

Multi-Player Bandits Revisited

Decentralized Multi-Player Multi-Arm Bandits

Lilian Besson

Joint work with Émilie Kaufmann

PhD Student

Team SCEE, IETR, CentraleSupélec, Rennes
& Team SequeL, CRISTAL, Inria, Lille

CMAP Seminar – 31st October 2018



Motivation

We control some communicating devices, they want to use a wireless access point.

Motivation

We control some communicating devices, they want to use a wireless access point.

- Insert them in a crowded wireless network.
- With a protocol slotted in both time and frequency.

Motivation

We control some communicating devices, they want to use a wireless access point.

- Insert them in a crowded wireless network.
- With a protocol slotted in both time and frequency.

Goal

- Maintain a good Quality of Service.
- With no centralized control as it costs network overhead.

Motivation

We control some communicating devices, they want to use a wireless access point.

- Insert them in a crowded wireless network.
- With a protocol slotted in both time and frequency.

Goal

- Maintain a good Quality of Service.
- With no centralized control as it costs network overhead.

How?

- Devices can choose a different radio channel at each time
↪ learn the best one with a sequential algorithm!

Outline

- 1 Introduction
- 2 Our model: 3 different feedback levels
- 3 Regret of the system, and our lower bound on regret

Outline

- 1 Introduction
- 2 Our model: 3 different feedback levels
- 3 Regret of the system, and our lower bound on regret

- 4 Quick reminder on single-player MAB algorithms
- 5 New multi-player non-coordinated decentralized algorithms
- 6 Our upper bound on regret for MCTopM

Outline

- 1 Introduction
- 2 Our model: 3 different feedback levels
- 3 Regret of the system, and our lower bound on regret

- 4 Quick reminder on single-player MAB algorithms
- 5 New multi-player non-coordinated decentralized algorithms
- 6 Our upper bound on regret for MCTopM

- 7 Experimental results
- 8 Review of two more recent articles

Outline

- 1 Introduction
- 2 Our model: 3 different feedback levels
- 3 Regret of the system, and our lower bound on regret

- 4 Quick reminder on single-player MAB algorithms
- 5 New multi-player non-coordinated decentralized algorithms
- 6 Our upper bound on regret for MCTopM

- 7 Experimental results
- 8 Review of two more recent articles

- 9 Conclusion

Outline and reference

- 1 Introduction
- 2 Our model: 3 different feedback levels
- 3 Regret of the system, and our lower bound on regret

- 4 Quick reminder on single-player MAB algorithms
- 5 New multi-player non-coordinated decentralized algorithms
- 6 Our upper bound on regret for MCTopM

- 7 Experimental results
- 8 Review of two more recent articles

- 9 Conclusion

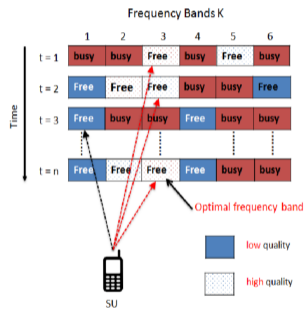
Based on “Multi-Player Bandits Revisited”, by Lilian Besson & Émilie Kaufmann. [arXiv:1711.02317](https://arxiv.org/abs/1711.02317), presented at ALT 2018 (Lanzarote, Spain) in April.

Our model

- ① Our communication model
- ② With or without sensing
- ③ Background traffic, and rewards
- ④ Different feedback levels
- ⑤ Goal

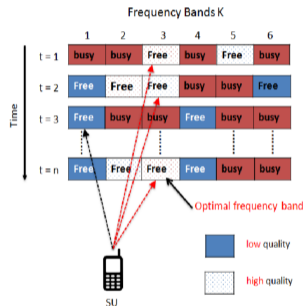
Our communication model

K radio channels (e.g., 10). Discrete and synchronized time $t \geq 1$.



Our communication model

K radio channels (e.g., 10). Discrete and synchronized time $t \geq 1$.



Dynamic device = dynamic radio reconfiguration

- It decides each time the channel it uses to send each packet.
- It can implement a simple decision algorithm.

Our model

“Easy” case

- $M \leq K$ devices always communicate and try to access the network, independently without centralized supervision,
- Background traffic is i.i.d..

Our model

“Easy” case

- $M \leq K$ devices always communicate and try to access the network, independently without centralized supervision,
- Background traffic is i.i.d..

Two variants : with or without sensing

- ① With sensing: Device first senses for presence of Primary Users that have strict priority (background traffic), then use Ack to detect collisions.
- ② Without sensing: same background traffic, but cannot sense, so only Ack is used.

Background traffic, and rewards

i.i.d. background traffic

- K channels, modeled as Bernoulli (0/1) distributions of mean $\mu_k =$ background traffic from Primary Users, bothering the dynamic devices,
- M devices, each uses channel $A^j(t) \in \{1, \dots, K\}$ at time t .

Background traffic, and rewards

i.i.d. background traffic

- K channels, modeled as Bernoulli (0/1) distributions of mean $\mu_k =$ background traffic from Primary Users, bothering the dynamic devices,
- M devices, each uses channel $A^j(t) \in \{1, \dots, K\}$ at time t .

Rewards

$$r^j(t) := Y_{A^j(t),t} \times \overline{\mathbb{1}(C^j(t))} = \mathbb{1}(\text{uplink \& Ack})$$

- with sensing information $\forall k, Y_{k,t} \stackrel{\text{iid}}{\sim} \text{Bern}(\mu_k) \in \{0, 1\}$,
- collision for device j : $C^j(t) = \mathbb{1}(\text{alone on arm } A^j(t))$.
 $\hookrightarrow r^j(t)$ **combined** binary reward but not from two Bernoulli!

3 feedback levels

$$r^j(t) := Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$$

- ① “Full **feedback**”: observe both $Y_{A^j(t),t}$ and $C^j(t)$ separately,
↪ Not realistic enough, we don't focus on it.

3 feedback levels

$$r^j(t) := Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$$

- ① “Full feedback”: observe both $Y_{A^j(t),t}$ and $C^j(t)$ separately,
↪ Not realistic enough, we don't focus on it.
- ② “Sensing”: first observe $Y_{A^j(t),t}$, then $C^j(t)$ only if $Y_{A^j(t),t} \neq 0$,
↪ Models licensed protocols (ex. ZigBee), our main focus.

3 feedback levels

$$r^j(t) := Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$$

- ① “Full feedback”: observe both $Y_{A^j(t),t}$ and $C^j(t)$ separately,
 \hookrightarrow Not realistic enough, we don't focus on it.
- ② “Sensing”: first observe $Y_{A^j(t),t}$, then $C^j(t)$ only if $Y_{A^j(t),t} \neq 0$,
 \hookrightarrow Models licensed protocols (ex. ZigBee), our main focus.
- ③ “No sensing”: observe only the combined $Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$,
 \hookrightarrow Unlicensed protocols (ex. LoRaWAN), harder to analyze !

3 feedback levels

$$r^j(t) := Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$$

- ① “Full feedback”: observe both $Y_{A^j(t),t}$ and $C^j(t)$ separately,
 \hookrightarrow Not realistic enough, we don't focus on it.
- ② “Sensing”: first observe $Y_{A^j(t),t}$, then $C^j(t)$ only if $Y_{A^j(t),t} \neq 0$,
 \hookrightarrow Models licensed protocols (ex. ZigBee), our main focus.
- ③ “No sensing”: observe only the combined $Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$,
 \hookrightarrow Unlicensed protocols (ex. LoRaWAN), harder to analyze !

But all consider the same instantaneous **reward** $r^j(t)$.

Goal

Goal

- Minimize packet loss ratio
(= maximize nb of received Ack)
- in a finite-space discrete-time Decision Making Problem.

Goal

Goal

- Minimize packet loss ratio
(= maximize nb of received Ack)
- in a finite-space discrete-time Decision Making Problem.

Solution ?

Multi-Armed Bandit algorithms

- decentralized and
- used independently by each dynamic device.

Centralized regret

A measure of success

- Not the network throughput or collision probability,
- We study the centralized (**expected**) regret:

$$R_T(\boldsymbol{\mu}, M, \rho) := \left(\sum_{k=1}^M \mu_k^* \right) T - \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{t=1}^T \sum_{j=1}^M r^j(t) \right].$$

Notation: μ_k^* is the mean of the k -best arm (k -th largest in $\boldsymbol{\mu}$):

- $\mu_1^* := \max \boldsymbol{\mu}$,
- $\mu_2^* := \max \boldsymbol{\mu} \setminus \{\mu_1^*\}$,
- etc.

Ref: [Lai & Robbins, 1985], [Liu & Zhao, 2009], [Anandkumar et al, 2010]

Centralized regret

A measure of success

- Not the network throughput or collision probability,
- We study the centralized (expected) regret:

$$R_T(\boldsymbol{\mu}, M, \rho) := \left(\sum_{k=1}^M \mu_k^* \right) T - \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{t=1}^T \sum_{j=1}^M r^j(t) \right].$$

Two directions of analysis

- How good a decentralized algorithm can be in this setting?
 ↪ **Lower Bound** on the regret, for **any** algorithm !
- How good is my decentralized algorithm in this setting?
 ↪ **Upper Bound** on the regret, for **one** algorithm !

Lower bound

- ① Decomposition of the regret in 3 terms,
- ② Asymptotic lower bound on one term,
- ③ And for the regret,
- ④ Possibly wrong result, not sure yet!

Decomposition on the regret

Decomposition

For any algorithm, decentralized or not, we have

$$R_T(\boldsymbol{\mu}, M, \rho) = \sum_{k \in M\text{-worst}} (\mu_M^* - \mu_k) \mathbb{E}_\mu[T_k(T)] \\ + \sum_{k \in M\text{-best}} (\mu_k - \mu_M^*) (T - \mathbb{E}_\mu[T_k(T)]) + \sum_{k=1}^K \mu_k \mathbb{E}_\mu[\mathcal{C}_k(T)].$$

Notations for an arm $k \in \{1, \dots, K\}$:

- $T_k^j(T) := \sum_{t=1}^T \mathbb{1}(A^j(t) = k)$, counts selections by the player $j \in \{1, \dots, M\}$,
- $T_k(T) := \sum_{j=1}^M T_k^j(T)$, counts selections by all M players,
- $\mathcal{C}_k(T) := \sum_{t=1}^T \mathbb{1}(\exists j_1 \neq j_2, A^{j_1}(t) = k = A^{j_2}(t))$, counts collisions.

Decomposition on the regret

Decomposition

For any algorithm, decentralized or not, we have

$$R_T(\boldsymbol{\mu}, M, \rho) = \sum_{k \in M\text{-worst}} (\mu_M^* - \mu_k) \mathbb{E}_\mu[T_k(T)] \\ + \sum_{k \in M\text{-best}} (\mu_k - \mu_M^*) (T - \mathbb{E}_\mu[T_k(T)]) + \sum_{k=1}^K \mu_k \mathbb{E}_\mu[\mathcal{C}_k(T)].$$

Small regret can be attained if...

- 1 Devices can quickly identify the bad arms M -worst, and not play them too much (number of sub-optimal selections),

Decomposition on the regret

Decomposition

For any algorithm, decentralized or not, we have

$$R_T(\boldsymbol{\mu}, M, \rho) = \sum_{k \in M\text{-worst}} (\mu_M^* - \mu_k) \mathbb{E}_\mu[T_k(T)] \\ + \sum_{k \in M\text{-best}} (\mu_k - \mu_M^*) (T - \mathbb{E}_\mu[T_k(T)]) + \sum_{k=1}^K \mu_k \mathbb{E}_\mu[\mathcal{C}_k(T)].$$

Small regret can be attained if...

- ① Devices can quickly identify the bad arms M -worst, and not play them too much (number of sub-optimal selections),
- ② Devices can quickly identify the best arms, and most surely play them (**number of optimal non-selections**),

Decomposition on the regret

Decomposition

For any algorithm, decentralized or not, we have

$$R_T(\boldsymbol{\mu}, M, \rho) = \sum_{k \in M\text{-worst}} (\mu_M^* - \mu_k) \mathbb{E}_\mu[T_k(T)] \\ + \sum_{k \in M\text{-best}} (\mu_k - \mu_M^*) (T - \mathbb{E}_\mu[T_k(T)]) + \sum_{k=1}^K \mu_k \mathbb{E}_\mu[\mathcal{C}_k(T)].$$

Small regret can be attained if...

- ① Devices can quickly identify the bad arms M -worst, and not play them too much (number of sub-optimal selections),
- ② Devices can quickly identify the best arms, and most surely play them (number of optimal non-selections),
- ③ Devices can use orthogonal channels (number of collisions).

Lower bound on the regret

Lower bound

For any algorithm, decentralized or not, we have

$$R_T(\boldsymbol{\mu}, M, \rho) \geq \sum_{k \in M\text{-worst}} (\mu_M^* - \mu_k) \mathbb{E}_\mu[T_k(T)]$$

Asymptotic lower bound on the regret I

Theorem 1

[Besson & Kaufmann, 2018]

Sub-optimal arms selections are lower bounded asymptotically,

$$\forall \text{ player } j, \text{ bad arm } k, \quad \liminf_{T \rightarrow +\infty} \frac{\mathbb{E}_{\mu}[T_k^j(T)]}{\log T} \geq \frac{1}{\text{kl}(\mu_k, \mu_M^*)},$$

Where $\text{kl}(x, y) := \mathcal{KL}(\mathcal{B}(x), \mathcal{B}(y)) = x \log\left(\frac{x}{y}\right) + (1-x) \log\left(\frac{1-x}{1-y}\right)$ is the binary KL divergence.

Proof: using classical information theory tools (Kullback-Leibler divergence, change of distributions)...

Ref: [Garivier et al, 2016]

Asymptotic lower bound on the regret II

Theorem 2

[Besson & Kaufmann, 2018]

For any uniformly efficient decentralized policy, and any non-degenerated problem μ ,

$$\liminf_{T \rightarrow +\infty} \frac{R_T(\mu, M, \rho)}{\log(T)} \geq M \times \left(\sum_{k \in M\text{-worst}} \frac{(\mu_M^* - \mu_k)}{\text{kl}(\mu_k, \mu_M^*)} \right).$$

Asymptotic lower bound on the regret II

Theorem 2

[Besson & Kaufmann, 2018]

For any uniformly efficient decentralized policy, and any non-degenerated problem μ ,

$$\liminf_{T \rightarrow +\infty} \frac{R_T(\mu, M, \rho)}{\log(T)} \geq M \times \left(\sum_{k \in M\text{-worst}} \frac{(\mu_M^* - \mu_k)}{\text{kl}(\mu_k, \mu_M^*)} \right).$$

Remarks

- The centralized multiple-play lower bound is the same without the M **multiplicative factor**...
 ↪ “**price of non-coordination**” = M = nb of player? Ref: [Anantharam et al, 1987]
- Improved state-of-the-art lower bound, but still not perfect: collisions should also be controlled!

Possibly wrong result, not sure yet?

- A recent article studied the same problem ([arXiv:1809.08151](https://arxiv.org/abs/1809.08151)).

SIC-MMAB: Synchronisation Involves Communication in Multiplayer Multi-Armed Bandits

Etienne Boursier ^{*1} and Vianney Perchet ^{1,2}

¹CMLA, ENS Cachan, CNRS, Université Paris-Saclay, 94235 Cachan, France

²Criteo AI Lab, Paris

September 24, 2018

Abstract

We consider the stochastic multiplayer multi-armed bandit problem, where several players pull arms simultaneously and a collision occurs if the same arm is pulled by more than one player; this is a standard model of cognitive radio networks. We construct a decentralized algorithm that achieves the same performances as a centralized one, if players are synchronized and observe their collisions. We actually construct a communication protocol between players by enforcing willingly collisions, allowing them to share their exploration.

With a weaker feedback, when collisions are not observed, we still maintain some communication between players but at the cost of some extra multiplicative term in the regret. We also prove that the logarithmic growth of the regret is still achievable in the dynamic case where players are not synchronized with each other, thus preventing communication.

Finally, we prove that if all players follow naively the celebrated UCB algorithm, the total regret grows linearly.

Possibly wrong result, not sure yet?

- A recent article studied the same problem ([arXiv:1809.08151](https://arxiv.org/abs/1809.08151)).
- They showed a regret upper bound for their SIC-MMAB algorithm which disproves our regret lower bound:
they do not suffer from any “price of decentralization” 😊!

2.3.5 Total regret

Theorem [1](#) finally provides an asymptotical upper bound of the regret:

Theorem 1. For any given set of parameters K , M and μ :

$$\lim_{T \rightarrow \infty} \frac{R_T}{\log(T)} \leq c_1 \sum_{k > M} \frac{1}{\mu_{(M)} - \mu_{(k)}} + c_2 K M$$

where c_1 and c_2 are two problem independent constants

Proof. This is a direct consequence of Lemmas [1](#), [2](#), [3](#) and [4](#) and the regret decomposition given by Equation [\(2\)](#). □

Possibly wrong result, not sure yet?

- A recent article studied the same problem ([arXiv:1809.08151](https://arxiv.org/abs/1809.08151)).
- They showed a regret upper bound for their SIC-MMAB algorithm which disproves our regret lower bound:
they do not suffer from any “price of decentralization” 😊!
- Their algorithm works fine in practice, see later, and their proof seems fine, but the point they indicate as wrong in our paper is not clear and we couldn't find an error.
- \implies I will work on this more in the near future!

“SIC-MMAB: Synchronisation Involves Communication in Multiplayer Multi-Armed Bandits”, by Etienne Boursier & Vianney Perchet, [arXiv:1809.08151](https://arxiv.org/abs/1809.08151)

Single-player MAB algorithms

- ① Upper Confidence Bound algorithm : UCB_1 ,
- ② Kullback-Leibler UCB algorithm : kl-UCB.

Upper Confidence Bound algorithm (UCB₁)

① For the first K steps ($t = 1, \dots, K$), try each channel once.

② Then for the next steps $t > K$:

- $T_k^j(t) := \sum_{s=1}^t \mathbb{1}(A^j(s) = k)$ selections of channel k ,

- $S_k^j(t) := \sum_{s=1}^t Y_k(s) \mathbb{1}(A^j(s) = k)$ sum of sensing information.

- Compute the index $\text{UCB}_k^j(t) := \underbrace{\frac{S_k^j(t)}{T_k^j(t)}}_{\text{Empirical Mean } \widehat{\mu}_k(t)} + \underbrace{\sqrt{\frac{\log(t)}{2 T_k^j(t)}}}_{\text{Confidence Bonus}}$,

- Choose channel $A^j(t) = \underset{k}{\text{argmax}} \text{UCB}_k^j(t)$,

- Update $T_k^j(t+1)$ and $S_k^j(t+1)$.

Kullback-Leibler UCB algorithm (kl-UCB)

① For the first K steps ($t = 1, \dots, K$), try each channel once.

② Then for the next steps $t > K$:

- $T_k^j(t) := \sum_{s=1}^t \mathbb{1}(A^j(s) = k)$ selections of channel k ,
- $S_k^j(t) := \sum_{s=1}^t Y_k(s) \mathbb{1}(A^j(s) = k)$ sum of sensing information.
- Compute $\text{UCB}_k^j(t)$, Upper Confidence Bound on mean μ_k

$$\text{UCB}_k^j(t) := \sup_{q \in [a, b]} \left\{ q : \text{kl} \left(\frac{S_k^j(t)}{T_k^j(t)}, q \right) \leq \frac{\log(t)}{T_k^j(t)} \right\},$$
- Choose channel $A^j(t) = \underset{k}{\text{argmax}} \text{UCB}_k^j(t)$,
- Update $T_k^j(t+1)$ and $S_k^j(t+1)$.

Known result: kl-UCB is asymptotically optimal for 1-player Bernoulli stochastic bandit. Ref: [Garivier

Multi-player decentralized algorithms

- ① Common building blocks of previous algorithms,
- ② One of our proposal: the MCTopM algorithm.

Algorithms for this easier model

Building blocks: separate the two aspects

- 1 MAB policy to learn the best arms (use sensing $Y_{A^j(t),t}$),
- 2 Orthogonalization scheme to avoid collisions (use collision indicators $C^j(t)$).

Many different proposals for decentralized learning policies

- “State-of-the-art”: RhoRand
- Recent: MEGA and Musical Chair.

Ref: [Anandkumar et al, 2011]

Ref: [Avner & Mannor, 2015], [Shamir et al, 2016]

Algorithms for this easier model

Building blocks: separate the two aspects

- ① MAB policy to learn the best arms (use sensing $Y_{A^j(t),t}$),
- ② Orthogonalization scheme to avoid collisions (use collision indicators $C^j(t)$).

Many different proposals for decentralized learning policies

- “State-of-the-art”: RhoRand
- Recent: MEGA and Musical Chair.

Ref: [Anandkumar et al, 2011]

Ref: [Avner & Mannor, 2015], [Shamir et al, 2016]

Our contributions:

[Besson & Kaufmann, 2018]

Two new orthogonalization scheme inspired by RhoRand and Musical Chair, combined with the use of kl-UCB indices.

Ideas for the MCTopM algorithm

- Based on sensing information, each user j keeps $UCB_k^j(t)$ for each arm k ,
- Use it to estimate the M best arms:

$$\widehat{M}^j(t) = \{\text{arms with } M \text{ largest } UCB_k^j(t)\}.$$

Two ideas:

- Always pick an arm $A^j(t) \in \widehat{M}^j(t)$,
- Try not to switch arm too often.

Ref: [Anandkumar et al, 2011]

Introduce a fixed state $s^j(t)$:

Ref: [Shamir et al, 2016]

first non fixed, then fix when happy about an arm and no collision.

MCTopM algorithm

```
1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$  and  $s^j(1) = \text{Non fixed}$ 
2 for  $t = 1, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : \text{UCB}_k^j(t-1) \leq \text{UCB}_{A^j(t)}^j(t-1)\})$  // not empty
5      $s^j(t+1) = \text{Non fixed}$  // go for arm with smaller index at  $t-1$ 
```

MCTopM algorithm

```
1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$  and  $s^j(1) = \text{Non fixed}$ 
2 for  $t = 1, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : \text{UCB}_k^j(t-1) \leq \text{UCB}_{A^j(t)}^j(t-1)\})$  // not empty
5      $s^j(t+1) = \text{Non fixed}$  // go for arm with smaller index at  $t-1$ 
6   else if  $C^j(t)$  and  $s^j(t) = \text{Non fixed}$  then // collision and not fixed
7      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t))$  // transition (2)
8      $s^j(t+1) = \text{Non fixed}$ 
```

MCTopM algorithm

```
1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$  and  $s^j(1) = \text{Non fixed}$ 
2 for  $t = 1, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : \text{UCB}_k^j(t-1) \leq \text{UCB}_{A^j(t)}^j(t-1)\})$  // not empty
5      $s^j(t+1) = \text{Non fixed}$  // go for arm with smaller index at  $t-1$ 
6   else if  $C^j(t)$  and  $s^j(t) = \text{Non fixed}$  then // collision and not fixed
7      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t))$  // transition (2)
8      $s^j(t+1) = \text{Non fixed}$ 
9   else // transition (1) or (4)
10     $A^j(t+1) = A^j(t)$  // stay on the previous arm
11     $s^j(t+1) = \text{Fixed}$  // become or stay fixed on a "chair"
12  end
```

MCTopM algorithm

```
1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$  and  $s^j(1) = \text{Non fixed}$ 
2 for  $t = 1, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : \text{UCB}_k^j(t-1) \leq \text{UCB}_{A^j(t)}^j(t-1)\})$  // not empty
5      $s^j(t+1) = \text{Non fixed}$  // go for arm with smaller index at  $t-1$ 
6   else if  $C^j(t)$  and  $s^j(t) = \text{Non fixed}$  then // collision and not fixed
7      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t))$  // transition (2)
8      $s^j(t+1) = \text{Non fixed}$ 
9   else // transition (1) or (4)
10     $A^j(t+1) = A^j(t)$  // stay on the previous arm
11     $s^j(t+1) = \text{Fixed}$  // become or stay fixed on a "chair"
12  end
13  Play arm  $A^j(t+1)$ , get new observations (sensing and collision),
14  Compute the indices  $\text{UCB}_k^j(t+1)$  and set  $\widehat{M}^j(t+1)$  for next step.
15 end
```

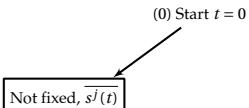

MCTopM algorithm illustrated, step by step

(0) Start $t = 0$

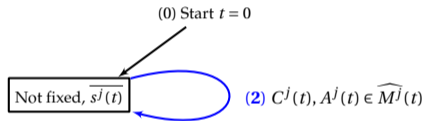
MCTopM algorithm illustrated, step by step

(0) Start $t = 0$

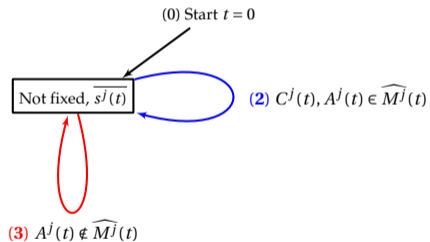
Not fixed, $\overline{s^j(t)}$



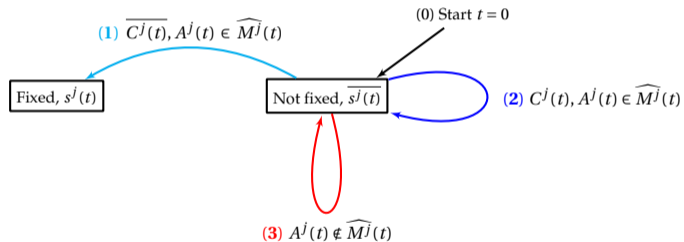
MCTopM algorithm illustrated, step by step



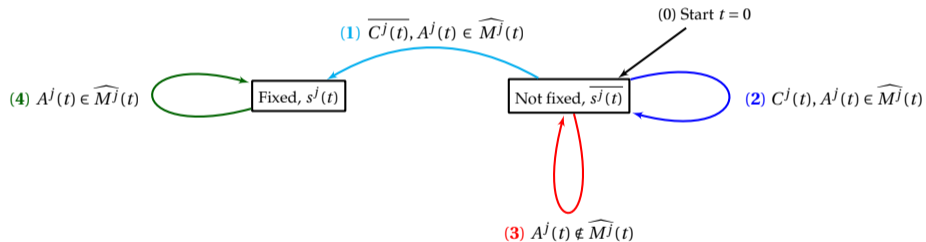
MCTopM algorithm illustrated, step by step



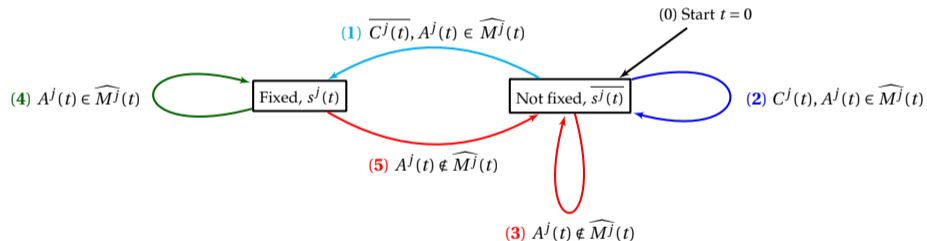
MCTopM algorithm illustrated, step by step



MCTopM algorithm illustrated, step by step



MCTopM algorithm illustrated, step by step



Regret upper bound

- ① Theorem,
- ② Remarks.

Regret upper bound for MCTopM

Theorem 3

[Besson & Kaufmann, 2018]

One term is controlled by the two others:

$$\begin{aligned} & \sum_{k \in M\text{-best}} (\mu_k - \mu_M^*) (T - \mathbb{E}_\mu[T_k(T)]) \\ & \leq (\mu_1^* - \mu_M^*) \left(\sum_{k \in M\text{-worst}} \mathbb{E}_\mu[T_k(T)] + \sum_{k \in M\text{-best}} \mathbb{E}_\mu[C_k(T)] \right) \end{aligned}$$

So only need to work on both **sub-optimal selections** and **collisions**.

Regret upper bound for MCTopM

Theorem 3

[Besson & Kaufmann, 2018]

One term is controlled by the two others:

$$\begin{aligned} & \sum_{k \in M\text{-best}} (\mu_k - \mu_M^*) (T - \mathbb{E}_\mu[T_k(T)]) \\ & \leq (\mu_1^* - \mu_M^*) \left(\sum_{k \in M\text{-worst}} \mathbb{E}_\mu[T_k(T)] + \sum_{k \in M\text{-best}} \mathbb{E}_\mu[C_k(T)] \right) \end{aligned}$$

Theorem 4

[Besson & Kaufmann, 2018]

If all M players use MCTopM with kl-UCB:

$$\forall \mu, \exists G_{M,\mu}, \quad R_T(\mu, M, \rho) \leq G_{M,\mu} \times \log(T) + o(\log T).$$

Regret upper bound for MCTopM

How?

Control both terms, both are logarithmic at finite horizon:

- Suboptimal selections with the “classical analysis” on kl-UCB indexes.
- Collisions are also controlled with inequalities on the kl-UCB indexes...

Regret upper bound for MCTopM

How?

Control both terms, both are logarithmic at finite horizon:

- Suboptimal selections with the “classical analysis” on kl-UCB indexes.
- Collisions are also controlled with inequalities on the kl-UCB indexes...

Remarks

- The constant $G_{M,\mu}$ scales as M^3 , way better than RhoRand’s constant scaling as $M^2 \binom{2M-1}{M}$,
- We also minimize the number of channel switching: interesting as changing arm costs energy in radio systems,
- For the suboptimal selections, we match our lower bound !

Sketch of the proof

- 1 Bound the expected number of collisions by M times the number of collisions for non-fixed players,

Sketch of the proof

- 1 Bound the expected number of collisions by M times the number of collisions for non-fixed players,
- 2 Bound the expected number of **transitions of type (3) and (5)**, by $\mathcal{O}(\log T)$ using the kl-UCB indexes and the forced choice of the algorithm:

$\text{UCB}_k^j(t-1) \leq \text{UCB}_{k'}^j(t-1)$, and $\text{UCB}_k^j(t) > \text{UCB}_{k'}^j(t)$ when switching from k' to k ,

Sketch of the proof

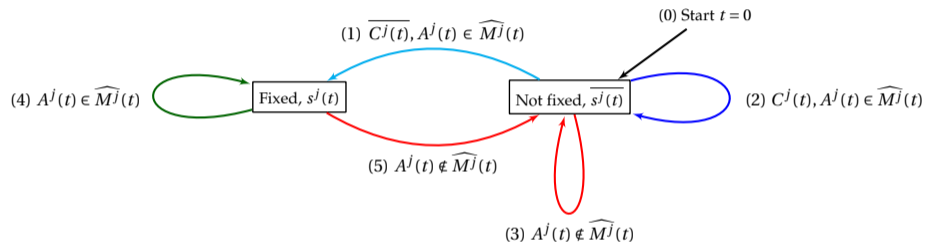
- 1 Bound the expected number of collisions by M times the number of collisions for non-fixed players,
- 2 Bound the expected number of transitions of type (3) and (5), by $\mathcal{O}(\log T)$ using the kl-UCB indexes and the forced choice of the algorithm:
$$\text{UCB}_k^j(t-1) \leq \text{UCB}_{k'}^j(t-1), \text{ and } \text{UCB}_k^j(t) > \text{UCB}_{k'}^j(t) \text{ when switching from } k' \text{ to } k,$$
- 3 Bound the expected length of a sequence in the non-fixed state by a constant,

Sketch of the proof

- ① Bound the expected number of collisions by M times the number of collisions for non-fixed players,
- ② Bound the expected number of transitions of type (3) and (5), by $\mathcal{O}(\log T)$ using the kl-UCB indexes and the forced choice of the algorithm:
$$\text{UCB}_k^j(t-1) \leq \text{UCB}_{k'}^j(t-1), \text{ and } \text{UCB}_k^j(t) > \text{UCB}_{k'}^j(t) \text{ when switching from } k' \text{ to } k,$$
- ③ Bound the expected length of a sequence in the non-fixed state by a constant,
- ④ So most of the times ($\mathcal{O}(T - \log T)$), players are fixed, and no collision happens when they are all fixed!

↪ See our paper for details!

Illustration of the proof



– Time in fixed state is $\mathcal{O}(\log T)$, and collisions are $\leq M$ collisions in fixed state $\implies \mathcal{O}(\log T)$ collisions.

– Suboptimal selections is $\mathcal{O}(\log T)$ also as $A^j(t+1)$ is always selected in $\widehat{M^j}(t)$ which is M -best at least $\mathcal{O}(T - \log T)$ (in average).

Experimental results

Experiments on Bernoulli problems $\mu \in [0, 1]^K$.

Illustration of the regret lower bound

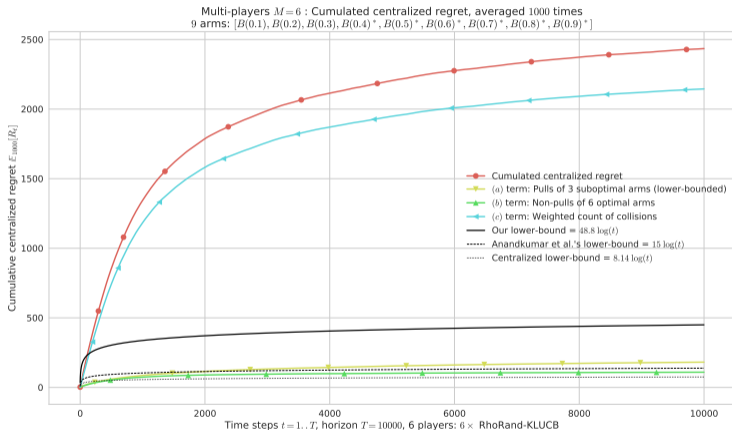


Figure 1: Any such lower bound is **very asymptotic**, usually not satisfied for small horizons. We can see the importance of the collisions!

Constant regret if $M = K$

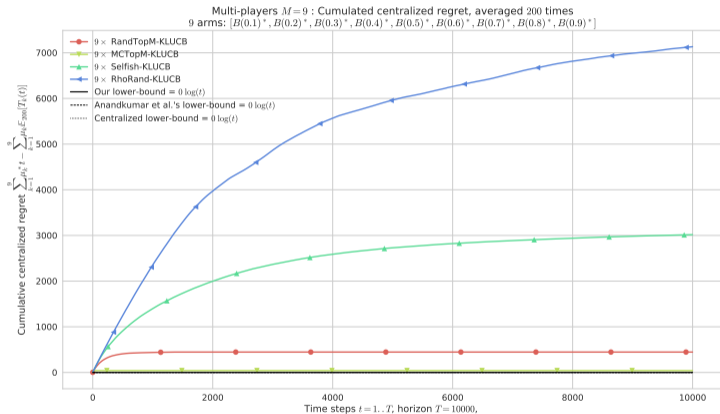


Figure 2: Regret, $M = 9$ players, $K = 9$ arms, horizon $T = 10000$, 200 repetitions. Only **RandTopM** and **MCTopM** achieve constant regret in this saturated case (proved).

Illustration of the regret of different algorithms

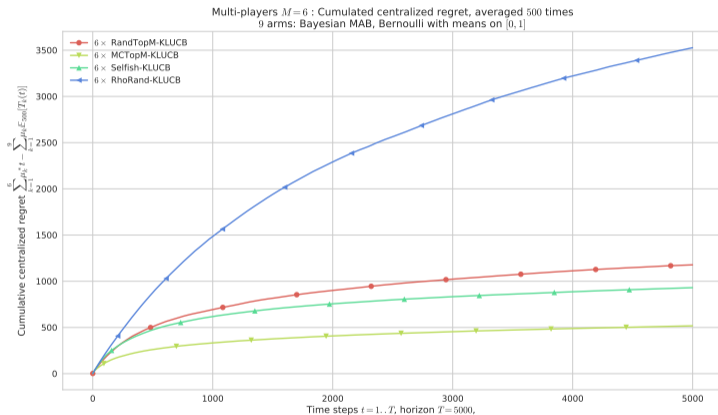


Figure 3: Regret, $M = 6$ players, $K = 9$ arms, horizon $T = 5000$, against 500 problems μ uniformly sampled in $[0, 1]^K$. Conclusion : **RhoRand** < **RandTopM** < **Selfish** < **MCTopM** in most cases.

Logarithmic number of collisions

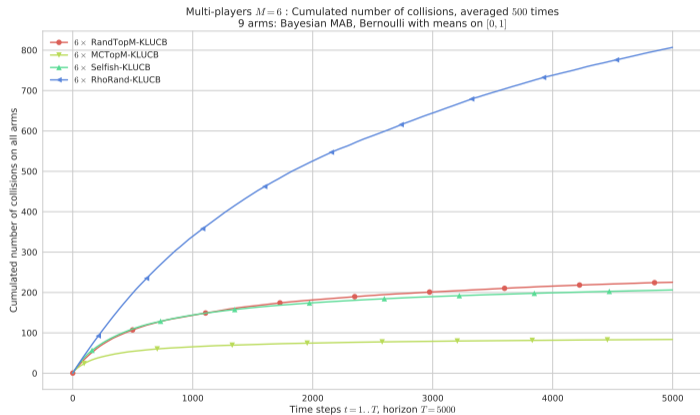


Figure 4: Cumulated number of collisions. Also **RhoRand** < **RandTopM** < **Selfish** < **MCTopM**.

Logarithmic number of arm switches

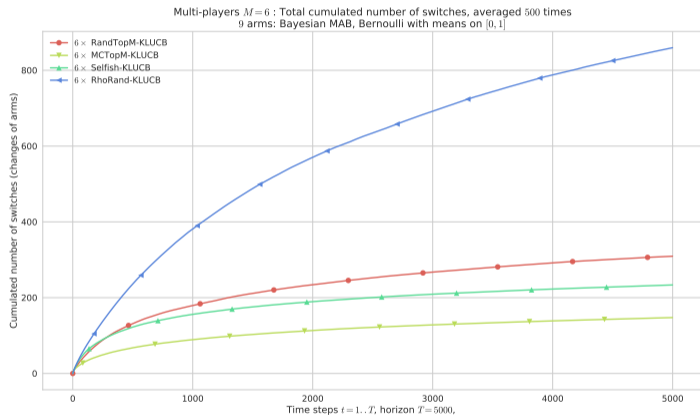


Figure 5: Cumulated number of arm switches. Again **RhoRand** < **RandTopM** < **Selfish** < **MCTopM**, but no guarantee for **RhoRand**. Bonus result: logarithmic arm switches for our algorithms!

Fairness

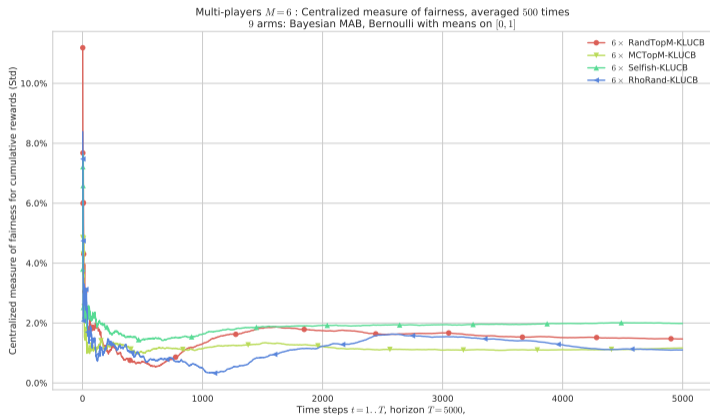


Figure 6: Measure of fairness among player. All 4 algorithms seem fair in average, but none is fair on a single run. It's quite hard to achieve both efficiency and single-run fairness!

A larger benchmark

Now I also want to compare more approaches.

- RhoRand, with UCB_1 or $kl\text{-}UCB$,
- RandTopM, with UCB_1 or $kl\text{-}UCB$,
- MCTopM, with UCB_1 or $kl\text{-}UCB$,
- Selfish, with UCB_1 or $kl\text{-}UCB$,
- a centralized agent (**not playing the same game, not fair to compare against it**), with UCB_1 or $kl\text{-}UCB$,
- three hand-tuned Musical-Chair algorithms,
- three variants of the SIC-MMAB algorithm (from [arXiv:1809.08151](https://arxiv.org/abs/1809.08151)), with UCB_1 , $kl\text{-}UCB$ and their proposal with $UCB\text{-}H$.

Comparison with other approaches (1/3)

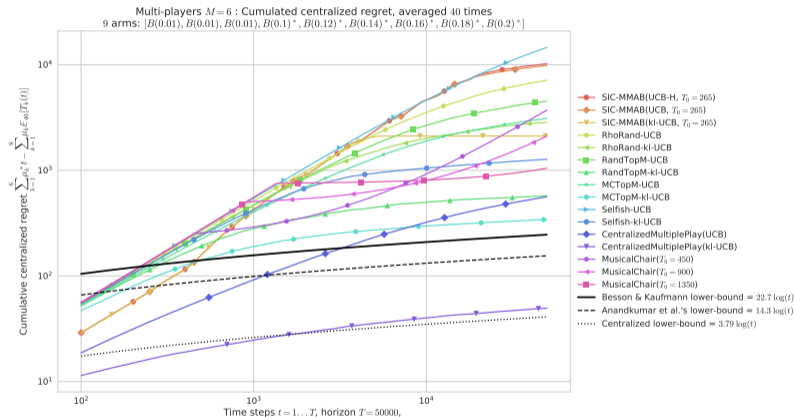


Figure 7: For $M = 6$ objects, MCTopM and RandTopM largely outperform SIC-MMAB and RhoRand.

Comparison with other approaches (2/3)

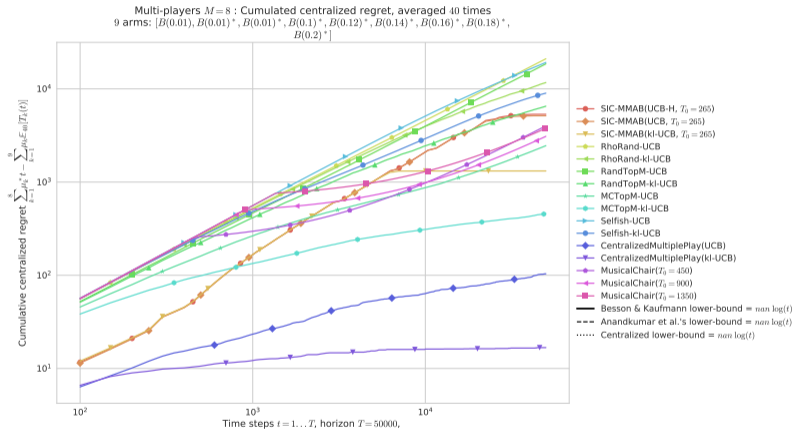


Figure 8: For $M = 8$ objects, MCTopM still outperforms SIC-MMAB for short term regret, but the constant in front of the $\log(T)$ term seems smaller for SIC-MMAB.

Comparison with other approaches (3/3)

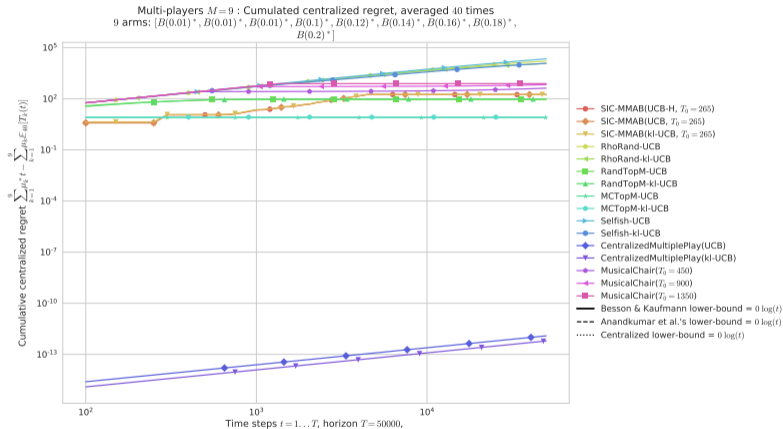


Figure 9: For $M = 9$ objects, MCTopM and RandTopM largely outperform all approaches, they have finite regret when the other don't. For our algorithm, $M = K$ is the easiest case: just orthogonalize and it's done!

Short summary of these benchmarks

In such experiments, and many more not showed here, I did the following observations:

- For any algorithm, the kl-UCB variant is uniformly better than the UCB_1 and UCB-H variant (obviously),
- Any decentralized approach is less efficient than the “cheating” centralized multiple-play approach,
- And for a fixed index policy, the following ordering on decentralized approaches can be observed (smaller means smaller regret, so a better algorithm):

$$\text{MCTopM} < \text{RandTopM} < \text{SIC-MMAB} < \text{Selfish} < \text{RhoRand}.$$

Other recent related works (1/2)

- Another recent article studied a similar problem.

Multi-user Communication Networks: A Coordinated Multi-armed Bandit Approach

Orly Avner, *Student Member, IEEE*, and Shie Mannor, *Senior Member, IEEE*

Abstract—Communication networks shared by many users are a widespread challenge nowadays. In this paper we address several aspects of this challenge simultaneously: learning unknown stochastic network characteristics, sharing resources with other users while keeping coordination overhead to a minimum. The proposed solution combines Multi-Armed Bandit learning with a lightweight signalling-based coordination scheme, and ensures convergence to a stable allocation of resources. Our work considers single-user level algorithms for two scenarios: an unknown fixed number of users, and a dynamic number of users. Analytic performance guarantees, proving convergence to stable marriage configurations, are presented for both setups. The algorithms are designed based on a system-wide perspective, rather than focusing on single user welfare. Thus, maximal resource utilization is ensured. An extensive experimental analysis covers convergence to a stable configuration as well as reward maximization. Experiments are carried out over a wide range of setups, demonstrating the advantages of our approach over existing state-of-the-art methods.

I. INTRODUCTION

THE world of modern multi-user communication networks poses many challenges that serve as an inspiration for our work. We focus on distributed setups such as cognitive radio networks (CRNs) that consist of several users accessing a set of communication channels. The users' goal is to make the best possible use of network resources.

Radios with enhanced capabilities such as spectrum sensing, memory and computational power can identify and use "gaps" in transmissions of licensed traditional radios, thus increasing utilization. From an algorithmic point of view, this framework gives rise to several interesting questions due to its dynamic, stochastic, distributed nature. Over the last decade this challenging assortment of problems has gained considerable attention from researchers and engineers [2], [3]. Both theoretical and practical issues have been addressed, along with the necessary increase of regulatory support [4].

B. Multi-armed bandits

Multi-armed bandits (MABs) are a well-studied framework from the world of machine learning. They model a sequential decision making problem in which a user repeatedly chooses one of K actions in order to maximize her acquired reward. The characteristics of the actions (also known as arms) are initially unknown, and learning to identify the best action needs to be balanced with reward maximization, in what is known as the exploration-exploitation dilemma. MABs have attracted much interest due to the wide range of applications they capture, combined with their relative simplicity, from both algorithmic and analytic points of view. Several papers propose solutions for the stochastic MAB problem [5], [7].

.04875v1 [cs.LG] 14 Aug 2018

Other recent related works (1/2)

- Another recent article studied a similar problem.
- Implementing their algorithms should be easy, but their model is quite different:
 - Objects can choose to not communicate, it is denoted by choosing arm 0 and not k in $\{1, \dots, K\}$,
 - $\triangle!$ But more importantly, objects can send some bits of data directly to each other...
 - So it's a little bit more complicated than my (simple) model.

¹I will try to code their model in my framework, see [GitHub.com/SMPyBandits/SMPyBandits/issues/139](https://github.com/SMPyBandits/SMPyBandits/issues/139)

Other recent related works (1/2)

- Another recent article studied a similar problem.
- Implementing their algorithms should be easy, but their model is quite different:
 - Objects can choose to not communicate, it is denoted by choosing arm 0 and not k in $\{1, \dots, K\}$,
 - \triangle But more importantly, objects can send some bits of data directly to each other...
 - So it's a little bit more complicated than my (simple) model.
- \implies I will¹ work on this more in the near future!

“Multi-user Communication Networks: A Coordinated Multi-armed Bandit Approach”, by Orly Avner & Shie Mannor, [arXiv:1808.04875](https://arxiv.org/abs/1808.04875)

¹I will try to code their model in my framework, see [GitHub.com/SMPyBandits/SMPyBandits/issues/139](https://github.com/SMPyBandits/SMPyBandits/issues/139)

Other recent related works (2/2)

- And another recent article also studied a similar problem.

Multiplayer bandits without observing collision information

Gábor Lugosi^{†*} Abbas Mehrabian[§]

August 28, 2018

Abstract

We study multiplayer stochastic multi-armed bandit problems in which the players cannot communicate, and if two or more players pull the same arm, a collision occurs and the involved players receive zero reward. We consider two feedback models: a model in which the players can observe whether a collision has occurred, and a more difficult setup when no collision information is available. We give the first theoretical guarantees for the second model: an algorithm with a logarithmic regret, and an algorithm with a square-root regret type that does not depend on the gaps between the means. For the first model, we give the first square-root regret bounds that do not depend on the gaps. Building on these ideas, we also give an algorithm for reaching approximate Nash equilibria quickly in stochastic anti-coordination games.

1 Introduction

The stochastic multi-armed bandit problem is a well-studied problem of machine learning: consider an agent that has to choose among several actions in each round of a game. To each action i is associated a real-valued parameter μ_i . Whenever the player performs the i -th action, she receives a random reward with mean μ_i . If the player knew the means associated to the actions before starting the game, she would play an action with the highest mean during all rounds. The problem is to design a strategy for the player to maximize her reward in the setting where she

arXiv:1808.08416v1 [cs.LG] 25 Aug 2018

Other recent related works (2/2)

- And another recent article also studied a similar problem.
- A very strong work from a theoretical point of view, but completely impractical even for simulations.
- Their analysis says that their algorithm can be efficient only after at least $T_{1,2}$ steps of uniform exploration (i.e., linear regret).

Other recent related works (2/2)

- And another recent article also studied a similar problem.
- A very strong work from a theoretical point of view, but completely impractical even for simulations.
- Their analysis says that their algorithm can be efficient only after at least $T_{1,2}$ steps of uniform exploration (i.e., linear regret).
- On very easy problems with minimal gap between arms of $\Delta_{\min} = 0.1$ (rewards in $[0, 1]$), and very small horizons, small M and K , $T_{1,2}$ is computed as:
 - For $M = 2$ and $K = 2$, and $T = 100$, $T_{1,2} = 198214307$,
 - For $M = 2$ and $K = 2$, and $T = 1000$, $T_{1,2} = 271897030$,
 - For $M = 2$ and $K = 3$, and $T = 100$, $T_{1,2} = 307052623$,
 - For $M = 2$ and $K = 5$, and $T = 100$, $T_{1,2} = 532187397$.
- ⚠ That's just unreasonable!

Other recent related works (2/2)

- And another recent article also studied a similar problem.
- A very strong work from a theoretical point of view, but completely impractical even for simulations.
- After discussing with the author, I tried using a much smaller value for their constant g (1 instead of 128), and their algorithm is still very much asymptotic in practice, even on very simple problems!
- \Rightarrow I will² work on this more in the near future!

“Multiplayer Bandits Without Observing Collision Information”, by Gabor Lugosi & Abbas Mehrabian, [arXiv:1808.08416](https://arxiv.org/abs/1808.08416)

²I already added their first algorithm in my framework, see [GitHub.com/SMPyBandits/SMPyBandits/issues/141](https://github.com/SMPyBandits/SMPyBandits/issues/141)

Sum up

- In a wireless network with an i.i.d. background traffic in K channels,
- M devices can use both sensing and acknowledgement feedback, to learn the most free channels and to find orthogonal configurations.

Sum up

- In a wireless network with an i.i.d. background traffic in K channels,
- M devices can use both sensing and acknowledgement feedback, to learn the most free channels and to find orthogonal configurations.

We showed

- Decentralized bandit algorithms can solve this problem,
- We have a lower bound for any decentralized algorithm,
- And we proposed an order-optimal algorithm, based on kl-UCB and an improved Musical Chair scheme, MCTopM.

Future works

- Implement and test this on real-world radio devices?

↪ Yes!

Demo presented at the ICT 2018 conference! (Saint-Malo, France)

Future works

- Implement and test this on real-world radio devices?

↪ Yes!

Demo presented at the ICT 2018 conference! (Saint-Malo, France)

- Remove hypothesis that objects know M ? (easy)
- Allow arrival/departure of objects? (harder)
- Non-stationarity of background traffic? (much harder)

Future works

- Implement and test this on real-world radio devices?
↳ Yes!
Demo presented at the ICT 2018 conference! (Saint-Malo, France)
- Remove hypothesis that objects know M ? (easy)
- Allow arrival/departure of objects? (harder)
- Non-stationarity of background traffic? (much harder)
- Extend to more objects (i.e., when $M > K$) ?
“Large-scale” IoT model, with (e.g., ZigBee networks), or without sensing (e.g., LoRaWAN networks).
↳ objects should no longer communicate at every time step!

Future works

- Implement and test this on real-world radio devices?
↳ Yes!
Demo presented at the ICT 2018 conference! (Saint-Malo, France)
- Remove hypothesis that objects know M ? (easy)
- Allow arrival/departure of objects? (harder)
- Non-stationarity of background traffic? (much harder)
- Extend to more objects (i.e., when $M > K$) ?
“Large-scale” IoT model, with (e.g., ZigBee networks), or without sensing (e.g., LoRaWAN networks).
↳ objects should no longer communicate at every time step!

Thanks!

Thanks! 😊

Any question?

Appendix

- An heuristic for the “IoT” case (no sensing): the Selfish algorithm,
- Success and failures case for Selfish.

An heuristic, Selfish

For the harder feedback model, without sensing.

- ① An heuristic,
- ② Problems with Selfish,
- ③ Illustration of failure cases.

Selfish heuristic I

Selfish decentralized approach = device don't use sensing:

Selfish

Use UCB_1 (or $kl\text{-}UCB$) indexes on the (non i.i.d.) rewards $r^j(t)$ and not on the sensing $Y_{Aj(t)}(t)$.

Ref: [Bonnefoi & Besson et al, 2017]

Works fine...

- More suited to model IoT networks,
- Use less information, and don't know the value of M : we expect Selfish to not have stronger guarantees.
- It works fine in practice!

Selfish heuristic II

But why would it work?

- Sensing feedback were i.i.d., so using UCB_1 to learn the μ_k makes sense,
- But collisions make the rewards not i.i.d. !
- Adversarial algorithms should be more appropriate here,
- But empirically, Selfish works much better with kl-UCB than, e.g., Exp3...

Works fine...

- Except... when it fails drastically! 😞
- In small problems with M and $K = 2$ or 3 , we found small probability of failures (i.e., linear regret), and this prevents from having a generic upper bound on the regret for Selfish.

Illustration of failing cases for Selfish

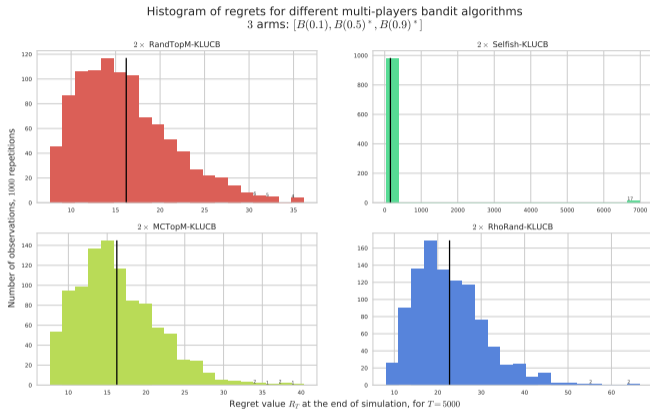


Figure 10: Regret for $M = 2$, $K = 3$, $T = 5000$, 1000 repetitions and $\mu = [0.1, 0.5, 0.9]$. Axis x is for regret (different scale for each), and **Selfish** have a small probability of failure (17/1000 cases of $R_T \gg \log T$). The regret for the three other algorithms is very small for this “easy” problem.