

The Explor-Exploit Algorithm

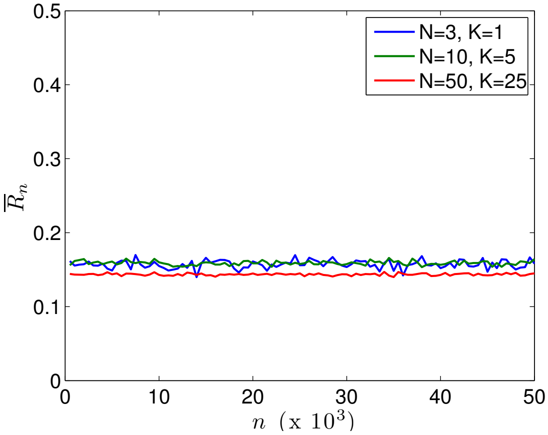
Question: first exploring and then exploiting... it does not seem a *smart* algorithm, can we do better?

The Explor-Exploit Algorithm

Question: first exploring and then exploiting... it does not seem a *smart* algorithm, can we do better?

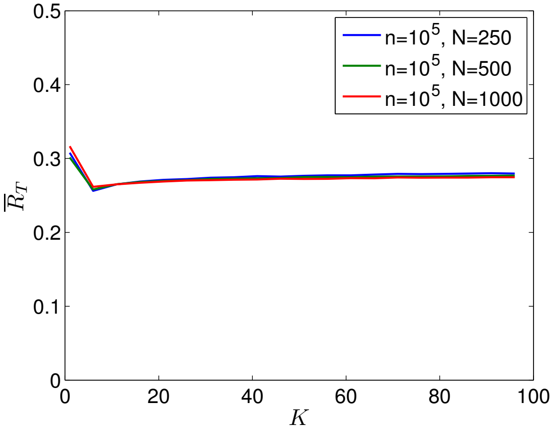
Answer: No! (with the current hard constraints)

The Explor-Exploit Algorithm



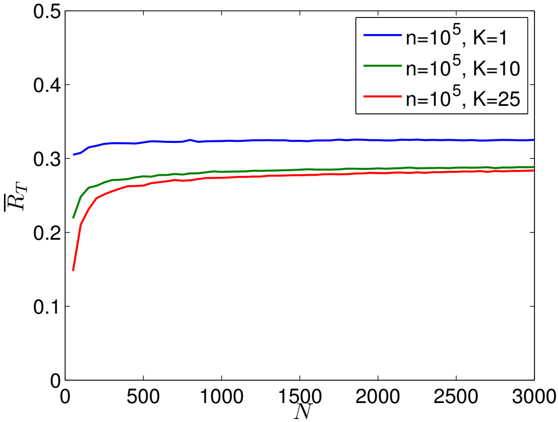
$$\overline{R}_n = R_n/\text{bound}$$

The Explor-Exploit Algorithm



$$\overline{R}_n = R_n/\text{bound}$$

The Explor-Exploit Algorithm



$$\overline{R}_n = R_n/\text{bound}$$

Outline

Conclusions

- Stochastic multi–armed bandit model (e.g., clinical trials)
- Best–arm identification (e.g., BCI application)
- Active bandit problem (e.g., production lines monitoring)
- Strategic bandits (e.g., sponsored search auctions)
- Many more!!!

Conclusions

Remark: all these problems seem to share the same structure...

Open problem: bandits as an online learning optimization method with limited feedback?

Thank you!!

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