

“MULTI-PLAYERS BANDIT ALGORITHMS FOR INTERNET OF THINGS NETWORKS”

- ▶ By **Lilian Besson**
- ▶ **PhD defense at CentraleSupélec** (Rennes)
- ▶ Wednesday 20th of November, 2019
- ▶ Supervisors:
 - ▶ Prof. Christophe Moy at SCEE team, IETR & CentraleSupélec
 - ▶ Dr. Émilie Kaufmann at Sequel team, CNRS & Inria, in Lille



INTRODUCTION:

SPECTRUM ISSUES IN WIRELESS NETWORKS

Ref: Chapter 1 of my thesis.



Wireless networks

- ▶ All spectrum is allocated to different applications
- ▶ But all zones are not always used everywhere

🤔 **What if we could dynamically use the (most) empty channels?**

UNITED STATES FREQUENCY ALLOCATIONS

THE RADIO SPECTRUM



United States of North America, Department of Commerce, © 16



Target of this study

Wireless networks. . .

We focus on **Internet of Things** networks (IoT) in unlicensed bands.

↪ networks with **decentralized access**. . .

↪ **many wireless devices**  access a wireless network served from **one access point**


the base station is **not** affecting devices to radio resources. . .

Target of this study

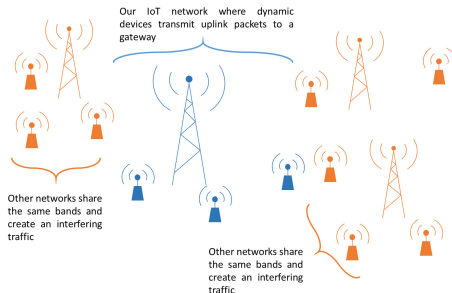
Wireless networks. . .

We focus on **Internet of Things** networks (IoT) in unlicensed bands.

↪ networks with **decentralized access**. . .

↪ **many wireless devices**  access a wireless network served from **one access point**

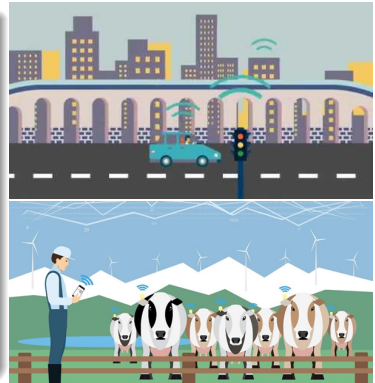
the base station is **not** affecting devices to radio resources. . .



The “Internet of Things”

Main constraints


- ▶ decentralized: **devices initiate transmission**
- ▶ can be in unlicensed radio bands
- ▶ **massive number of devices**
- ▶ long range
- ▶ ultra-low power devices
- ▶ **low duty cycle**
- ▶ **low data rate**





Images from <http://IBM.com/blogs/internet-of-things/what-is-the-iot> and <http://www.globalsign.com/en/blog/connected-cows-and-crop-control-to-drones-the-internet-of-things-is-rapidly-improving-agriculture/>





Main questions

- ▶ Can the IoT devices  optimize their access to the radio resources in a **simple**, **efficient**, **automatic** and **decentralized** way?
In a given location, and a given time, for a given radio standard...


Main questions

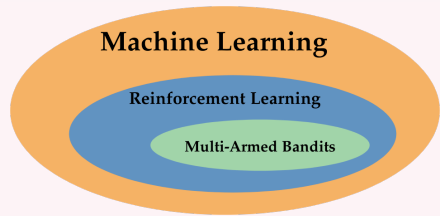
- ▶ Can the IoT devices  optimize their access to the radio resources in a **simple**, **efficient**, **automatic** and **decentralized** way?
In a given location, and a given time, for a given radio standard...
- ▶ Goal: increase the battery life of IoT devices 
- ▶ Fight the spectrum scarcity issue by using the spectrum more efficiently than a static or uniformly random allocation

Main questions

- ▶ Can the IoT devices  optimize their access to the radio resources in a **simple, efficient, automatic** and **decentralized** way?
In a given location, and a given time, for a given radio standard...
- ▶ Goal: increase the battery life of IoT devices 
- ▶ Fight the spectrum scarcity issue by using the spectrum more efficiently than a static or uniformly random allocation

Main solutions !

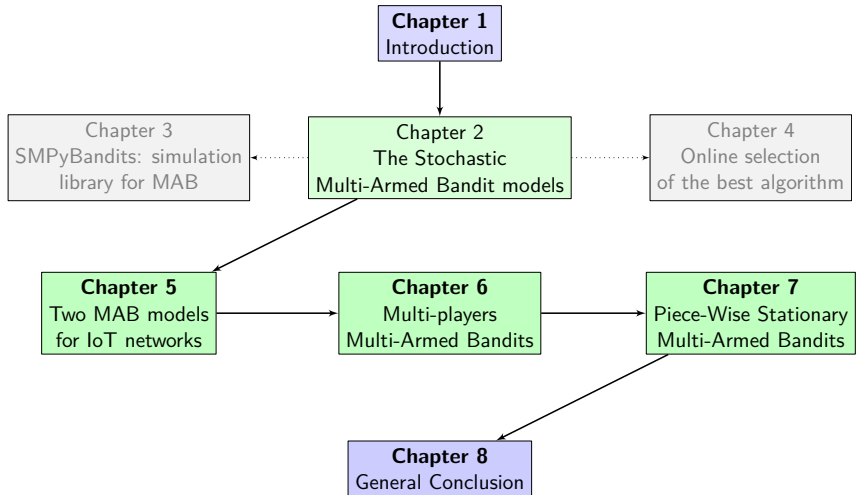
- ▶ Yes we can!
- ▶ By **letting the radio devices**  **become “intelligent”**
- ▶ With **MAB algorithms** !



OUTLINE OF THIS PRESENTATION



Contributions of my thesis highlighted today



Outline of this presentation

- ▶ **Introduction.** Spectrum issues in wireless networks
- ▶ **Part I.** Selfish MAB learning in a new model of IoT network
- ▶ **Part II.** *Two tractable problems* extending the classical bandit
 - ▶ multi-player bandits in stationary settings
 - ▶ single-player bandits in piece-wise stationary settings
- ▶ **Conclusion and perspectives**



PART I.

SELFISH MAB LEARNING IN IoT NETWORKS

Ref: Chapter 5 of my thesis, and [Bonnetoi, Besson et al, 17].



We want

We control a *lot* of IoT devices 📶

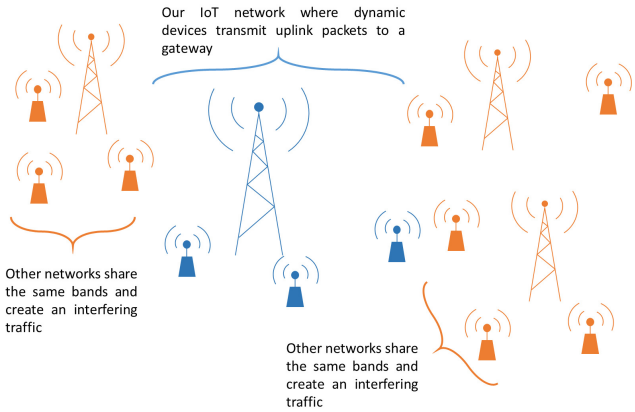
- ▶ We want to insert them in an already **crowded wireless network**
- ▶ Within a protocol **slotted in time and frequency**
- ▶ Each device 📶 / 📶 has a **low duty cycle** ex: few messages per day



We want

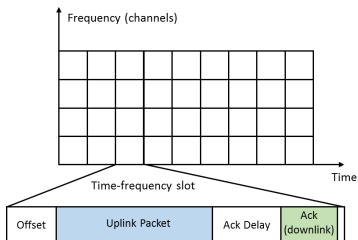
We control a *lot* of IoT devices 📶

- ▶ We want to insert them in an already **crowded wireless network**
- ▶ Within a protocol **slotted in time and frequency**
- ▶ Each device 📶 / 📶 has a **low duty cycle** ex: few messages per day



A new model for IoT networks

- ▶ Discrete time $t \in \mathbb{N}^*$ and K radio channels (e.g., 10) (known)

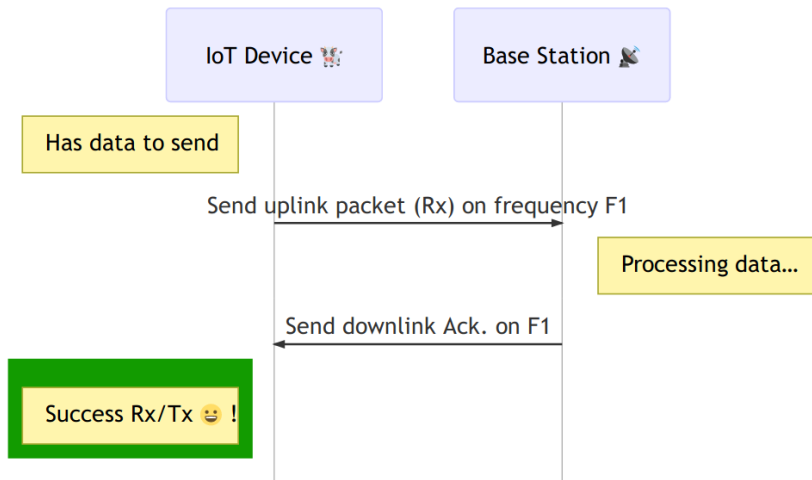


Chosen protocol: **uplink messages** ↗ followed by **acknowledgements** ↘

[Bonnefoi, Besson et al, 17], Sec.5.2

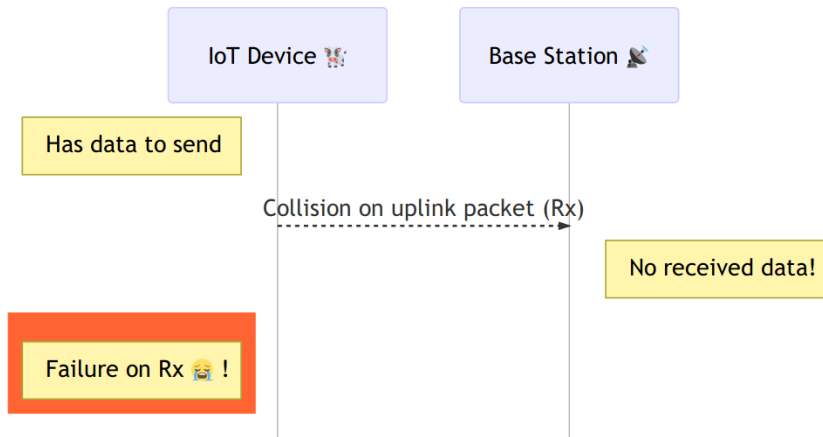
- ▶ D **dynamic** devices 📶 trying to access the network *independently*
- ▶ $S = S_1 + \dots + S_K$ **static** devices 📶 occupying the network:
 S_1, \dots, S_K in each channel $\{1, \dots, K\}$ (unknown)

Protocol: decentralized access with Ack. mode





1st case: Successful transmission if no collision on **uplink messages** ↗ !

Protocol: decentralized access with Ack. mode





2nd case: Failed transmission if collision on **uplink messages** ↗...



Emission model for IoT devices with *low duty cycle*

- ▶ Each device  /  has the same *low* emission probability: each step, each device sends a packet with probability p



Emission model for IoT devices with *low duty cycle*

- ▶ Each device  /  has the same *low* emission probability: each step, each device sends a packet with probability p



Background **stationary** ambient traffic

- ▶ Each static device  uses only one channel (S_k devices in channel k)
 - ▶ Their repartition is fixed in time
- ⇒ This surrounding traffic *is disturbing* the dynamic devices 


Emission model for IoT devices with *low duty cycle*

- ▶ Each device  /  has the same *low* emission probability: each step, each device sends a packet with probability p

Background **stationary** ambient traffic

- ▶ Each static device  uses only one channel (S_k devices in channel k)
 - ▶ Their repartition is fixed in time
- ⇒ This surrounding traffic *is disturbing* the dynamic devices 

Dynamic radio reconfiguration


- ▶ **Dynamic device**  **decide** the channel to use to send their packets
- ▶ They all have memory and computational capacity to implement small decision algorithms

Problem

Goal



- ▶ *minimize packet loss ratio* (max = number of received Ack)
- ▶ *in a finite-space discrete-time Decision Making Problem*

Baseline (naive solution)



Purely random (uniform) spectrum access for the D dynamic devices .

A possible solution

Embed a **decentralized Multi-Armed Bandit** algorithm, running **independently on each dynamic device** .

- If an oracle can affect D_k dynamic devices  to channel k , the **successful transmission probability** of the entire network is

$$\mathbb{P}(\text{success}|\text{sent}) = \sum_{k=1}^K \underbrace{(1-p)^{D_k-1}}_{D_k-1 \text{ others}} \times \underbrace{(1-p)^{S_k}}_{\text{No static device}} \times \underbrace{D_k/D}_{\text{Sent in channel } k}$$

- If an oracle can affect D_k dynamic devices  to channel k , the **successful transmission probability** of the entire network is

$$\mathbb{P}(\text{success}|\text{sent}) = \sum_{k=1}^K \underbrace{(1-p)^{D_k-1}}_{D_k-1 \text{ others}} \times \underbrace{(1-p)^{S_k}}_{\text{No static device}} \times \underbrace{D_k/D}_{\text{Sent in channel } k}$$

- The oracle has to solve this **optimization problem**:

$$\left\{ \begin{array}{l} \arg \max_{D_1, \dots, D_K} \quad \sum_{k=1}^K D_k (1-p)^{S_k + D_k - 1} \\ \text{such that} \quad \sum_{k=1}^K D_k = D \text{ and } D_k \geq 0, \quad \forall 1 \leq k \leq K. \end{array} \right.$$

Contribution: a (numerical) solver for this quasi-convex optimization problem, with *Lagrange multipliers*.

1) Oracle centralized strategy

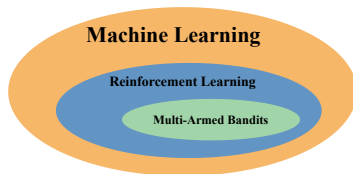
⇒ This *oracle* strategy has very good performance, as it maximizes the transmission rate of all the D dynamic devices 📶

But unrealistic

But **not achievable in practice!**

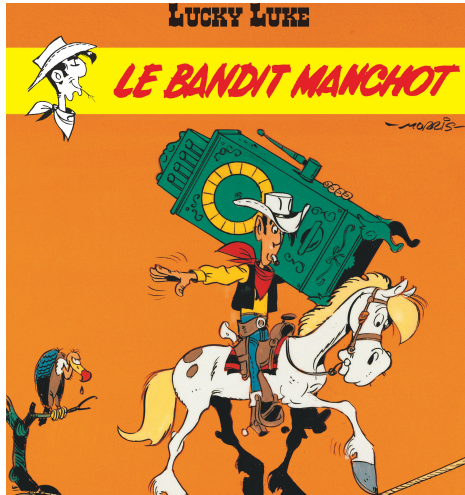
- ▶ because there is no centralized supervision!
- ▶ and (S_1, \dots, S_K) are unknown!

We propose a *realistic decentralized* approach, with bandits!



Hum, what is a (one-armed) *bandit*?

It's an old name for a casino machine 🎰 !




© Dargaud 1981, [Lucky Luke tome 18](#).



Stochastic *Multi-Armed Bandit* formulation

A player tries to collect **rewards** when playing a K -armed  bandit game.

At each round $t \in \{1, \dots, T\}$

- ▶ player chooses an *arm*  $A(t) \in \{1, \dots, K\}$
- ▶ the arm generates an i.i.d. **reward** $r_{A(t)}(t) \sim \nu_{A(t)}$
Ex: from a Bernoulli distribution $\nu_k = \mathcal{B}(\mu_k)$
- ▶ player observes the reward $r_{A(t)}(t)$

Stochastic *Multi-Armed Bandit* formulation

A player tries to collect **rewards** when playing a K -armed 🎰 bandit game.

At each round $t \in \{1, \dots, T\}$

- ▶ player chooses an *arm* 🎰 $A(t) \in \{1, \dots, K\}$
- ▶ the arm generates an i.i.d. **reward** $r_{A(t)}(t) \sim \nu_{A(t)}$
Ex: from a Bernoulli distribution $\nu_k = \mathcal{B}(\mu_k)$
- ▶ player observes the reward $r_{A(t)}(t)$

Goal (Reinforcement Learning)

Maximize the **sum reward** or **its expectation**


$$\max_A \sum_{t=1}^T r_{A(t)} \quad \text{or} \quad \max_A \mathbb{E} \left[\sum_{t=1}^T r_{A(t)} \right].$$

[Bubeck, 12], [Lattimore & Szepesvári, 19], [Slivkins, 19]




2) Pseudo *MAB* formulation of our IoT problem

A dynamic device  tries to collect **rewards** when transmitting:

- ▶ it transmits following a random Bernoulli process
(ie. probability p of transmitting at each round t)
- ▶ it chooses a channel $A(\tau) \in \{1, \dots, K\}$ (= arm )
 - ▶ if Ack (no collision) \implies **reward** $r_{A(\tau)} = 1$ (successful transm.!)
 - ▶ if collision (no Ack) \implies **reward** $r_{A(\tau)} = 0$ (failed transmission!)

2) Pseudo *MAB* formulation of our IoT problem

A dynamic device  tries to collect **rewards** when transmitting:

- ▶ it transmits following a random Bernoulli process
(ie. probability p of transmitting at each round t)
- ▶ it chooses a channel $A(\tau) \in \{1, \dots, K\}$ (= arm )
 - ▶ if Ack (no collision) \implies reward $r_{A(\tau)} = 1$ (successful transm.!)
▶ if collision (no Ack) \implies reward $r_{A(\tau)} = 0$ (failed transmission!)

Goal: Maximize transmission rate \equiv maximize **cumulated rewards**

It is not a *stochastic* Multi-Armed Bandit problem

It looks like a **MAB** but the **environment is not stochastic or stationary**

A dynamic device keeps τ number of sent packets

- 1 For the first K activations ($\tau = 1, \dots, K$), try each channel *once*.



A dynamic device keeps τ number of sent packets

- 1 For the first K activations ($\tau = 1, \dots, K$), try each channel *once*.
- 2 Then for the next steps t :
 - ▶ With probability p , the device is active ($\tau := \tau + 1$)

▶ Compute the index
$$\text{UCB}_k(\tau) := \underbrace{\frac{X_k(\tau)}{N_k(\tau)}}_{\text{Mean } \widehat{\mu}_k(\tau)} + \underbrace{\sqrt{\frac{\log(\tau)}{2N_k(\tau)}}}_{\text{Confidence Bonus}},$$

▶ Choose channel $A(\tau) = \arg \max_k \text{UCB}_k(\tau)$,

▶ Observe reward $r_{A(\tau)}(\tau)$ from arm $A(\tau)$

- ▶ Update $N_k(\tau + 1)$ nb selections of channel k
- ▶ Update $X_k(\tau)$ nb of successful transmissions

▶ Wait for next message... (mean waiting time $\simeq 1/p$)

- ④ For any dynamic device i , for any round t :
 - ▶ With probability p , the device is active ($\tau := \tau + 1$)
 - ▶ Play UCB algorithm. . . [Auer et al, 02]
 - ▶ Wait for next message. . . (mean waiting time $\simeq 1/p$)

Problem 1: multiple dynamic devices




- ▶ The collisions between dynamic devices i are **not stochastic!**

Problem 2: random activation times τ ?




- ▶ Devices transmits only with probability p at each time t (following its Bernoulli activation pattern)
- ▶ The times τ are **not** the global time indexes t (synchronized clock) !

⇒ These two problems make the model **hard to analyze** !

Experimental setting: simulation parameters

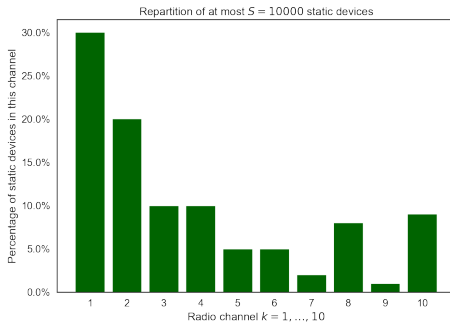
- ▶ $K = 10$ channels ,
- ▶ S  + D  = 10000 devices in total,

Experimental setting: simulation parameters

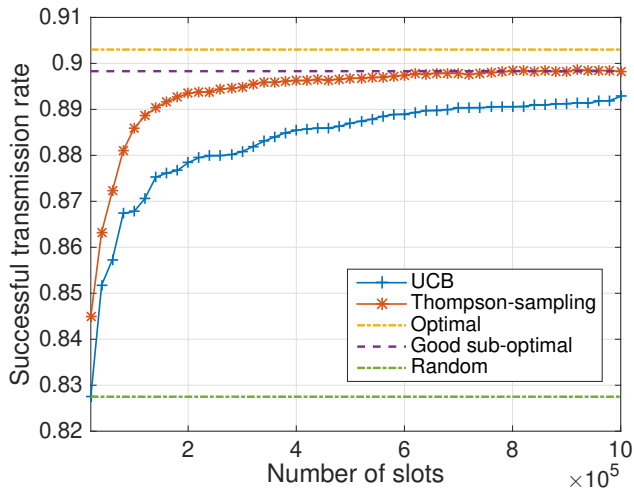
- ▶ $K = 10$ channels ,
- ▶ S  + D  = 10000 devices in total,
- ▶ $p = 10^{-3}$ probability of emission,
- ▶ Horizon $T = 10^5$ total time slots (avg. $\simeq 100$ messages / device),


Experimental setting: simulation parameters

- ▶ $K = 10$ channels 📶,
- ▶ S 📶 + D 📶 = 10000 devices in total,
- ▶ $p = 10^{-3}$ probability of emission,
- ▶ Horizon $T = 10^5$ total time slots (avg. $\simeq 100$ messages / device),
- ▶ We change the proportion of dynamic devices D 📶 / $(S$ 📶 + D 📶),
- ▶ For one example of repartition of (S_1, \dots, S_K) static devices 📶.



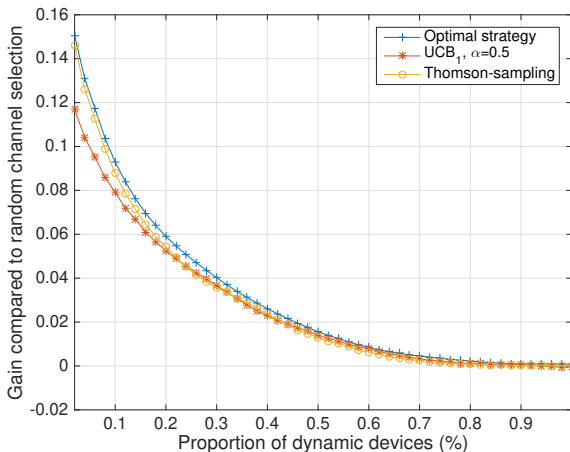
One result for 10% of dynamic devices



10% of dynamic devices . Gives 7% of gain.

[Bonnefoi, Besson et al, 17], Sec.5.2

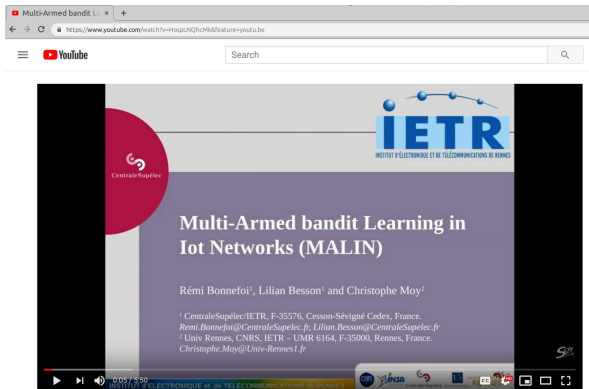
Growing proportion of dynamic devices $D/(S + D)$



- ▶ The MAB selfish learning is *almost optimal*, for any proportion of dynamic devices $\frac{D}{S+D}$, after a short learning time.
- ▶ In this example, it gives up-to 16% gain over the naive approach!



We developed a realistic demonstration using USRP boards and GNU Radio, as a proof-of-concept in a “toy” IoT network.



Multi-Armed bandit Learning in Iot Networks (MALIN) - Demo at ICT 2018

[Bonnefoi et al, ICT 18], [Besson et al, WCNC 19], Ch.5.3 and video published on [YouTu.be/HospLNQhcMk](https://www.youtube.com/watch?v=HospLNQhcMk)

Interfering traffic generator

Generates the interfering traffic created by neighboring networks that share the same unlicensed band.

Gateway

Receives and decodes the packets sent by the dynamic device.

If the packet is successfully decoded, it transmits an ACK in the channel used for the uplink communication.

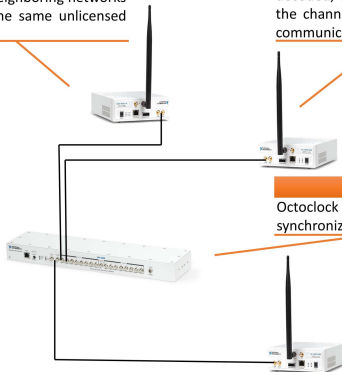
Octoclock

Octoclock that eases the synchronization between USRPs

Dynamic device

Sends uplink packets to the gateway.

Can choose a channel for each transmission.






The screenshot shows the GNU Radio GUI for an IoT demo, divided into several panels:

- IoT Traffic Generator (1):** Controls for generating traffic. Includes options for "Enable IoT traffic" and "Number of messages by second: 4".
- Intelligent Dynamic User (2):** Controls for channel selection. Includes "Active channels: #2, #4, #6, #8" and "Algorithm: UCB and symbols -1 ~ 1j".
- IoT Gateway (3):** Controls for listening and replying. Includes "Active channels: #2, #4, #6, #8" and "Speed: Packet of 1 second, 1 second to receive ack".
- Waterfall Displays (a, b, c):** Real-time frequency spectrum plots showing signal activity over time. (a) shows a single signal, (b) shows multiple signals, and (c) shows a dense signal structure.
- IoT Object (d):** Summary statistics including:
 - Number of Trials, total = 63
 - Successes, total = 38
 - UCB indexes, max = #8
 - Success Rates, mean = 60%




From practice to theory

It works very well empirically! But **random activation times** and **collisions due to multiple devices** make the model **hard to analyze**...

- ▶ Hyp 1: in avg. $p \times D$ dynamic devices  are using K channels 
 \implies so $p \leq \frac{K}{D}$ or $D \leq \frac{K}{p}$ gives best performance
- ▶ Hyp 2: we assumed a stationary background traffic  ...

From practice to theory




It works very well empirically! But **random activation times** and **collisions due to multiple devices** make the model **hard to analyze**...

- ▶ Hyp 1: in avg. $p \times D$ dynamic devices  are using K channels 
 \implies so $p \leq \frac{K}{D}$ or $D \leq \frac{K}{p}$ gives best performance
- ▶ Hyp 2: we assumed a stationary background traffic  ...

Goal: obtain theoretical result for our proposed model of IoT networks, and guarantees about the observed behavior of *Selfish MAB learning*.



From practice to theory

It works very well empirically! But **random activation times** and **collisions due to multiple devices** make the model **hard to analyze**...

- ▶ Hyp 1: in avg. $p \times D$ dynamic devices  are using K channels 
 \implies so $p \leq \frac{K}{D}$ or $D \leq \frac{K}{p}$ gives best performance
- ▶ Hyp 2: we assumed a stationary background traffic  ...

Goal: obtain theoretical result for our proposed model of IoT networks, and guarantees about the observed behavior of *Selfish MAB learning*.

We can study theoretically two more specific models

- ▶ Model 1: **multi-player bandits**: devices  are always activated
ie. $p = 1$ in their random activation process $\implies D = M \leq \frac{K}{p} = K$
- ▶ Model 2: **non-stationary bandits** (for one device )

PART II.

THEORETICAL ANALYSIS OF TWO RELAXED MODELS

Ref: Chapters 6 and 7 of my thesis
and [Besson & Kaufmann, 18] and [Besson et al, 19].



Theoretical analysis of two relaxed models


Multi-player bandits

Ref: Chapter 6 of my thesis, and [Besson & Kaufmann, 18].


Piece-wise stationary bandits



Multi-players bandits: setup


$M \geq 2$ players  playing *the same* K -armed bandit ($2 \leq M \leq K$)
they are all activated at each time step, ie. $p = 1$

At round $t \in \{1, \dots, T\}$:


- ▶ player m selects arm A_t^m  ; then *this arm generates* $s_{A_t^m, t} \in \{0, 1\}$
- ▶ and the reward is computed as

$$r_{m,t} = \begin{cases} s_{A_t^m, t} & \text{if no other player chose the same arm} \\ 0 & \text{else (= COLLISION)} \end{cases}$$

Multi-players bandits: setup

$M \geq 2$ players  playing the same K -armed bandit $(2 \leq M \leq K)$
they are all activated at each time step, ie. $p = 1$

At round $t \in \{1, \dots, T\}$:

- ▶ player m selects arm A_t^m ; then *this arm generates* $s_{A_t^m, t} \in \{0, 1\}$
- ▶ and the reward is computed as

$$r_{m,t} = \begin{cases} s_{A_t^m, t} & \text{if no other player chose the same arm} \\ 0 & \text{else (= COLLISION)} \end{cases}$$

Goal

- ▶ maximize centralized (sum) rewards $\sum_{m=1}^M \sum_{t=1}^T r_{m,t}$
- ▶ ... without (explicit) communication between players
- ▶ trade-off: exploration / exploitation / and collisions !



Different observation models: players observe $s_{A_t^m, t}$ and/or $r_{m, t}$

1: “Listen before talk”

[Liu & Zhao, 10], [Jouini et al. 10], [Anandkumar et al. 11]

- ▶ Good model for **Opportunistic Spectrum Access** (OSA)
- ▶ First do sensing, attempt of transmission if no Primary User (PU), possible collisions with other Secondary Users (SU).
- ▶ *Feedback model:*
 - ▶ observe first $s_{A_t^m, t}$,
 - ▶ if $s_{A_t^m, t} = 1$, transmit and then observe the joint $r_{m, t}$,
 - ▶ else don't transmit and don't observe a reward.

2: “Talk and maybe collide”

[Besson & Kaufmann, 18]

- ▶ Good model for **Internet of Things** (IoT)
- ▶ Do not do any sensing, just transmit, and wait for an acknowledgment before any next message.
- ▶ *Feedback model:*
 - ▶ observe only the joint information $r_{m,t}$,
 - ▶ no collision if $r_{m,t} \neq 0$,
 - ▶ but cannot distinguish between collision or zero reward if $r_{m,t} = 0$.

2: “Talk and maybe collide”

[Besson & Kaufmann, 18]

- ▶ Good model for **Internet of Things** (IoT)
- ▶ Do not do any sensing, just transmit, and wait for an acknowledgment before any next message.
- ▶ *Feedback model:*
 - ▶ observe only the joint information $r_{m,t}$,
 - ▶ no collision if $r_{m,t} \neq 0$,
 - ▶ but cannot distinguish between collision or zero reward if $r_{m,t} = 0$.

3: “Observe collision then talk?”

[Besson & Kaufmann, 18], [Boursier et al, 19]

- ▶ A third “hybrid” model studied by several recent papers, following our work
- ▶ *Feedback model:*
 - ▶ first check if collision,
 - ▶ then if not collision, receive joint reward $r_{m,t}$.



Regret for multi-player bandits (M players on K arms)

Hypothesis: arms sorted by decreasing mean: $\mu_1 \geq \mu_2 \geq \dots \geq \mu_K$

$$R_{\mu}(\mathcal{A}, T) := \underbrace{\left(\sum_{k=1}^M \mu_k \right) T}_{\text{oracle total reward}} - \mathbb{E}_{\mu}^{\mathcal{A}} \left[\sum_{t=1}^T \sum_{m=1}^M r_{m,t} \right]$$

Regret decomposition

[Besson & Kaufmann, 18]

$$R_{\mu}(\mathcal{A}, T) = \sum_{k=M+1}^K (\mu_M - \mu_k) \mathbb{E}[N_k(T)] \\ + \sum_{k=1}^M (\mu_k - \mu_M) (T - \mathbb{E}[N_k(T)]) + \sum_{k=1}^K \mu_k \mathbb{E}[C_k(T)].$$

- ▶ $N_k(T)$ total number of selections of arm k
- ▶ $C_k(T)$ total number of collisions experienced on arm k



Regret decomposition

[Besson & Kaufmann, 18]

$$R_{\mu}(\mathcal{A}, T) \leq \text{cst} \sum_{k=M+1}^K \mathbb{E}[N_k(T)] + \text{cst}' \sum_{k=1}^M \mathbb{E}[C_k(T)].$$

A good algorithm has to control both

- ▶ the number of **selections of sub-optimal arms**
↔ with a good classical bandit policy: like kl-UCB
- ▶ the number of **collisions** on optimal arms
↔ with a good orthogonalization procedure



The MC-Top- M algorithm (for the OSA case)

At round t , player m uses his past sensing information to:

- ▶ compute an Upper Confidence Bound for each mean μ_k , $\text{UCB}_k^m(t)$
- ▶ use the UCBs to **estimate the M best arms**

$$\hat{M}^m(t) := \{\text{arms with } M \text{ largest } \text{UCB}_k^m(t)\}$$

Two simple ideas: inspired by Musical Chair [Rosenski et al. 16]

- ▶ always pick an arm estimated as “good” $A^m(t) \in \hat{M}^m(t-1)$
- ▶ try not to switch arm too often

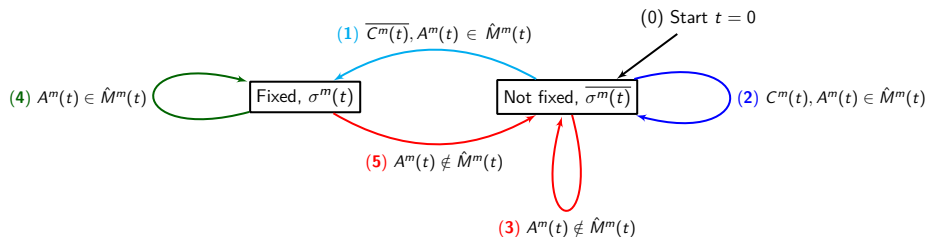
$$\sigma^m(t) := \{\text{player } m \text{ is “fixed” at the end of round } t\}$$

Other UCB-based algorithms: TDFS [Lui and Zhao, 10],

Rho-Rand [Anandkumar et al., 11], Selfish [Bonnefoi, Besson et al., 17]



The MC-Top- M algorithm (for the OSA case)



Sketch of the proof to bound number of collisions

- ▶ any sequence of transitions (2) has constant length
 - ▶ $\mathcal{O}(\log T)$ number of transitions (3) and (5), by kl-UCB
- \Rightarrow player m is fixed, for almost all rounds ($\mathcal{O}(T - \log T)$ times)
- ▶ nb of collisions $\leq M \times$ nb of collisions of non fixed players
- \Rightarrow nb of collisions = $\mathcal{O}(\log T)$ & $\mathcal{O}(\log(T))$ sub-optimal selections (4)

Theoretical results for MC-Top- M

MC-Top- M with kl-based confidence intervals [Cappé et al. 13]

$$\text{UCB}_k^m(t) = \max \{q : N_k^m(t) \text{kl}(\hat{\mu}_k^m(t), q) \leq \ln(t)\},$$

where $\text{kl}(x, y) = \text{KL}(\mathcal{B}(x), \mathcal{B}(y)) = x \ln\left(\frac{x}{y}\right) + (1-x) \ln\left(\frac{1-x}{1-y}\right)$.

Control of the sub-optimal selections (state-of-the-art)

For all sub-optimal arms $k \in \{M+1, \dots, K\}$,

$$\mathbb{E}[N_k^m(T)] \leq \frac{\ln(T)}{\text{kl}(\mu_k, \mu_M)} + C_\mu \sqrt{\ln(T)}.$$

Control of the collisions (new result)

$$\mathbb{E}\left[\sum_{k=1}^K C_k(T)\right] \leq M^2 \left(\sum_{a,b:\mu_a < \mu_b} \frac{2M+1}{\text{kl}(\mu_a, \mu_b)} \right) \ln(T) + \mathcal{O}(\ln T).$$

Theoretical results for MC-Top-M

MC-Top-M with kl-based confidence intervals [Cappé et al. 13]

$$\text{UCB}_k^m(t) = \max \{q : N_k^m(t) \text{kl}(\hat{\mu}_k^m(t), q) \leq \ln(t)\},$$

where $\text{kl}(x, y) = \text{KL}(\mathcal{B}(x), \mathcal{B}(y)) = x \ln\left(\frac{x}{y}\right) + (1-x) \ln\left(\frac{1-x}{1-y}\right)$.

Control of the sub-optimal selections (state-of-the-art)

For all sub-optimal arms $k \in \{M+1, \dots, K\}$,

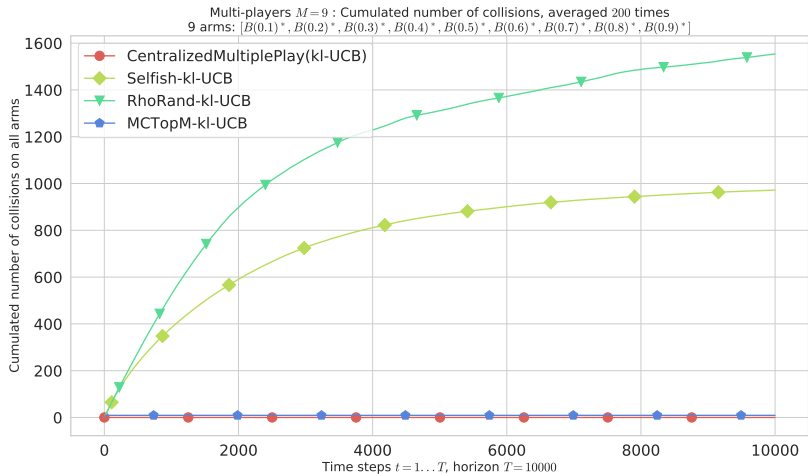
$$\mathbb{E}[N_k^m(T)] \leq \frac{\ln(T)}{\text{kl}(\mu_k, \mu_M)} + C_\mu \sqrt{\ln(T)}.$$

logarithmic regret $\implies R_\mu(\mathcal{A}, T) = \mathcal{O}((MC_{M,\mu} + M^2 C_6) \log(T))$

Control of the collisions (new result)

$$\mathbb{E} \left[\sum_{k=1}^K C_k(T) \right] \leq M^2 \left(\sum_{a,b:\mu_a < \mu_b} \frac{2M+1}{\text{kl}(\mu_a, \mu_b)} \right) \ln(T) + \mathcal{O}(\ln T).$$

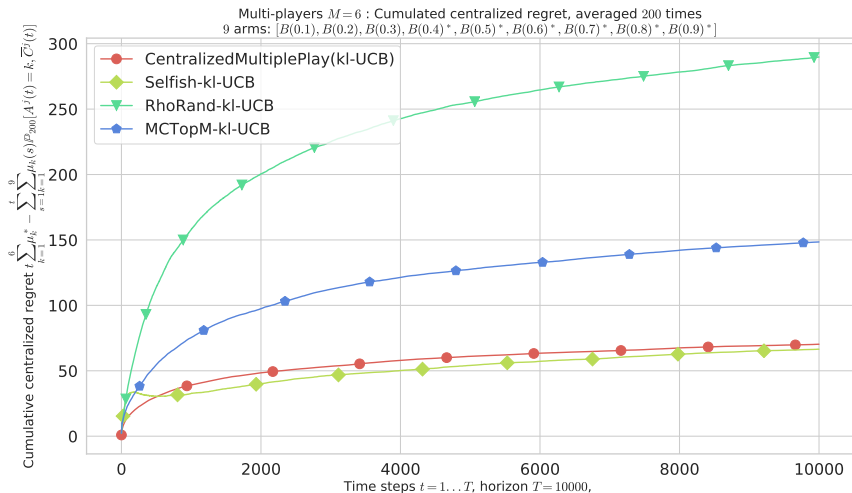
Results on a multi-player MAB problem (1/2)



For $M = K$, our strategy MC-Top- M (\diamond) achieves **constant** nb of collisions!
 \Rightarrow Our new orthogonalization procedure is very efficient!



Results on a multi-player MAB problem (2/2)



For $M = 6$ devices, our strategy MC-Top- M (\diamond) largely outperforms ρ^{rand} and other previous state-of-the-art policies (not included).

State-of-the-art multi-player algorithms

Algorithm	Ref.	Regret bound	Performance <small>* is worst</small>	Speed <small>🚀 is worst</small>	Parameters
Centralized multi-play kl-UCB	[1]	$C_{M,\mu} \log(T)$	★★★★★	🚀🚀	just M but in another model
ρ^{rand} UCB	[2]	$M^3 C_2 \log(T)$	★★	🚀🚀	just M
MEGA	[3]	$C_3 T^{3/4}$	★	🚀🚀	4 params, <i>impossible to tune</i>
Musical Chair	[4]	$\binom{2M}{M} C_4 \log(T)$	★★	🚀🚀	1 parameter T_0 <i>hard to tune</i>
Selfish UCB	[5]	T in some case	★ / ★★★★★	🚀🚀🚀	none!
MCTopM klUCB	[6]	$(M C_{M,\mu} + M^2 C_6) \log(T)$	★★★★	🚀🚀	just M
SIC-MMAB	[7]	$(C_{M,\mu} + MK) \log(T)$	★★★★	🚀	none! but in another model
DPE	[8]	$C_{M,\mu} \log(T)$??	🚀	none! but in another model

Optimal **regret bound** is multiple-play bound $\mathcal{R}(\mathcal{A}, T) \leq C_{M,\mu} \log(T) + o(\log(T))$, with

$$C_{M,\mu} = \sum_{k:\mu_k < \mu_M^*} \sum_{j=1}^M \frac{\mu_M^*}{\text{kl}(\mu_k, \mu_j^*)}, \text{ and } C_i \gg C_{M,\mu} \text{ are much larger constants.}$$

Papers: [1] Anantharam et al, 87 [2] Anandkumar et al, 11 [3] Avner et al, 15 [4] Rosenski et al, 15 [5] Bonnefoi et al 17 [6] Besson & Kaufmann, 18 [7] Boursier et al, 19 [8] Proutière et al, 19



Theoretical analysis of two relaxed models

Multi-player bandits

Piece-wise stationary bandits

Ref: Chapter 7 of my thesis, and [Besson et al, 19].



Piece-wise stationary bandits

Stationary MAB problems

Arm k  samples rewards from **the same distribution** for any round

$$\forall t, r_k(t) \stackrel{\text{iid}}{\sim} \nu_k = \mathcal{B}(\mu_k).$$



Piece-wise stationary bandits

Stationary MAB problems

Arm k  samples rewards from **the same distribution** for any round

$$\forall t, r_k(t) \stackrel{\text{iid}}{\sim} \nu_k = \mathcal{B}(\mu_k).$$

Non stationary MAB problems?

(possibly) **different distributions** for any round !

$$\forall t, r_k(t) \stackrel{\text{iid}}{\sim} \nu_k(t) = \mathcal{B}(\mu_k(t)).$$

⇒ **harder problem!** And impossible with no extra hypothesis



Piece-wise stationary bandits

Stationary MAB problems

Arm k  samples rewards from **the same distribution** for any round

$$\forall t, r_k(t) \stackrel{\text{iid}}{\sim} \nu_k = \mathcal{B}(\mu_k).$$

Non stationary MAB problems?

(possibly) **different distributions** for any round !

$$\forall t, r_k(t) \stackrel{\text{iid}}{\sim} \nu_k(t) = \mathcal{B}(\mu_k(t)).$$


\Rightarrow **harder problem!** And impossible with no extra hypothesis

Piece-wise stationary problems!

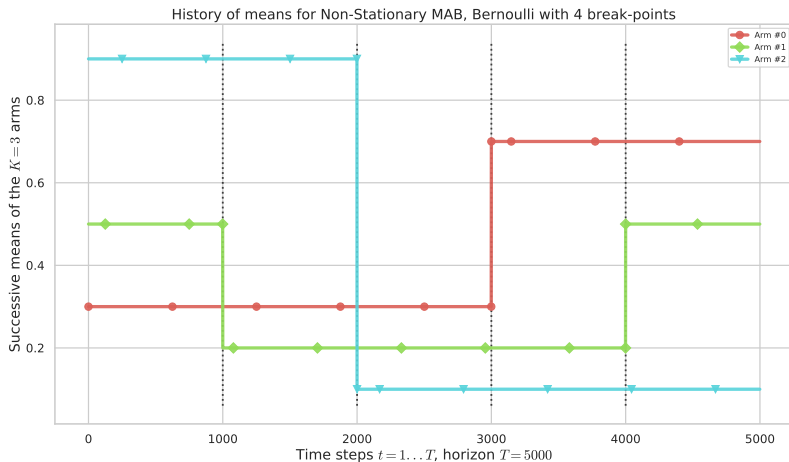
The literature usually focuses on the easier case, when there are at most $\Upsilon_T = o(\sqrt{T})$ intervals, on which the means are all stationary.



Example of a piece-wise stationary MAB problem

We plot the means $\mu_1(t)$, $\mu_2(t)$, $\mu_3(t)$ of $K = 3$ arms .

There are $\Upsilon_T = 4$ break-points and 5 sequences in $\{1, \dots, T = 5000\}$



Regret for piece-wise stationary bandits

The “oracle” plays the (**unknown**) best arm $k^*(t) = \operatorname{argmax} \mu_k(t)$
(which changes between the $\Upsilon_T \geq 1$ stationary sequences)

$$\begin{aligned}\mathcal{R}(\mathcal{A}, T) &= \mathbb{E} \left[\sum_{t=1}^T r_{k^*(t)}(t) \right] - \sum_{t=1}^T \mathbb{E} [r(t)] \\ &= \underbrace{\left(\sum_{t=1}^T \max_k \mu_k(t) \right)}_{\text{oracle total reward}} - \sum_{t=1}^T \mathbb{E} [r(t)].\end{aligned}$$



Regret for piece-wise stationary bandits

The “oracle” plays the (**unknown**) best arm $k^*(t) = \operatorname{argmax} \mu_k(t)$ (which changes between the $\Upsilon_T \geq 1$ stationary sequences)

$$\begin{aligned}\mathcal{R}(\mathcal{A}, T) &= \mathbb{E} \left[\sum_{t=1}^T r_{k^*(t)}(t) \right] - \sum_{t=1}^T \mathbb{E} [r(t)] \\ &= \underbrace{\left(\sum_{t=1}^T \max_k \mu_k(t) \right)}_{\text{oracle total reward}} - \sum_{t=1}^T \mathbb{E} [r(t)].\end{aligned}$$

Typical regimes for piece-wise stationary bandits

- ▶ The (minimax) *worst-case lower-bound* is $\mathcal{R}(\mathcal{A}, T) \geq \Omega(\sqrt{KT\Upsilon_T})$
- ▶ State-of-the-art algorithms \mathcal{A} obtain $\mathcal{R}(\mathcal{A}, T) \leq \mathcal{O}(K\sqrt{T\Upsilon_T \log(T)})$



Our new algorithm: **kl-UCB** index + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

- ▶ A classical bandit index policy: **kl-UCB**
which gets *restarted* after **a change-point is detected**



Our new algorithm: **kl-UCB** index + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

- ▶ A classical bandit index policy: **kl-UCB** which gets *restarted* after a **change-point is detected**
- ▶ A **change-point detection algorithm**: the **Generalized Likelihood Ratio Test** for sub-Bernoulli observations (BGLR), we can bound



Our new algorithm: **kl-UCB** + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

- ▶ A classical bandit index policy: **kl-UCB** which gets *restarted* after a **change-point is detected**
- ▶ A **change-point detection algorithm**: the **Generalized Likelihood Ratio Test** for sub-Bernoulli observations (BGLR), we can bound
 - ▶ its false alarm probability (if enough samples between two restarts)



Our new algorithm: **kl-UCB** index + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

- ▶ A classical bandit index policy: **kl-UCB** which gets *restarted* after a **change-point is detected**
- ▶ A **change-point detection algorithm**: the **Generalized Likelihood Ratio Test** for sub-Bernoulli observations (BGLR), we can bound
 - ▶ its false alarm probability (if enough samples between two restarts)
 - ▶ its detection delay (for “easy enough” problems)



Our new algorithm: **kl-UCB** + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

- ▶ A classical bandit index policy: **kl-UCB** which gets *restarted* after a **change-point is detected**
- ▶ A **change-point detection algorithm**: the **Generalized Likelihood Ratio Test** for sub-Bernoulli observations (BGLR), we can bound
 - ▶ its false alarm probability (if enough samples between two restarts)
 - ▶ its detection delay (for “easy enough” problems)
- ▶ Forced exploration of parameter $\alpha \in (0, 1)$ (tuned with Υ_T)



Our new algorithm: **kl-UCB** index + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

- ▶ A classical bandit index policy: **kl-UCB** which gets *restarted* after a **change-point is detected**
- ▶ A **change-point detection algorithm**: the **Generalized Likelihood Ratio Test** for sub-Bernoulli observations (BGLR), we can bound
 - ▶ its false alarm probability (if enough samples between two restarts)
 - ▶ its detection delay (for “easy enough” problems)
- ▶ Forced exploration of parameter $\alpha \in (0, 1)$ (tuned with Υ_T)



Our new algorithm: **kl-UCB** index + **BGLR** detector

Three components of our algorithm

[Besson et al, 19]

Our algorithm is inspired by CUSUM-UCB [Liu et al, 18] and M-UCB [Cao et al, 19], and new analysis of the GLR test [Maillard, 19]

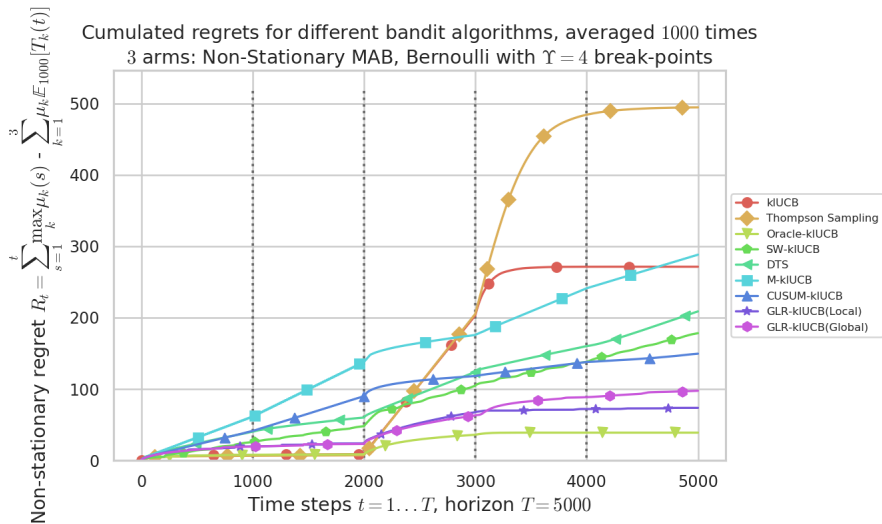
- ▶ A classical bandit index policy: **kl-UCB** which gets *restarted* after a **change-point is detected**
- ▶ A **change-point detection algorithm**: the **Generalized Likelihood Ratio Test** for sub-Bernoulli observations (BGLR), we can bound
 - ▶ its false alarm probability (if enough samples between two restarts)
 - ▶ its detection delay (for “easy enough” problems)
- ▶ Forced exploration of parameter $\alpha \in (0, 1)$ (tuned with Υ_T)

Regret bound (if T and Υ_T are both known)

Our algorithm obtains $\mathcal{R}(\mathcal{A}, T) \leq \mathcal{O}\left(\frac{K}{\Delta_{\text{change}}^2} \sqrt{T \Upsilon_T \log(T)}\right)$



Results on a piece-wise stationary MAB problem



↙ **kl-UCB + BGLR (*)** achieves the best performance (among non-oracle)!



State-of-the-art piece-wise stationary algorithms

Algorithm	Ref.	Regret bound	Performance <small>* is worst</small>	Speed <small>🐢 is worst</small>	Parameters
Naive UCB	[1]	T in worst case	★	🐢🐢🐢🐢🐢	none!
Oracle-Restart UCB	[1]	$\mathbf{C}\Upsilon_T \log(T)$	★★★★★	🐢🐢🐢🐢🐢	the break-points (unrealistic oracle!)
Discounted UCB	[2]	$\mathbf{C}_2 \sqrt{T \Upsilon_T \log(T)}$	★	🐢🐢🐢🐢	T and Υ_T
Sliding-Window UCB	[2]	$\mathbf{C}'_2 \sqrt{T \Upsilon_T \log(T)}$	★	🐢🐢🐢🐢	T and Υ_T
Exp3.S	[3]	$\mathbf{C} \sqrt{T \Upsilon_T \log(T)}$	★	🐢🐢🐢🐢🐢	Υ_T
Discounted TS	[4]	not yet proven	★★	🐢🐢🐢🐢	how to tune γ ?
CUSUM-UCB	[5]	$\mathbf{C}_5 \sqrt{T \Upsilon_T \log(\frac{T}{\Upsilon_T})}$	★★★	🐢🐢	T , Υ_T and δ_{\min}
M-UCB	[6]	$\mathbf{C}_6 \sqrt{T \Upsilon_T \log(T)}$	★★	🐢🐢🐢	T , Υ_T and δ_{\min}
BGLR + kl-UCB	[7]	$\mathbf{C} \sqrt{T \Upsilon_T \log(T)}$	★★★★★	🐢🐢	T and Υ_T
AdSwitch	[8]	$\mathbf{C}_8 \sqrt{T \Upsilon_T \log(T)}$	★★	🐢	just T
Ada-ILTCB ⁺	[9]	$\mathbf{C}_9 \sqrt{T \Upsilon_T \log(T)}$??	🐢	just T

Optimal minimax **regret bound** is $\mathcal{R}(\mathcal{A}, T) = \mathcal{O}(\sqrt{KT\Upsilon_T})$, and $\mathbf{C} = \mathbf{C}_{\Upsilon_T, \mu} = \mathcal{O}(\frac{K}{\Delta_{\text{change}}^2})$.

$\mathbf{C}_i \gg \mathbf{C}_{\Upsilon_T, \mu}$ are much larger constants, and $\delta_{\min} < \Delta_{\text{change}}$ lower-bounds the problem difficulty.

Papers: [1] Auer et al. 02 [2] Garivier et al. 09 [3] Auer et al. 02 [5] Raj et al. 17

[5] Liu et al. 18 [6] Cao et al. 19 [7] **Besson et al. 19** [8] Auer et al. 19 [9] Chen et al. 19



SUMMARY



Part I:

Part II:



Part I:

- ▶ A simple model of IoT network, where autonomous IoT devices can embed decentralized learning (“selfish MAB learning”),
- ▶ numerical simulations proving the quality of our solution,
- ▶ a realistic implementation on radio hardware.

Part II:



Part I:

- ▶ A simple model of IoT network, where autonomous IoT devices can embed decentralized learning (“**selfish MAB learning**”),
- ▶ numerical simulations proving the quality of our solution,
- ▶ a realistic implementation on radio hardware.

Part II:

- ▶ New algorithms and regret bounds, in two simplified models:
 - ▶ for **multi-player bandits**, with $M \leq K$ players,
 - ▶ for **piece-wise stationary bandits**, with $\Upsilon_T = o(T)$ break-points,
- ▶ our proposed algorithms achieve state-of-the-art performance
 - ▶ on both numerical,
 - ▶ and theoretical results.



- ▶ Unify the *multi-player* and *non-stationary* bandit models
 - ↪ in progress: already one paper from last year (arXiv:1812.05165), we can probably do a better job with our tools!

- ▶ Unify the *multi-player* and *non-stationary* bandit models
↪ in progress: already one paper from last year (arXiv:1812.05165), we can probably do a better job with our tools!
- ▶ More validation of our contributions in real-world IoT environments
↪ started in summer 2019 with an intern working with Christophe Moy

- ▶ Unify the *multi-player* and *non-stationary* bandit models
 - ↪ in progress: already one paper from last year (arXiv:1812.05165), we can probably do a better job with our tools!
- ▶ More validation of our contributions in real-world IoT environments
 - ↪ started in summer 2019 with an intern working with Christophe Moy
- ▶ Study the “Graal” goal:

- ▶ Unify the *multi-player* and *non-stationary* bandit models
 - ↪ in progress: already one paper from last year (arXiv:1812.05165), we can probably do a better job with our tools!
- ▶ More validation of our contributions in real-world IoT environments
 - ↪ started in summer 2019 with an intern working with Christophe Moy
- ▶ Study the “Graal” goal:
 - ▶ propose a more realistic model for IoT networks (exogenous activation, non stationary traffic, etc)

- ▶ Unify the *multi-player* and *non-stationary* bandit models
 - ↪ in progress: already one paper from last year (arXiv:1812.05165), we can probably do a better job with our tools!
- ▶ More validation of our contributions in real-world IoT environments
 - ↪ started in summer 2019 with an intern working with Christophe Moy
- ▶ Study the “Graal” goal:
 - ▶ propose a more realistic model for IoT networks (exogenous activation, non stationary traffic, etc)
 - ▶ propose an efficient decentralized low-cost algorithm

- ▶ Unify the *multi-player* and *non-stationary* bandit models
 - ↪ in progress: already one paper from last year (arXiv:1812.05165), we can probably do a better job with our tools!
- ▶ More validation of our contributions in real-world IoT environments
 - ↪ started in summer 2019 with an intern working with Christophe Moy
- ▶ Study the “Graal” goal:
 - ▶ propose a more realistic model for IoT networks (exogenous activation, non stationary traffic, etc)
 - ▶ propose an efficient decentralized low-cost algorithm
 - ▶ that works empirically *and* has strong theoretical guarantees!

- ▶ Unify the *multi-player* and *non-stationary* bandit models
 - ↪ in progress: already one paper from last year ([arXiv:1812.05165](https://arxiv.org/abs/1812.05165)), we can probably do a better job with our tools!
- ▶ More validation of our contributions in real-world IoT environments
 - ↪ started in summer 2019 with an intern working with Christophe Moy
- ▶ Study the “Gaal” goal:
 - ▶ propose a more realistic model for IoT networks (exogenous activation, non stationary traffic, etc)
 - ▶ propose an efficient decentralized low-cost algorithm
 - ▶ that works empirically *and* has strong theoretical guarantees!
- ▶ Extend my Python library [SMPyBandits](#) to cover many other bandit models (cascading, delay feedback, combinatorial, contextual etc)
 - ↪ it is already online, free and open-source on [GitHub.com/SMPyBandits](https://github.com/SMPyBandits)

8 International conferences with proceedings:

- ▶ “MAB Learning in IoT Networks”, Bonnefoi, **Besson** et al, [CROWNCOM](#), 2017
- ▶ “Aggregation of MAB for OSA”, **Besson**, Kaufmman, Moy, [IEEE WCNC](#), 2018
- ▶ “Multi-Player Bandits Revisited”, **Besson** & Kaufmann, [ALT](#), 2018
- ▶ “MALIN with GRC ...”, Bonnefoi, **Besson**, Moy, [demo at ICT](#), 2018
- ▶ “GNU Radio Implementation of MALIN ...”, **Besson** et al, [IEEE WCNC](#), 2019
- ▶ “UCB ... LPWAN w/ Retransmissions”, Bonnefoi, **Besson** et al, [IEEE WCNC](#), 2019
- ▶ “Decentralized Spectrum Learning ...”, Moy & **Besson**, [ISIoT](#), 2019
- ▶ “Analyse non asymptotique ...”, **Besson** & Kaufmann, [GRETSI](#), 2019

1 Preprints:

- ▶ “Doubling-Trick ...”, **Besson** & Kaufmann, [arXiv:1803.06971](#), 2018

3 Submitted works:

- ▶ “Decentralized Spectrum Learning ...”, Moy, **Besson** et al, for [Annals of Telecommunications](#), [July 2019](#)
- ▶ “GLRT meets *klUCB* ...”, **Besson** & Kaufmann & Maillard, for [AISTATS](#), [Oct.2019](#)
- ▶ “SMPyBandits ...”, **Besson**, for [JMLR MLOSS](#), [October 2019](#)



Thanks for your attention!

 **Questions & Discussion**

- ▶ an extension of our model of IoT network to account for retransmissions (Section 5.4),

- ▶ an extension of our model of IoT network to account for retransmissions (Section 5.4),
- ▶ my Python library [SMPyBandits](#) (Chapter 3),

- ▶ an extension of our model of IoT network to account for retransmissions (Section 5.4),
- ▶ my Python library [SMPyBandits](#) (Chapter 3),
- ▶ our proposed algorithm for aggregating bandit algorithms (Chapter 4),

- ▶ an extension of our model of IoT network to account for retransmissions (Section 5.4),
- ▶ my Python library [SMPyBandits](#) (Chapter 3),
- ▶ our proposed algorithm for aggregating bandit algorithms (Chapter 4),
- ▶ details about our algorithms, their precise theoretical results and proofs (Chapters 6 & 7),

- ▶ an extension of our model of IoT network to account for retransmissions (Section 5.4),
- ▶ my Python library [SMPyBandits](#) (Chapter 3),
- ▶ our proposed algorithm for aggregating bandit algorithms (Chapter 4),
- ▶ details about our algorithms, their precise theoretical results and proofs (Chapters 6 & 7),
- ▶ our work on the “doubling trick” (to make an algorithm \mathcal{A} anytime and keep its regret bounds).

REFERENCES AND PUBLICATIONS



Check out the

“The Bandit Book”

by Tor Lattimore and Csaba Szepesvári
Cambridge University Press, 2019.

↪ tor-lattimore.com/downloads/book/book.pdf



Reach me (or Christophe or Émilie) out by email, if you have questions

`Lilian.Besson @ CentraleSupélec.fr`

↪ `perso.crans.org/besson/`

`Christophe.Moy @ Univ-Rennes1.fr`

↪ `moychristophe.wordpress.com`

`Emilie.Kaufmann @ Univ-Lille.fr`

↪ `chercheurs.lille.inria.fr/ekaufman`



Experiment with bandits by yourself!

Interactive demo on this web-page

↪ perso.crans.org/besson/phd/MAB_interactive_demo/

Use my Python library for simulations of MAB problems **SMPyBandits**

↪ SMPyBandits.GitHub.io & GitHub.com/SMPyBandits

- ▶ Install with `$ pip install SMPyBandits`
- ▶ Free and open-source (MIT license)
- ▶ Easy to set up your own bandit experiments, add new algorithms etc.



Welcome to SMPyBandits documentation!

Open-Source Python package for Single- and Multi-Players multi-armed Bandits algorithms.

A research framework for Single and Multi-Players Multi-Arms Bandits (MAB) Algorithms: UCB, KL-UCB, Thompson and many more for single-players, and MCTopM & RandTopM, MusicalChair, ALOHA, MEGA, rhoRand for multi-players simulations. It runs on Python 2 and 3, and is publically released as an open-source software under the MIT License.

Note

See more on the [GitHub page](https://github.com/SMPyBandits/SMPyBandits/) for this project: <https://github.com/SMPyBandits/SMPyBandits/>. The project is also hosted on [Inria GForge](https://bandits.lillia.inria.fr/), and the documentation can be seen online at <https://smpybandits.github.io/> or <http://http://bandits.lillia.inria.fr/> or <https://smpybandits.readthedocs.io/>.

This repository contains the code of my numerical environment, written in Python, in order to perform numerical simulations on single-player and multi-players Multi-Armed Bandits (MAB) algorithms.

Open Source? **Yes!** Maintained? **yes** Ask me **anything** **python** **python 2.7 | 3.4 | 3.5 | 3.6** **docs** **passing** **bulletproof** **implementation** **cpython**

I (Lilian Besson) have started my PhD in October 2016, and this is a part of my on going research since December 2016.

How to cite this work?

If you use this package for your own work, please consider citing it with this piece of BibTeX:

```
@misc{SMPyBandits,
  title = {{SMPyBandits: an Open-Source Research Framework for S},
  author = {Lilian Besson},
  year = {2018},
  url = {https://github.com/SMPyBandits/SMPyBandits/},
  howpublished = {Online at: \url{GitHub.com/SMPyBandits/SMPyBandits/}},
  note = {Code at https://github.com/SMPyBandits/SMPyBandits/},
}
```



Main references

- ▶ My PhD thesis (Lilian Besson)
“Multi-players Bandit Algorithms for Internet of Things Networks”
↪ Online at perso.crans.org/besson/phd/
↪ Open-source at [GitHub.com/Naareen/phd-thesis/](https://github.com/Naareen/phd-thesis/)
- ▶ My Python library for simulations of MAB problems, **SMPyBandits**
↪ [SMPyBandits.GitHub.io](https://github.com/Naareen/SMPyBandits)
- ▶ “The Bandit Book”, by Tor Lattimore and Csaba Szepesvári
↪ tor-lattimore.com/downloads/book/book.pdf
- ▶ “Introduction to Multi-Armed Bandits”, by Alex Slivkins
↪ [arXiv.org/abs/1904.07272](https://arxiv.org/abs/1904.07272)



List of publications

Cf.: [CV.archives-ouvertes.fr/lilian-besson](https://cv.archives-ouvertes.fr/lilian-besson)



- ▶ *Decentralized Spectrum Learning for IoT Wireless Networks Collision Mitigation*, by Christophe Moy & **Lilian Besson**.
1st International ISIoT workshop, at *Conference on Distributed Computing in Sensor Systems*, Santorini, Greece, [May 2019](#).
See *Chapter 5*.
- ▶ *Upper-Confidence Bound for Channel Selection in LPWA Networks with Retransmissions*, by Rémi Bonnefoi, **Lilian Besson**, Julio Manco-Vasquez & Christophe Moy.
1st International MOTIoN workshop, at *WCNC*, Marrakech, Morocco, [April 2019](#).
See *Section 5.4*.
- ▶ *GNU Radio Implementation of MALIN: “Multi-Armed bandits Learning for Internet-of-things Networks”*, by **Lilian Besson**, Rémi Bonnefoi & Christophe Moy.
Wireless Communication and Networks Conference, Marrakech, [April 2019](#).
See *Section 5.3*.

For more details, see: CV.Archives-Ouvertes.fr/lilian-besson.



- ▶ *Multi-Player Bandits Revisited*,
by **Lilian Besson** & Émilie Kaufmann.
Algorithmic Learning Theory, Lanzarote, Spain, [April 2018](#).
See Chapter 6.
- ▶ *Aggregation of Multi-Armed Bandits learning algorithms for Opportunistic Spectrum Access*,
by **Lilian Besson**, Émilie Kaufmann & Christophe Moy.
Wireless Communication and Networks Conference, Barcelona, Spain, [April 2018](#).
See Chapter 4.
- ▶ *Multi-Armed Bandit Learning in IoT Networks and non-stationary settings*,
by Rémi Bonnefoi, **L.Besson**, C.Moy, É.Kaufmann & Jacques Palicot.
Conference on Cognitive Radio Oriented Wireless Networks, Lisboa, Portugal,
[September 2017](#). **Best Paper Award**.
See Section 5.2.

- ▶ *MALIN: “Multi-Arm bandit Learning for Iot Networks” with GRC: A TestBed Implementation and Demonstration that Learning Helps*, by **Lilian Besson**, Rémi Bonnefoi, Christophe Moy.
Demonstration presented in *International Conference on Communication*, Saint-Malo, France, [June 2018](#).
See [YouTu.be/HospLNQhcMk](https://youtu.be/HospLNQhcMk) for a 6-minutes presentation video.
See *Section 5.3*.

- ▶ *Analyse non asymptotique d'un test séquentiel de détection de ruptures et application aux bandits non stationnaires* (in French),
by **Lilian Besson** & Émilie Kaufmann,
GRETSI, August 2019.
See Chapter 7.

Submitted works. . .

- ▶ *Decentralized Spectrum Learning for Radio Collision Mitigation in Ultra-Dense IoT Networks: LoRaWAN Case Study and Measurements*,
by Christophe Moy, **Lilian Besson**, G. Delbarre & L. Toutain, [July 2019](#).
Submitted for a special volume of [the Annals of Telecommunications](#) journal, on
“Machine Learning for Intelligent Wireless Communications and Networking”.
See Chapter 5.
- ▶ *The Generalized Likelihood Ratio Test meets $klUCB$: an Improved Algorithm for Piece-Wise Non-Stationary Bandits*,
by **Lilian Besson** & Émilie Kaufmann & Odalric-Ambrym Maillard, [October 2019](#).
Submitted for [AISTATS 2020](#). Preprint at [HAL.Inria.fr/hal-02006471](https://hal.inria.fr/hal-02006471).
See Chapter 7.
- ▶ *SMPyBandits: an Open-Source Research Framework for Single and Multi-Players Multi-Arms Bandits (MAB) Algorithms in Python*,
by **Lilian Besson**
Active development since October 2016, [HAL.Inria.fr/hal-01840022](https://hal.inria.fr/hal-01840022).
It currently consists in about 45000 lines of code, hosted on
[GitHub.com/SMPyBandits](https://github.com/SMPyBandits), and a complete documentation accessible on
SMPyBandits.rtf.d.io or [SMPyBandits.GitHub.io](https://SMPyBandits.github.io).
Submitted for [JMLR MLOSS](#), in [October 2019](#).
See Chapter 3.



- ▶ *What Doubling-Trick Can and Can't Do for Multi-Armed Bandits*,
by **Lilian Besson** & Émilie Kaufmann, September 2018.
Preprint at [HAL.Inria.fr/hal-01736357](https://hal.inria.fr/hal-01736357).

I included here some extra slides. . .

- ▶ pseudo code of Rand-Top- M + kl-UCB
- ▶ pseudo code of MC-Top- M + kl-UCB
- ▶ exact regret bound of MC-Top- M + kl-UCB

- ▶ pseudo code of GLRT + kl-UCB
- ▶ exact regret bound of GLRT + kl-UCB

Our algorithm Rand-Top- M

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

Algorithm 6.1: The RandTopM decentralized learning policy (for an index policy U^j).



Our algorithm MC-Top- M

```
1 Let  $A^j(0) \sim \mathcal{U}([K])$  and  $C^j(0) = \text{False}$  and  $s^j(1) = \text{False}$ 
2 for  $t = 1, \dots, T$  do
3   if  $A^j(t-1) \notin \widehat{M}^j(t)$  then // transition (3) or (5)
4      $A^j(t) \sim \mathcal{U}(\widehat{M}^j(t) \cap \{k : U_k^j(t-1) \leq U_{A^j(t)}^j(t-1)\})$  // not empty
5      $s^j(t) = \text{False}$  // aim at an arm with a smaller UCB at  $t-1$ 
6   else if  $C^j(t-1)$  and  $\overline{s^j(t-1)}$  then // collision and not fixed
7      $A^j(t) \sim \mathcal{U}(\widehat{M}^j(t))$  // transition (2)
8      $s^j(t) = \text{False}$ 
9   else // transition (1) or (4)
10     $A^j(t) = A^j(t-1)$  // stay on the previous arm
11     $s^j(t) = \text{True}$  // become or stay fixed on a "chair"
12    Play arm  $A^j(t)$ , get new observations (sensing and collision),
13    Compute the indices  $U_k^j(t+1)$  and set  $\widehat{M}^j(t+1)$  for next step.
14 end
```

Algorithm 6.2: The MCTopM decentralized learning policy (for an index policy U^j).



Multi-Players Multi-Armed Bandits

Lemma 6.10. *For any $\mu \in \mathcal{P}_M$, let player $j \in [M]$ use the RandTopM-, MCTopM- or RhoRand-kl-UCB decentralized policy with exploration function $f(t) \doteq \ln(t) + 3 \ln(\ln(t))$. Then for any sub-optimal arm $k \in M$ -worst there exists problem-dependent constants $C_\mu, D_\mu > 0$ such that*

$$\mathbb{E}_\mu[N_k^j(T)] \leq \frac{\ln(T)}{\text{kl}(\mu_k, \mu_M^*)} + \underbrace{C_\mu \sqrt{\ln(T)} + D_\mu \ln(\ln(T)) + 3M + 1}_{=o(\ln(T))}. \quad (6.19)$$

Lemma 6.14. For any $\mu \in \mathcal{P}_M$, if all players use the MCTopM-kl-UCB decentralized policy, and $M \leq K$, then the total average number of collisions (on all arms) is upper-bounded by

$$\mathbb{E}_\mu \left[\sum_{k=1}^K C_k(T) \right] \leq M^2 (2M + 1) \left(\sum_{\substack{a,b=1,\dots,K \\ \mu_a < \mu_b}} \frac{1}{\text{kl}(\mu_a, \mu_b)} \right) \ln(T) + o(\ln T). \quad (6.26)$$

Theorem: regret for MC-Top- M with kl-UCB

Theorem 6.15. *If all M players use MCTopM-kl-UCB, and $M \leq K$, then for any problem $\boldsymbol{\mu} \in \mathcal{P}_M$, there exists a problem dependent constant $G_{M,\boldsymbol{\mu}}$, such that the regret satisfies:*

$$R_T^A(\boldsymbol{\mu}, M) \leq G_{M,\boldsymbol{\mu}} \ln(T) + o(\ln T). \quad (6.31)$$

Moreover, the dependency of the constant regarding the number of players is $G_{M,\boldsymbol{\mu}} = \mathcal{O}(M^3)$.



Our algorithm GLRT and kl-UCB

```
1 Input: Parameters: exploration rate  $\omega \in (0, 1)$ , confidence level  $\delta > 0$ 
2 Input: Option: Local or Global restart
3 initialization:  $\forall k \in [K], \tau_k = 0$  and  $n_k = 0$ ;
4 for  $t = 1, 2, \dots, T$  do
5   if  $t \bmod \lfloor \frac{K}{\omega} \rfloor \in [K]$  then // forced exploration
6      $A(t) = t \bmod \lfloor \frac{K}{\omega} \rfloor$ ;
7   else
8      $A(t) \in \mathcal{U}(\arg \max_{k \in [K]} \text{UCB}_k(t))$ , with  $\text{UCB}_k(t)$  defined in (7.13);
9     Play arm  $A(t)$ :  $n_{A(t)} = n_{A(t)} + 1$ ;
10    Observe the reward  $Y_{A(t),t}$ :  $Z_{A(t),n_{A(t)}} = Y_{A(t),t}$ ;
11    if  $\text{GLR}_\delta(Z_{A(t),1}, \dots, Z_{A(t),n_{A(t)}}) = \text{True}$  then // change-point is detected
12      if Global restart then
13         $\forall k \in [K], \tau_k = t$  and  $n_k = 0$ ; // restart all arms
14      else
15         $\tau_{A(t)} = t$  and  $n_{A(t)} = 0$ ; // restart only this arm
16 end
```

Algorithm 7.1: The GLR-klUCB algorithm, with **Local** or **Global** restarts.



Theorem: regret bound for GLRT + kl-UCB (global)

Theorem 7.8. For ω and δ for which Assumption 7.7 is satisfied, the regret of GLR-klUCB with parameters ω and δ based on **Global** Restart satisfies the following finite-time regret bound

$$R_T \leq 2 \sum_{i=1}^{\Upsilon_T} \frac{4K}{\omega (\Delta^{(i)})^2} \beta(T, \delta) + \omega T + \delta(K+1)\Upsilon_T \quad (7.14)$$
$$+ \sum_{k=1}^K \sum_{\substack{i=1, \dots, \Upsilon_T \\ \mu_k^{(i)} \neq \mu_{k^*}^{(i)}}} \frac{(\mu_{k^*}^{(i)} - \mu_k^{(i)})}{\text{kl}(\mu_k^{(i)}, \mu_{k^*}^{(i)})} \ln(T) + \mathcal{O}\left(\sqrt{\ln(T)}\right).$$

7.6 Finite-time upper-bounds on the regret of GLR-klUCB

1. Choosing $\omega = \sqrt{\ln(T)/T}$, $\delta = 1/\sqrt{T}$ (with no prior knowledge of Υ_T) gives

$$R_T = \mathcal{O}\left(\frac{K}{(\Delta^{\text{change}})^2} \Upsilon_T \sqrt{T \ln(T)} + \frac{(K-1)}{\Delta^{\text{opt}}} \Upsilon_T \ln(T)\right), \quad (7.15)$$

2. Choosing $\omega = \sqrt{\Upsilon_T \ln(T)/T}$, $\delta = 1/\sqrt{\Upsilon_T T}$ (with prior knowledge of Υ_T) gives

$$R_T = \mathcal{O}\left(\frac{K}{(\Delta^{\text{change}})^2} \sqrt{\Upsilon_T T \ln(T)} + \frac{(K-1)}{\Delta^{\text{opt}}} \Upsilon_T \ln(T)\right). \quad (7.16)$$

Theorem: regret bound for GLRT + kl-UCB (local)

Theorem 7.11. For ω and δ for which Assumption 7.10 is satisfied, the regret of GLR-klUCB with parameters ω and δ based on Local Restart satisfies the following finite-time regret bound

$$R_T \leq 2 \sum_{k=1}^K \sum_{\ell=1}^{NC_k} \frac{4K}{\omega \left(\Delta_k^{(\ell)}\right)^2} \beta(T, \delta) + \omega T + 2\delta C_T + \sum_{k=1}^K \sum_{\ell=1}^{NC_k} \frac{\ln(T)}{\text{kl}(\bar{\mu}_k^{(\ell)}, \mu_{i,\ell}^*)} + \mathcal{O}\left(\sqrt{\ln(T)}\right), \quad (7.17)$$

where $\mu_{i,\ell}^* \doteq \inf \left\{ \mu_{k_t^*}(t) : \mu_{k_t^*}(t) \neq \bar{\mu}_k^{(\ell)}, t \in [\tau_k^{(\ell)} + 1, \tau_k^{(\ell+1)}] \right\}$.

Corollary: regret bounds for GLRT + kl-UCB (local)

Corollary 7.12. For “easy” problems satisfying the corresponding Assumption 7.10, with Δ^{opt} and Δ^{change} defined as in Corollary 7.9, then the regret of GLR-klUCB with parameters ω and δ based Local Restarts satisfies

1. Choosing $\omega = \sqrt{\ln(T)/T}$, $\delta = 1/\sqrt{T}$ (with no prior knowledge of Υ_T or C_T) gives

$$R_T = \mathcal{O}\left(\frac{K}{(\Delta^{\text{change}})^2} C_T \sqrt{T \ln(T)} + \frac{C_T}{(\Delta^{\text{opt}})^2} \ln(T)\right), \quad (7.18)$$

2. Choosing $\omega = \sqrt{\Upsilon_T \ln(T)/T}$, $\delta = 1/\sqrt{\Upsilon_T T}$ (with prior knowledge of Υ_T and “optimist” guess $\Upsilon_T \simeq C_T \ll K \Upsilon_T$) gives

$$R_T = \mathcal{O}\left(\frac{K^2}{(\Delta^{\text{change}})^2} \sqrt{\Upsilon_T T \ln(T)} + \frac{K \Upsilon_T}{(\Delta^{\text{opt}})^2} \ln(T)\right), \quad (7.19)$$

3. Choosing $\omega = \sqrt{C_T \ln(T)/T}$, $\delta = 1/\sqrt{C_T T}$ (with prior knowledge of C_T) gives

$$R_T = \mathcal{O}\left(\frac{K}{(\Delta^{\text{change}})^2} \sqrt{C_T T \ln(T)} + \frac{C_T}{(\Delta^{\text{opt}})^2} \ln(T)\right), \quad (7.20)$$

4. Choosing $\omega = \sqrt{K \Upsilon_T \ln(T)/T}$, $\delta = 1/\sqrt{K \Upsilon_T T}$ (with prior knowledge of Υ_T and “pessimist” guess $C_T \simeq K \Upsilon_T$) gives

$$R_T = \mathcal{O}\left(\frac{K}{(\Delta^{\text{change}})^2} \sqrt{C_T T \ln(T)} + \frac{C_T}{(\Delta^{\text{opt}})^2} \ln(T)\right). \quad (7.21)$$

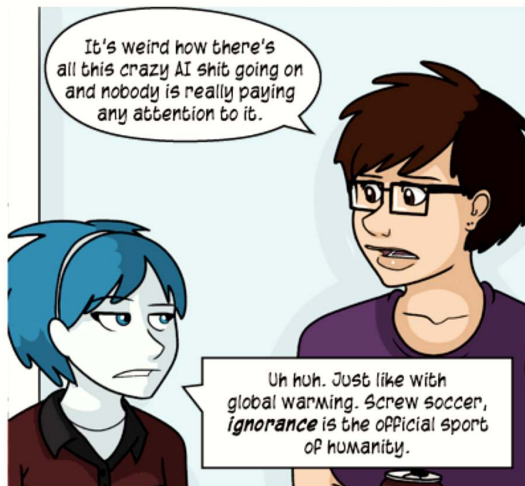


End of backup slides

Thanks for your attention!



What about the climatic crisis?



Copyright 2003-2015 J. Jacques

© Jeph Jacques, 2015, QuestionableContent.net/view.php?comic=3074

