

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

ALT Conference – 08-04-2018



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

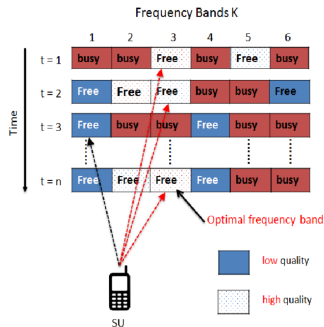
- Maintain a **good Quality of Service**.
- **With no centralized control** as it costs network overhead.

How?

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

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.*.

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 .

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 : $C^j(t) = \mathbb{1}(\text{alone on arm } A^j(t))$.
 $\hookrightarrow r^j(t)$ **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,
 \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 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.*

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?*
 \hookrightarrow **Lower Bound** on the regret, for **any** algorithm !
- *How good is my decentralized algorithm in this setting?*
 \hookrightarrow **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.

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

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

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

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

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

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

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 $\text{kl}(x, y) := \mathcal{KL}(\mathcal{B}(x), \mathcal{B}(y)) = x \log(\frac{x}{y}) + (1-x) \log(\frac{1-x}{1-y})$ 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**...

Ref: [Anantharam et al, 1987]

↪ “**price of non-coordination**” = M = nb of player?

- Improved state-of-the-art lower bound, but still not perfect: collisions should also be controlled!

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

$$\text{UCB}_k^j(t) := \sup_{q \in [a, b]} \left\{ q : \text{kl} \left(\frac{S_k^j(t)}{T_k^j(t)}, q \right) \leq \frac{\log(t)}{T_k^j(t)} \right\}, \text{ Ref: [Garivier \& Cappé, 2011]}$$
- Choose channel $A^j(t) = \arg \max_k \text{UCB}_k^j(t)$,
- Update $T_k^j(t+1)$ and $S_k^j(t+1)$.

kl-UCB is asymptotically optimal for 1-player Bernoulli stochastic bandit.

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 Ref: [Anandkumar et al, 2011]
- Recent: MEGA and Musical Chair. 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      $s^j(t+1) = \text{Non fixed}$  // 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      $s^j(t+1) = \text{Non fixed}$  // 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      $s^j(t+1) = \text{Non fixed}$  // 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

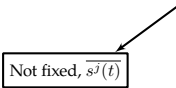
```
1 Let  $A^j(1) \sim \mathcal{U}(\{1, \dots, K\})$  and  $C^j(1) = \text{False}$  and  $s^j(1) = \text{Non fixed}$ 
2 for  $t = 1, \dots, T - 1$  do
3   if  $A^j(t) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t+1) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : \text{UCB}_k^j(t-1) \leq \text{UCB}_{A^j(t)}^j(t-1)\})$  // not
      empty
5      $s^j(t+1) = \text{Non fixed}$  // 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 $t = 0$

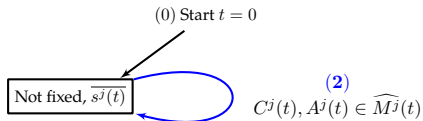
MCTopM algorithm illustrated, step by step

(0) Start $t = 0$

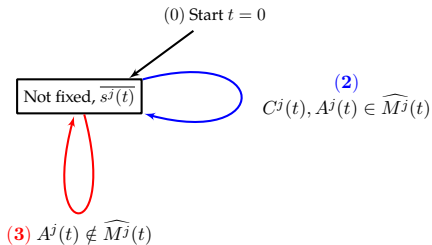


Not fixed, $\overline{s^j(t)}$

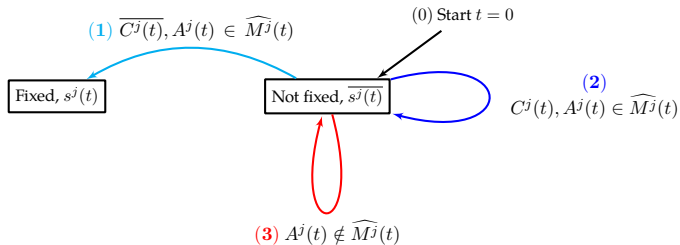
MCTopM algorithm illustrated, step by step



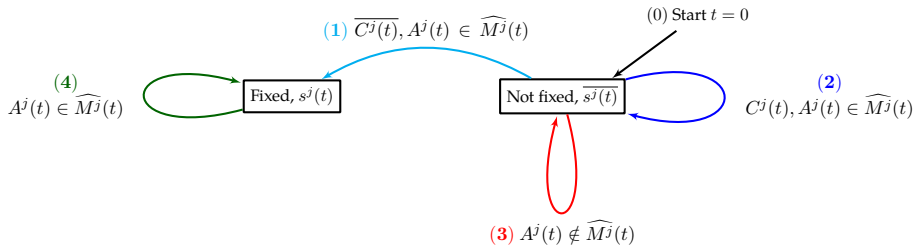
MCTopM algorithm illustrated, step by step



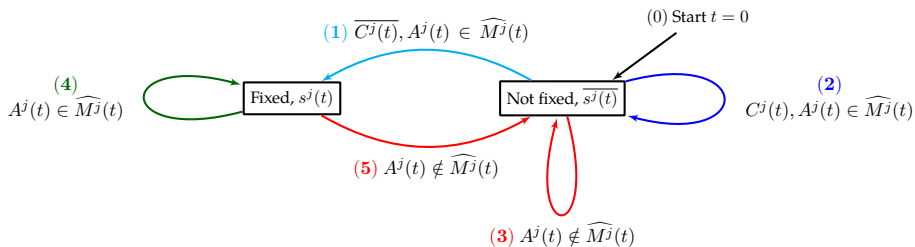
MCTopM algorithm illustrated, step by step



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:

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

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

Theorem 4

[Besson & Kaufmann, 2018]

If all M players use MCTopM with kl-UCB:

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

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* !

Experimental results

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

Illustration of the regret lower bound

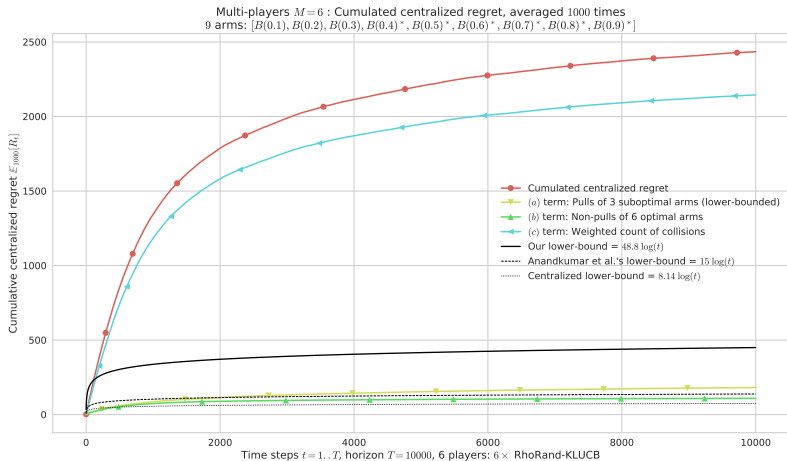


Figure 1: Any such lower bound is **very asymptotic**, usually not satisfied for small horizons. We can see the importance of the collisions!

Constant regret if $M = K$

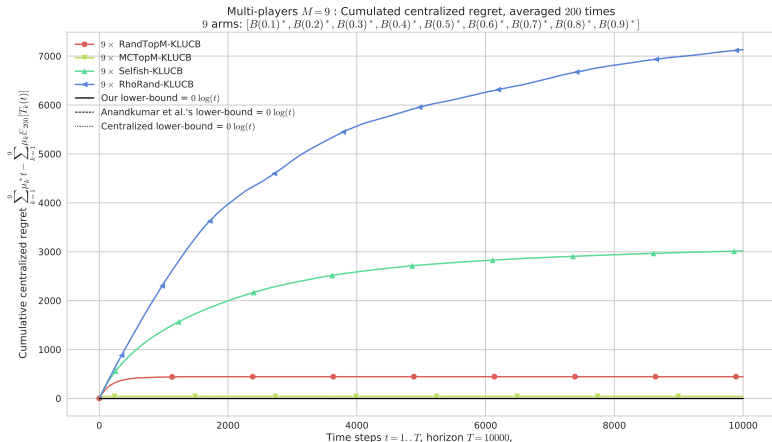


Figure 2: Regret, $M = 9$ players, $K = 9$ arms, horizon $T = 10000$, 200 repetitions. Only **RandTopM** and **MCTopM** achieve constant regret in this saturated case (proved).

Illustration of the regret of different algorithms

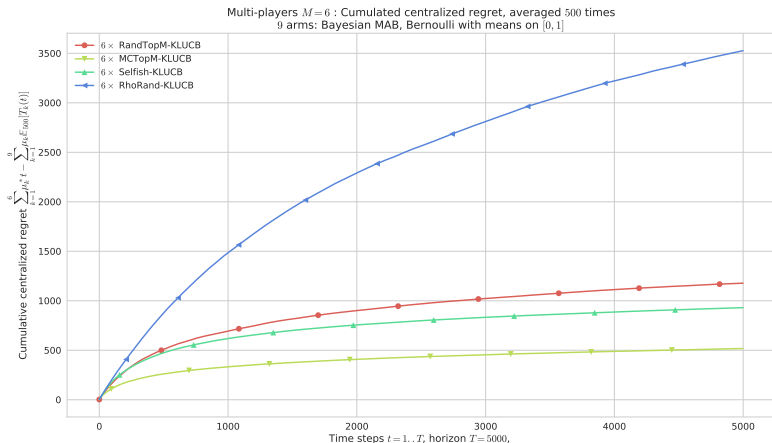


Figure 3: Regret, $M = 6$ players, $K = 9$ arms, horizon $T = 5000$, against 500 problems μ uniformly sampled in $[0, 1]^K$. Conclusion : **RhoRand** < **RandTopM** < **Selfish** < **MCTopM** in most cases.

Logarithmic number of collisions

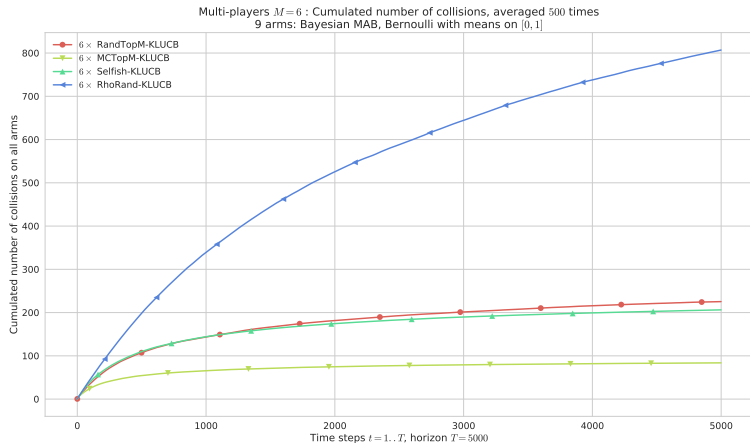


Figure 4: Cumulated number of collisions. Also $\text{RhoRand} < \text{RandTopM} < \text{Selfish} < \text{MCTopM}$.

Logarithmic number of arm switches

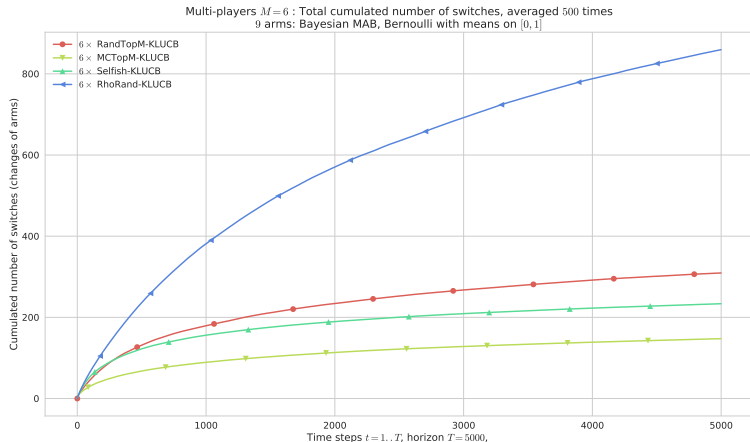


Figure 5: Cumulated number of arm switches. Again **RhoRand** < **RandTopM** < **Selfish** < **MCTopM**, but no guarantee for **RhoRand**. *Bonus* result: logarithmic arm switches for our algorithms!

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

- Remove hypothesis that objects know M ?
- Allow arrival/departure of objects?
- Non-stationarity of background traffic?

Future works

- Remove hypothesis that objects know M ?
- Allow arrival/departure of objects?
- Non-stationarity of background traffic?

- 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

- Remove hypothesis that objects know M ?
- Allow arrival/departure of objects?
- Non-stationarity of background traffic?

- 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!

- Maybe study **other emission models**?
- Implement and test this on **real-world radio devices**?
↔ Yes! Demo presented at the ICT 2018 conference!
(Saint-Malo, France)

Thanks!

Thanks! 😊

Any question ?

Appendix

- Proof of the regret upper bound,
- Illustration of the proof,
- An heuristic for the “IoT” case (no sensing): the Selfish algorithm,
- Success and failures case for Selfish.

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

Sketch of the proof

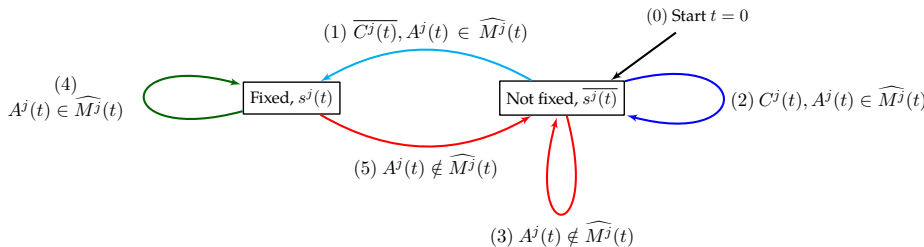
- 1 Bound the expected number of collisions by M times the number of collisions for non-fixed players,
- 2 Bound the expected number of transitions of type (3) and (5), by $\mathcal{O}(\log T)$ using the kl-UCB indexes and the forced choice of the algorithm:
$$\text{UCB}_k^j(t-1) \leq \text{UCB}_{k'}^j(t-1),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

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 $Y_{A^j(t)}(t)$.

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

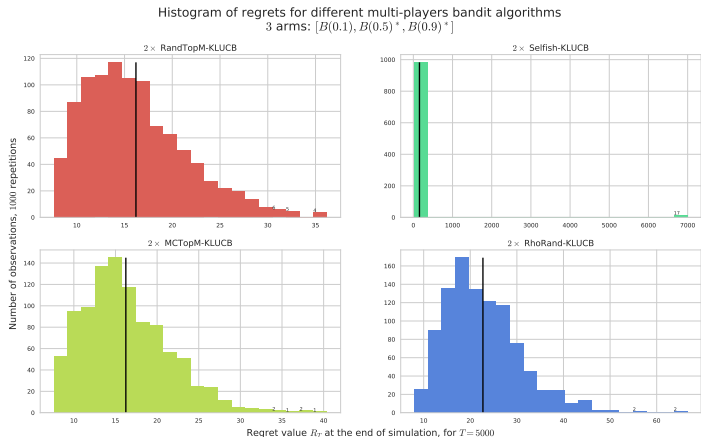


Figure 6: Regret for $M = 2$, $K = 3$, $T = 5000$, 1000 repetitions and $\mu = [0.1, 0.5, 0.9]$. Axis x is for regret (different scale for each), and **Selfish** have a small probability of failure (17/1000 cases of $R_T \gg \log T$). The regret for the three other algorithms is very small for this “easy” problem.