

Multi-Player Bandits Revisited

Decentralized Multi-Player Multi-Arm Bandits

Lilian Besson

Advised by Christophe Moy Émilie Kaufmann

PhD Student

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

SequeL Seminar - 22 December 2017



Motivation

We control some communicating devices, they want to access to a single base station.

- 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 sequential algorithm!

Outline

- 2 Our model: 3 different feedback levels
- 3 Decomposition and lower bound on regret
- 4 Quick reminder on single-player MAB algorithms
- 5 Two new multi-player decentralized algorithms
- 6 Upper bounds on regret for MCTopM
- 7 Experimental results
- 8 An heuristic (Selfish), and disappointing results
- 9 Conclusion

Outline and reference

- 2 Our model: 3 different feedback levels
- 3 Decomposition and lower bound on regret
- 4 Quick reminder on single-player MAB algorithms
- 5 Two new multi-player decentralized algorithms
- 6 Upper bounds on regret for MCTopM
- 7 Experimental results
- 8 An heuristic (Selfish), and disappointing results
- 9 Conclusion

This is based on our latest article:

- “Multi-Player Bandits Models Revisited”, Besson & Kaufmann.
[arXiv:1711.02317](https://arxiv.org/abs/1711.02317)

Our model

- K radio channels (e.g., 10) (known)
- Discrete and synchronized time $t \geq 1$. Every time frame t is:

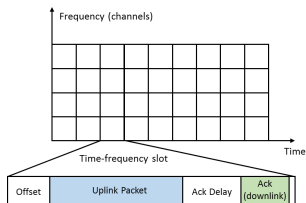


Figure 1: Protocol in time and frequency, with an **Acknowledgement**.

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..

Two variants : with or without sensing

- ① With sensing: Device first senses for presence of Primary Users (background traffic), then use Ack to detect collisions.
Model the “classical” Opportunistic Spectrum Access problem. Not exactly suited for Internet of Things, but can model ZigBee, and can be analyzed mathematically...

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 (background traffic), then use Ack to detect collisions.
 Model the “classical” Opportunistic Spectrum Access problem. Not exactly suited for Internet of Things, but can model ZigBee, and can be analyzed mathematically...
- ② Without sensing: same background traffic, but cannot sense, so only Ack is used. More suited for “IoT” networks like LoRa or SigFox (Harder to analyze mathematically).

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

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

- 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 **joint** 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 joint $Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$,
 \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 joint $Y_{A^j(t),t} \times \mathbb{1}(\overline{C^j(t)})$,
 \hookrightarrow Unlicensed protocols (ex. LoRaWAN), harder to analyze !

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

Goal

Problem

- 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.

Goal

Problem

- 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.

Decentralized reinforcement learning optimization!

- Max transmission rate \equiv max cumulated rewards

$$\max_{\text{algorithm } A} \sum_{t=1}^T \sum_{j=1}^M r_{A(t)}^j.$$

- Each player wants to maximize its cumulated reward,
- With no central control, and no exchange of information,
- Only possible if : each player converges to one of the M best arms, orthogonally (without collisions).

Centralized regret

A measure of success

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

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

Centralized regret

A measure of success

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

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

Centralized regret

A measure of success

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

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

Two directions of analysis

- Clearly $R_T = \mathcal{O}(T)$, but we want a sub-linear regret, as small as possible!

Centralized regret

A measure of success

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

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

Two directions of analysis

- Clearly $R_T = \mathcal{O}(T)$, but we want a sub-linear regret, as small as possible!
- How good a decentralized algorithm can be in this setting?
 \hookrightarrow Lower Bound on regret, for any algorithm !
- How good is my decentralized algorithm in this setting?

Centralized regret

A measure of success

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

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

Two directions of analysis

- Clearly $R_T = \mathcal{O}(T)$, but we want a sub-linear regret, as small as possible!
- How good a decentralized algorithm can be in this setting?
 - ↪ Lower Bound on regret, for any algorithm !
- How good is my decentralized algorithm in this setting?
 - ↪ Upper Bound on regret, for one algorithm !

Lower bound

- ① Decomposition of regret in 3 terms,
- ② Asymptotic lower bound of one term,
- ③ And for regret,
- ④ Sketch of proof,
- ⑤ Illustration.

Decomposition on regret

Decomposition

For any algorithm, decentralized or not, we have

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

Small regret can be attained if...

Decomposition on 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 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 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**).

Asymptotic Lower Bound on regret

3 terms to lower bound...

- The first term for sub-optimal arms selections is lower bounded asymptotically,

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

using technical information theory tools (Kullback-Leibler divergence, entropy),

Asymptotic Lower Bound on regret

3 terms to lower bound...

- The first term for sub-optimal arms selections is lower bounded asymptotically,

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

using technical information theory tools (Kullback-Leibler divergence, entropy),

- And we lower bound the rest (including collisions) by... 0

$$T - \mathbb{E}_\mu[T_k(T)] \geq 0 \quad \text{and} \quad \mathbb{E}_\mu[\mathcal{C}_k(T)] \geq 0,$$

☹ we should be able to do better!

Asymptotic Lower Bound on regret

Theorem 1

[Besson & Kaufmann, 2017]

- For any uniformly efficient decentralized policy, and any non-degenerated problem $\boldsymbol{\mu}$,

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

Asymptotic Lower Bound on regret

Theorem 1

[Besson & Kaufmann, 2017]

- For any uniformly efficient decentralized policy, and any non-degenerated problem $\boldsymbol{\mu}$,

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

Where $\text{kl}(x, y) := x \log(\frac{x}{y}) + (1-x) \log(\frac{1-x}{1-y})$ is the binary Kullback-Leibler divergence.

Remarks

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

Illustration of the Lower Bound on regret

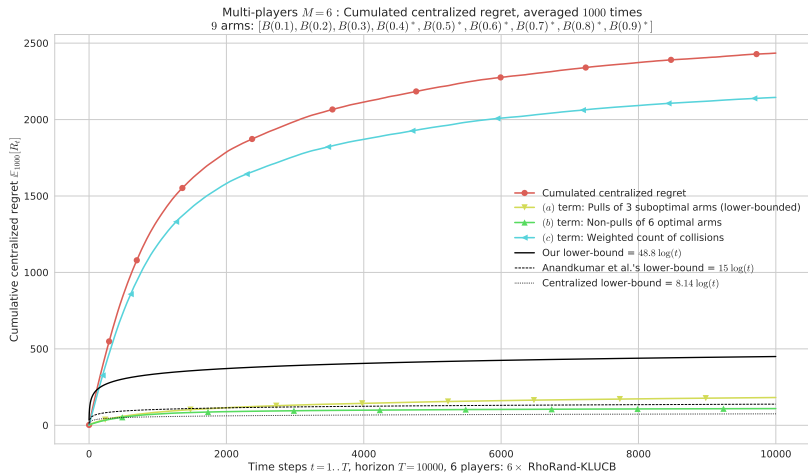


Figure 2: Any such lower bound is **very asymptotic**, usually not satisfied for small horizons. We can see the importance of the collisions!

Sketch of the proof

- Like for single-player bandit, focus on $\mathbb{E}_\mu[T_k^j(T)]$ expected number of selections of any sub-optimal arm k .
- Same information-theoretic tools, using a “change of law” lemma.
Ref: [Garivier et al, 2016]
- It improved the state-of-the-art because of our decomposition, not because of new tools.

↔ See our paper for details!

Single-player MAB algorithms

- 1 Index-based MAB deterministic policies,
- 2 Upper Confidence Bound algorithm : UCB_1 ,
- 3 Kullback-Leibler UCB algorithm : $kl\text{-UCB}$.

Upper Confidence Bound algorithm (UCB₁)

The device keep t number of sent packets, $T_k(t)$ selections of channel k , $X_k(t)$ successful transmissions in channel k .

- ① For the first K steps ($t = 1, \dots, K$), try each channel once.
- ② Then for the next steps $t > K$:

- Compute the index $g_k(t) := \underbrace{\frac{X_k(t)}{T_k(t)}}_{\text{Mean } \hat{\mu}_k(t)} + \underbrace{\sqrt{\frac{\log(t)}{2 T_k(t)}}}_{\text{Upper Confidence Bound}}$
- Choose channel $A(t) = \arg \max_k g_k(t)$,
- Update $T_k(t+1)$ and $X_k(t+1)$.

References: [Lai & Robbins, 1985], [Auer et al, 2002], [Bubeck & Cesa-Bianchi, 2012]

Kullback-Leibler UCB algorithm (kl-UCB)

The device keep t number of sent packets, $T_k(t)$ selections of channel k , $X_k(t)$ successful transmissions in channel k .

- ① For the first K steps ($t = 1, \dots, K$), try each channel once.
- ② Then for the next steps $t > K$:
 - Compute the index $g_k(t) := \sup_{q \in [a, b]} \left\{ q : \text{kl} \left(\frac{X_k(t)}{T_k(t)}, q \right) \leq \frac{\log(t)}{T_k(t)} \right\}$
 - Choose channel $A(t) = \arg \max_k g_k(t)$,
 - Update $T_k(t+1)$ and $X_k(t+1)$.

Kullback-Leibler UCB algorithm (kl-UCB)

The device keep t number of sent packets, $T_k(t)$ selections of channel k , $X_k(t)$ successful transmissions in channel k .

- ① For the first K steps ($t = 1, \dots, K$), try each channel once.
- ② Then for the next steps $t > K$:
 - Compute the index $g_k(t) := \sup_{q \in [a, b]} \left\{ q : \text{kl} \left(\frac{X_k(t)}{T_k(t)}, q \right) \leq \frac{\log(t)}{T_k(t)} \right\}$
 - Choose channel $A(t) = \arg \max_k g_k(t)$,
 - Update $T_k(t+1)$ and $X_k(t+1)$.

Why bother? kl-UCB is proved to be more efficient than UCB₁, and asymptotically optimal for single-player stochastic bandit.

References: [Garivier & Cappé, 2011], [Cappé & Garivier & Maillard & Munos & Stoltz, 2013]

Multi-player decentralized algorithms

- ① Common building blocks of previous algorithms,
- ② First proposal: RandTopM,
- ③ Second proposal: MCTopM,
- ④ Algorithm and illustration.

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 $C^j(t)$).

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 $C^j(t)$).

Many different proposals for decentralized learning policies

- Recent: MEGA and Musical Chair, [Avner & Mannor, 2015], [Shamir et al, 2016]
- State-of-the-art: RhoRand policy and variants. [Anandkumar et al, 2011]

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 $C^j(t)$).

Many different proposals for decentralized learning policies

- Recent: MEGA and Musical Chair, [Avner & Mannor, 2015], [Shamir et al, 2016]
- State-of-the-art: RhoRand policy and variants. [Anandkumar et al, 2011]

Our proposals: [Besson & Kaufmann, 2017]

- With sensing: RandTopM and MCTopM are sort of mixes between RhoRand and Musical Chair, using UCB indexes or more efficient index policies (kl-UCB),
- Without sensing: Selfish use a UCB index directly on the

A first decentralized algorithm

```

1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$ 
2 for  $t = 0, \dots, T - 1$  do
3   | if  $A^j(t) \notin \widehat{M}^j(t)$  or  $C^j(t)$  then
4     |  $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t))$  // randomly switch
5     | else
6       |  $A^j(t+1) = A^j(t)$  // stays on the same arm
7     | end
8     | Play arm  $A^j(t+1)$ , get new observations (sensing and collision),
9     | Compute the indices  $g_k^j(t+1)$  and set  $\widehat{M}^j(t+1)$  for next step.
10 end

```

Algorithm 1: A first decentralized learning policy (for a fixed underlying index policy g^j). The set $\widehat{M}^j(t)$ is the M best arms according to indexes $g^j(t)$.

The RandTopM algorithm

```

1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$ 
2 for  $t = 0, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then
4     if  $C^j(t)$  then // collision
5        $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t))$  // randomly switch
6     else // aim arm with smaller UCB at  $t - 1$ 
7        $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : g_k^j(t-1) \leq g_{A^j(t)}^j(t-1)\})$ 
8     end
9   else
10     $A^j(t+1) = A^j(t)$  // stays on the same arm
11  end
12  Play arm  $A^j(t+1)$ , get new observations (sensing and collision),
13  Compute the indices  $g_k^j(t+1)$  and set  $\widehat{M}^j(t+1)$  for next step.
14 end

```


The MCTopM algorithm

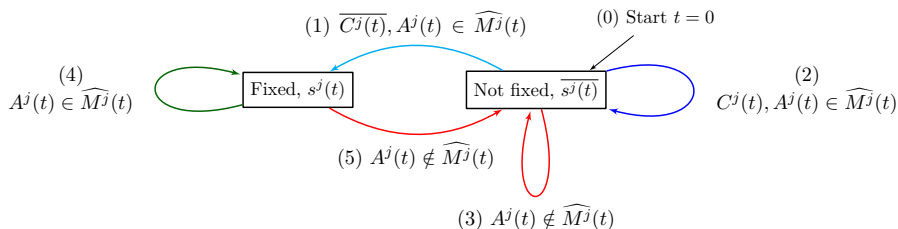


Figure 3: Player j using MCTopM, represented as “state machine” with 5 transitions. Taking one of the five transitions means playing one round of Algorithm MCTopM, to decide $A^j(t + 1)$ using information of previous steps.

The MCTopM algorithm

```
1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$  and  $s^j(1) = \text{False}$ 
2 for  $t = 0, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : g_k^j(t-1) \leq g_{A^j(t)}^j(t-1)\})$  // not empty
5      $s^j(t+1) = \text{False}$  // aim at an arm with smaller UCB at  $t-1$ 
6   else if  $C^j(t)$  and  $\overline{s^j(t)}$  then // collision and not fixed
7      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t))$  // transition (2)
8      $s^j(t+1) = \text{False}$ 
9   else // transition (1) or (4)
10     $A^j(t+1) = A^j(t)$  // stay on the previous arm
11     $s^j(t+1) = \text{True}$  // 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  $g_k^j(t+1)$  and set  $\widehat{M}^j(t+1)$  for next step.
15 end
```

Regret upper bound

- 1 Theorem,
- 2 Remarks,
- 3 Idea of the proof.

Regret upper bound for MCTopM

Theorem 2

[Besson & Kaufmann, 2017]

- If all M players use MCTopM with kl-UCB, then for any non-degenerated problem $\boldsymbol{\mu}$, there exists a problem dependent constant $G_{M,\boldsymbol{\mu}}$, such that the regret satisfies:

$$R_T(\boldsymbol{\mu}, M, \rho) \leq G_{M,\boldsymbol{\mu}} \log(T) + o(\log T).$$

Regret upper bound for MCTopM

Theorem 2

[Besson & Kaufmann, 2017]

- If all M players use MCTopM with kl-UCB, then for any non-degenerated problem $\boldsymbol{\mu}$, there exists a problem dependent constant $G_{M,\boldsymbol{\mu}}$, such that the regret satisfies:

$$R_T(\boldsymbol{\mu}, M, \rho) \leq G_{M,\boldsymbol{\mu}} \log(T) + o(\log T).$$

How?

- Decomposition of regret controlled with two terms,
- Control both terms, both are logarithmic:
 - Suboptimal selections with the “classical analysis” on kl-UCB indexes
 - Collisions are harder to control...

Regret upper bound for MCTopM

Remarks

- Hard to prove, we had to carefully design the MCTopM algorithm to conclude the proof,

Regret upper bound for MCTopM

Remarks

- Hard to prove, we had to carefully design the MCTopM algorithm to conclude the proof,
- The constant $G_{M,\mu}$ scales as M^3 , way better than RhoRand's constant scaling as $M \binom{2M-1}{M}$,

Regret upper bound for MCTopM

Remarks

- Hard to prove, we had to carefully design the MCTopM algorithm to conclude the proof,
- The constant $G_{M,\mu}$ scales as M^3 , way better than RhoRand's constant scaling as $M \binom{2M-1}{M}$,
- We also minimize the number of channel switching:
interesting as changing arm costs energy in radio systems,

Regret upper bound for MCTopM

Remarks

- Hard to prove, we had to carefully design the MCTopM algorithm to conclude the proof,
- The constant $G_{M,\mu}$ scales as M^3 , way better than RhoRand's constant scaling as $M \binom{2M-1}{M}$,
- 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 !

Regret upper bound for MCTopM

Remarks

- Hard to prove, we had to carefully design the MCTopM algorithm to conclude the proof,
- The constant $G_{M,\mu}$ scales as M^3 , way better than RhoRand's constant scaling as $M \binom{2M-1}{M}$,
- 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 !
- Not yet possible to know what is the best possible control of collisions...

Sketch of the proof

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

Sketch of the proof

- 1 Bound the expected number of collisions by M times the number of collisions for non-sitted 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: $g_k^j(t-1) \leq g_{k'}^j(t-1)$, and $g_k^j(t) > g_{k'}^j(t)$ 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-sitted 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: $g_k^j(t-1) \leq g_{k'}^j(t-1)$, and $g_k^j(t) > g_{k'}^j(t)$ when switching from k' to k ,
- 3 Bound the expected length of a sequence in the non-sitted state by a constant,

Sketch of the proof

- 1 Bound the expected number of collisions by M times the number of collisions for non-sitted 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: $g_k^j(t-1) \leq g_{k'}^j(t-1)$, and $g_k^j(t) > g_{k'}^j(t)$ when switching from k' to k ,
- 3 Bound the expected length of a sequence in the non-sitted state by a constant,
- 4 So most of the times ($\mathcal{O}(T - \log T)$), players are sitted, and no collision happens when they are all sitted!

↔ See our paper for details!

Experimental results

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

- 1 Illustration of regret for a single problem and $M = K$,
- 2 Regret for uniformly sampled problems and $M < K$,
- 3 Logarithmic number of collisions,
- 4 Logarithmic number of arm switches,
- 5 Fairness?

Constant regret if $M = K$

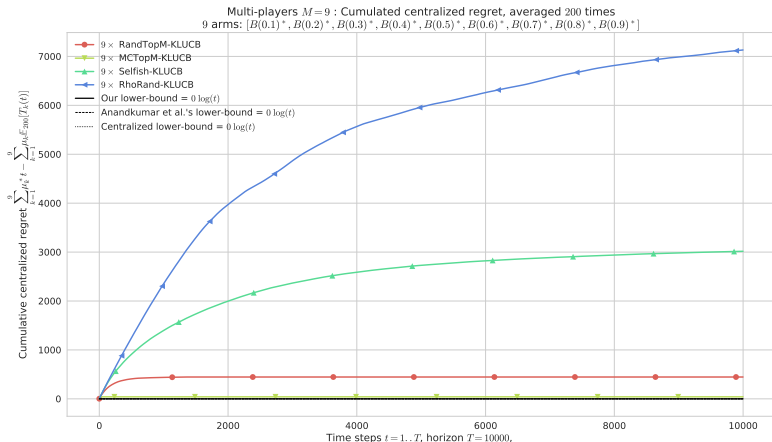


Figure 4: Regret, $M = 9$ players, $K = 9$ arms, horizon $T = 10000$, 200 repetitions. Only **RandTopM** and **MCTopM** achieve constant regret in this saturated case (proved).

Illustration of regret of different algorithms

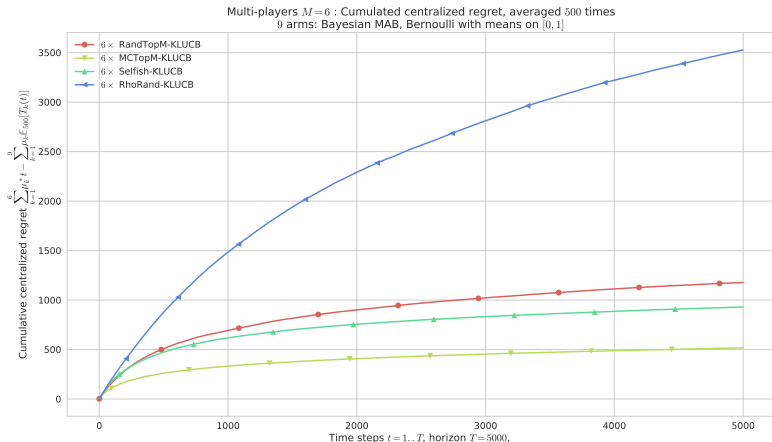


Figure 5: Regret, $M = 6$ players, $K = 9$ arms, horizon $T = 5000$, against 500 problems μ uniformly sampled in $[0, 1]^K$.

Conclusion : RhoRand < RandTopM < Selfish < MCTopM in most cases.

Logarithmic number of collisions

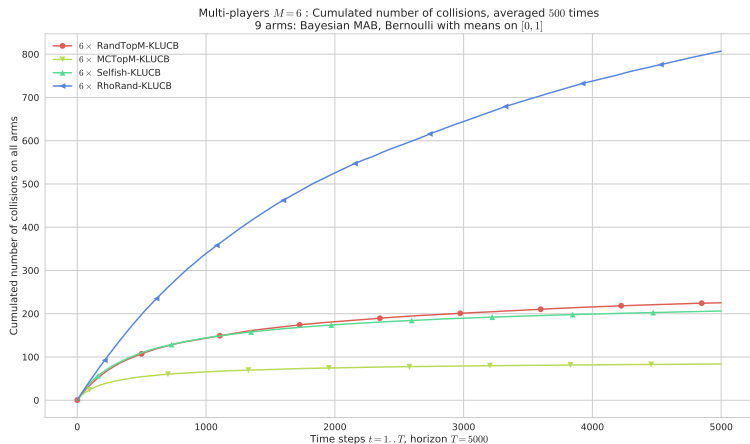


Figure 6: Cumulated number of collisions. Also $\text{RhoRand} < \text{RandTopM} < \text{Selfish} < \text{MCTopM}$ in most cases.

Logarithmic number of arm switches

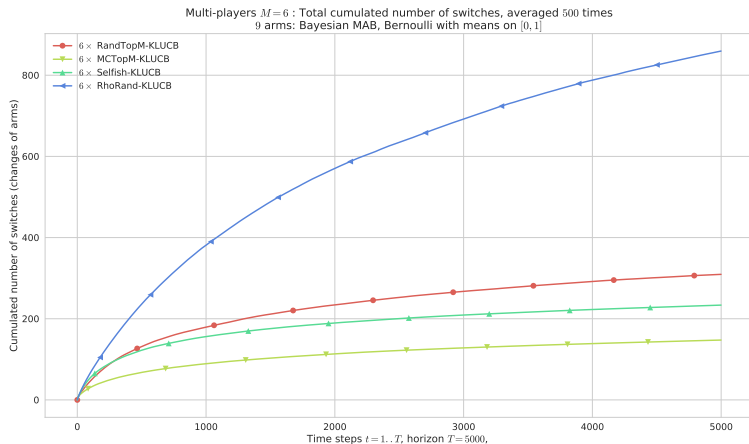


Figure 7: Cumulated number of arm switches. Again $\text{RhoRand} < \text{RandTopM} < \text{Selfish} < \text{MCTopM}$, but no guarantee for RhoRand .

Fairness

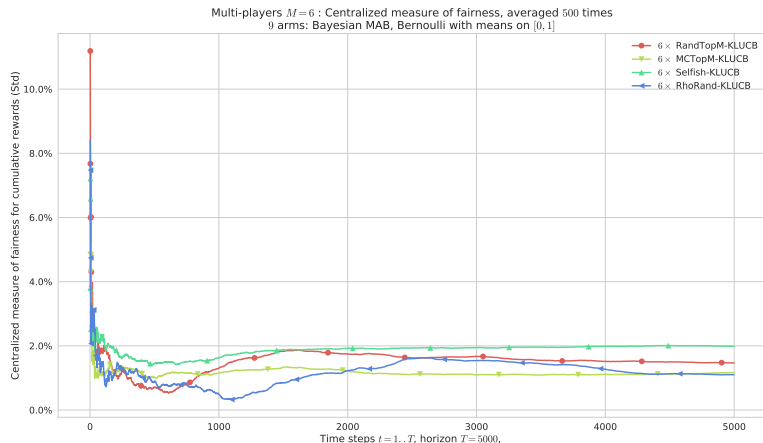


Figure 8: Measure of fairness among player. All 4 algorithms seem fair in average, but none is fair on a single run. It's quite hard to achieve both efficiency and single-run fairness!

An heuristic, Selfish

For the harder feedback model, without sensing.

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

The Selfish heuristic I

The Selfish decentralized approach = device don't use sensing, just learn on the reward (acknowledgement or not, $r^j(t)$).

Reference: [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!

The Selfish heuristic II

But why would it work?

- Sensing was i.i.d. so using UCB_1 to learn the μ_k makes sense,
- But collisions are not i.i.d.,
- Adversarial algorithms are more appropriate here,
- But empirically, Selfish with UCB_1 or kl -UCB works much better than, e.g., Exp3...

Illustration of failing cases for Selfish

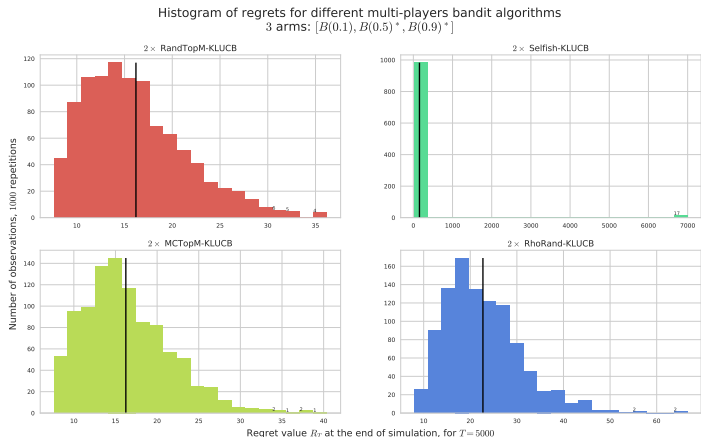


Figure 9: Regret for $M = 2$ players, $K = 3$ arms, horizon $T = 5000$, 1000 repetitions and $\boldsymbol{\mu} = [0.1, 0.5, 0.9]$. Axis x is for regret (different scale for each), and **Selfish** have a small probability of failure (17/1000 cases of $R_T \gg \log T$). The regret for the three other algorithms is very small for this

Sum-up

Wait, what was the problem ?

- MAB algorithms have guarantees for i.i.d. settings,
- But here the collisions cancel the i.i.d. hypothesis...
- Not easy to obtain guarantees in this mixed setting (i.i.d. emissions process, “game theoretic” collisions).

Sum-up

Wait, what was the problem ?

- MAB algorithms have guarantees for i.i.d. settings,
- But here the collisions cancel the i.i.d. hypothesis...
- Not easy to obtain guarantees in this mixed setting (i.i.d. emissions process, “game theoretic” collisions).

Theoretical results

- With sensing (“OSA”), we obtained strong results: a lower bound, and an order-optimal algorithm,
- But without sensing (“IoT”), it is harder... our heuristic Selfish usually works but can fail!

Other directions of future work

Conclude the Multi-Player OSA analysis

- Remove hypothesis that objects know M ,
- Allow arrival/departure of objects,
- Non-stationarity of background traffic etc
- More realistic emission model: maybe driven by number of packets in a whole day, instead of emission probability.

Other directions of future work

Conclude the Multi-Player OSA analysis

- Remove hypothesis that objects know M ,
- Allow arrival/departure of objects,
- Non-stationarity of background traffic etc
- More realistic emission model: maybe driven by number of packets in a whole day, instead of emission probability.

Extend to more objects $M > K$

- Extend the theoretical analysis to the large-scale IoT model, first with sensing (e.g., models ZigBee networks), then without sensing (e.g., LoRaWAN networks).

Conclusion I

- 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

Conclusion II

But more work is still needed... 😞

- Theoretical guarantees are still missing for the “IoT” model (without sensing), and can be improved (slightly) for the “OSA” model (with sensing).
- Maybe study other emission models...
- Implement and test this on real-world radio devices
↔ demo (in progress) for the ICT 2018 conference!

Thanks! 😊

Any question or idea ?