Bandits à Mémoire pour la prise de décision en environnement dynamique. Application à l'optimisation des réseaux de télécommunications

Réda Alami

Thèse de doctorat

NNT : 2021UPASG063

université PARIS-SACLAY

Bandits à mémoire pour la prise de décision en environnement dynamique

Application à l'optimisation des réseaux de télécommunications

Memory bandits for decision making in dynamic environments

Application to network optimization

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Mathématiques et Informatique
Unité de recherche : *Université Paris-Saclay, CNRS, Laboratoire interdisciplinaire des sciences du numérique, 91405, Orsay, France*
Référent : Faculté des sciences d'Orsay

Thèse présentée et soutenue à Paris-Saclay, le 12/10/2021, par

Réda ALAMI

Composition du Jury

Florence D'ALCHE-BUC Professeure, TELECOM Paris	Présidente
Aurélien GARIVIER Professeur, ENS Lyon	Rapporteur & Examinateur
Vianney PERCHET Professeur, ENSAE	Rapporteur & Examinateur
Florence D'ALCHE-BUC Professeure, TELECOM Paris	Examinatrice
Aurélien BELLET Chargé de recherche, INRIA-Lille	Examinateur

Direction de la thèse

Michèle SEBAG Directrice de recherche, INRIA	Directrice de thèse
Odalric-Ambrym MAILLARD Chargé de recherche, INRIA-Lille	Co-Directeur de thèse
Raphael FERAUD Ingénieur de recheche, Orange	Co-Encadrant, tuteur en entreprise

Abstract

In this PhD thesis, we study the non-stationary multi-armed bandit problem where the non-stationarity behavior of the environment is characterized by several abrupt changes called "change-points". We propose **Memory Bandits**: a combination between an algorithm for the stochastic multi-armed bandit and the Bayesian Online Change-point detector (BOCPD).

The analysis of the latter has always been an open problem in the statistical and sequential learning theory community. For this reason, we derive a variant of the Bayesian Online Change-point detector which is easier to mathematically analyze in term of false alarm rate and detection delay (which are the most common criteria for online change-point detection).

Then, we introduce the decentralized exploration problem in the multi-armed bandit paradigm where a set of players collaborate to identify the best arm by asynchronously interacting with the same stochastic environment. We propose a first generic solution called decentralized elimination: which uses any best arm identification algorithm as a subroutine with the guarantee that the algorithm ensures privacy, with a low communication cost.

Finally, we perform an evaluation of the multi-armed bandit strategies in two different context of telecommunication networks.

First, in LoRaWAN (Long Range Wide Area Network) context, we propose to use multi-armed bandit algorithms instead of the default algorithm ADR (Adaptive Data Rate) in order to minimize the energy consumption and the packet losses of end-devices.

Then, in a IEEE802.15.4-TSCH context, we perform an evaluation of 9 multi-armed bandit algorithms in order to select the ones that choose high-performance channels, using data collected through the FIT IoT-LAB platform. The performance evaluation suggests that our proposal can significantly improve the packet delivery ratio compared to the default TSCH operation, thereby increasing the reliability and the energy efficiency of the transmissions.

Keywords: Non-stationary multi-armed bandits, online change-point detection, network optimization, exploration in multi-armed bandits

Contents

List of Figures

List of Tables

List of Algorithms

Part I

General Introduction

Chapter 1

Introduction

1.1 Motivations

Consider the situation in which a gambler is in front of a row of slot machines (a.k.a. one-armed bandit) and has to decide which one to play in order to collect the maximum amount of money. Each machine has a different winning probability, meaning that the gambler can maximize his earnings by playing the arm with highest winning probability. However, no prior information is given to the player. For this reason, the gambler initially may want to invest her/his money in trying different slot machines and record all the rewards she/he collected. As soon as there is enough evidence that an option is better than another, the player will tend to only play consistently the best identified arm, with the purpose of maximizing the total reward he/she gained during this process. This and many other real-world problems can be modeled as an online decision making problem, and, more specifically, a Multi-Armed Bandit (MAB) problem. This framework has been introduced by [135, 95, 120] and takes its name from the aforementioned gambling example. The problem concerns an agent whose goals is to maximize its reward (or equivalently to minimize the loss) gained during the process of learning the option providing the largest reward.

In the literature, the MAB framework is regarded as one of the most fundamental formalization of the sequential decision making problem, and in particular an illustration of the Exploration vs. Exploitation dilemma:

- Exploration : the agent is assumed to have no prior information about the machine payoffs. This assumption implies the need for the agent to play, i.e. to explore, arms that were not or rarely tried.
- Exploitation : in order to gather the maximal amount of reward, the agent should play as often as possible, or exploit, the arms with estimated best payoffs.

1.2 General problem statement

The multi-armed bandit setting considers a finite set of A actions or bandit arms. For $a \in \mathcal{A} = \{1 \dots A\}$, the a -th arm is associated with a reward distribution $\mathbf{P}_a$ with expectation $\mu_a \stackrel{\text{def}}{=} \mathbb{E}[X \sim \mathbf{P}_a]$. At a given time step t, the agent choose, based on the previous observations, an arm $A_t \in \mathcal{A}$ and observes an instantaneous reward $X_t \sim \mathbf{P}_{A_t}$ Letting T denote a finite time horizon ($T \in \mathbb{N}^\star$) , the agent's goal is to find a policy π that maximizes

the cumulative sum of instantaneous rewards:

$$\mathcal{S}_T^\pi = \sum_{t=1}^{T} X_t$$

1.3 Some applications of the multi-armed bandit domain

This section presents some applications presented in the multi-armed bandit literature. This non-exhaustive list aims to illustrate the wide variety of successful applications of the multi-armed bandit framework and motivate its study in the present document.

1.3.1 A/B testing

A/B testing is a standard approach for evaluating select variants in an online setting [77]. As the name implies, the technique is an experiment to determine the performance of two options, "A" and "B". These could be ad banners or web-page formatting styles, for example. Option "A" generally represents what is currently in use and acts as a control to compare to option "B," though this need not be the case. In a real setting, any number of alternative options can be tested at the same time.

During an A/B test, each option is presented to an equal number of viewers to explore its performance. After the test concludes, the best option is identified and used exclusively, exploiting the knowledge that was gained from the test. One drawback with A/B testing is that it incurs "regret," which is the concept that during the test, inferior options were presented to some viewers who may have had a better interaction with a better choice.

With a regular application of A/B testing, the regret quietly affects a company's bottom line over time, through less-than-ideal user experiences. For any company that regularly runs tests to find an optimal version, minimizing regret as much as possible might provide a significant advantage.

Bandit algorithms reduce the amount of regret that occurs with A/B tests because they continuously balance exploration with exploitation. After every new sample, the knowledge that was learned is used to make a better choice the next time around. Over time, the options that perform better are used more often than the underperformers, and eventually the best option wins out.

1.3.2 Recommendation services

Many online streaming services like Netflix, Amazon Prime, Hulu, HBO now, recommend movies or TV shows to their users in order to increase the users' engagement with the service. If the service makes good recommendations, especially early in user adoption, then the users will watch more content. Incorporating methods that adaptively respond to new users, like multi- armed bandits, can provide a better service and generate more user engagement.

1.3.3 Network routing

Another problem with an interesting structure is network routing, where the learning agent tries to control the internet traffic through the shortest possible path on a network [52]. At each round, the learning agent receives the start/end destinations for a packet of data. The set of actions corresponds to the set of all possible paths starting and ending at the

appropriate points on some graph assumed to be known. The feedback in this case is the time it takes for the packet to be received at its destination and the reward is the negation of this value.

Again the action set is combinatorially large with even relatively small graphs possessing an enormous number of paths. The routing problem can obviously be applied to more physical networks such as transportation systems used in operations research.

1.3.4 Dynamic pricing

One of the most important issues that any retailer has to solve is how to properly price items [56]. Setting the prices too high or too low can cause bad customer experience and dramatically effect the bottom-line of the retailer. The size of the catalog carried by a brick-and-mortar store is very small, and because of the difficulty in changing prices, pricing of an items is an easy problem and can be solved by carrying out elementary data analysis. However, the presence of a large collection of products on an e-commerce website, the ability to collect large amounts of user behaviour data easily, and also the relative ease with which prices change can be made on a large scale, makes the problem of pricing both challenging and rich. In traditional brick-and-mortar stores price of items tend to remain unchanged. However, such static pricing policies do not work in an e-commerce setting. Static pricing does not simulate the demand/price curve, nor does it allow a firm to respond to competitor price changes. Moreover static pricing is unable to fully exploit the power of computation, data, and the ease of implementing price changes in an online world, As a result, dynamic pricing is the pricing policy of choice in an e-commerce setting.

1.3.5 Tree search

The Upper Confidence Tree search algorithm (UCT) [86] is an algorithm used when playing a so-called perfect information game. In short, perfect information games are games in which, at any point in time, each player has perfect information about all event actions that have previously taken place. Examples of such games are Chess, Go or Tic-Tac-Toe. The idea is to iteratively build a search tree where in each iteration the algorithm takes three steps:

1. Chooses a path from the root to a leaf.
2. Expands the leaf (if possible).
3. Performs a Monte-Carlo roll-out to the end of the game.

The multi armed bandit algorithm is used in order to select the path from the root to the leaves. At each node in the tree, the bandit selects the child based on the series of rewards observed through that node so far. The resulting algorithm can be analyzed theoretically, but more importantly has demonstrated outstanding empirical performance in game playing problems.

1.4 Importance of the non-stationary multi-armed bandit problem

We propose that a switching environment is a good model for a wide variety of decision-making scenarios. In this section, in order to motivate our work we briefly explore a few

scenarios that we deem should exhibit switching behaviour.

1.4.1 Game optimization

Game playing is a typical situation where it is asked to make decisions under uncertainty. Two or more agents compete against each other and sometimes cooperate with each other in order to earn rewards in a game. For example in the famous card game Poker, there is a number of agents who play together in a game [38]. Cards are dealt to each of the players in a series of turns so that each player ignores the cards that the others have been dealt. A player must decide whether he wishes to bid, call or fold in order to maximize the size of his winnings. The strategy that leads to the greatest payouts for a player depends not only on their hands and those of all other players, but also on the strategies of other competitors (how other agents decide to bid, call, or bid). sleep). The strategies of the competitors can be assumed to be stationary, so that in the same hand the player behaves according to the same statistical rule. However, a common situation is that the player learns from experience and adapts his policy based on the results of previous games. This means that the competing agent's strategy will change (given the same observable state, the expected decisions will be different) during playing time. This in itself is an argument for an agent to adapt to non-stationary environments within the framework of the game.

1.4.2 Financial markets

The bandit model for decision-making has been widely employed in the economic field in order to model problems in financial markets. One of the first use of the bandit model was due to Rothschild [122]. Rothschild was focusing on the process by which companies learn to estimate the market demand for their products and services. As an illustration, let us consider a single company estimating the market demand. The actual demand is given by a probability distribution over the consumer valuations observable to the company. The demand is assumed to be one of a finite set of possible values each with a prior probability of being the true demand. The company can then set its prices in a sequence of rounds and observe consumer valuations in order to learn what the true demand is. The problem of Rothschild was assumed to be stationary. In [82], the authors have also considered a similar problem to Rothschild, but the essential difference is that they have assumed that the unknown demand is non-stationary. In the context of Venture capital funding, the author of [128] has also considered bandit-like models, where a Venture Capitalist aims at assessing the continued investment in some start-up companies.

Several research works on the application of bandit problems to financial markets are summarised by the survey of Dirk Bergemann and Juuso Valimaki [21].

1.4.3 Network management

Networks and graph structures arise in several applications from computer networks to road networks [101]. A network is defined by a set of nodes connected by edges. For instance, in a computer network, the computers are represented by nodes, and the edges represent communication channels between the nodes. Such networks are prone to various kinds of failure. In computer network, hardware is indeed prone to failure causing nodes to become unreachable, and thus the network structure changes fundamentally.

We can consider that such unpredictable changes constitute a abrupt change in the network behaviour. Thus, we believe that decision-making systems operating in environments dependent on the behaviour of networks should be adaptable to sudden abrupt changes as with a piece-wise stationary environment.

1.5 Contributions

The presented thesis brings five contributions, three of them are related to the multi-armed bandits and statistical learning theory community and the other two are related to the communication networks optimization community.

In this section, we provide a summary of these five contributions.

1.5.1 Contribution n° 1: memory bandits for the piece-wise stationary stochastic multi-armed bandit problem

An adaptation of the switching Thompson sampling for non-stationary multi-armed bandits is provided. The adaptation is based on expert aggregation instead of runlength distribution sampling. Empirically, the proposal compares favorably against the original switching Thompson Sampling for both the global and the per-arm switching settings. (This contribution gave rise to a paper at *NIPS-BayesOpt 2017* workshop [4]).

1.5.2 Contribution n° 2: the restarted Bayesian online change-point detection strategy

A modified version of the original Bayesian online change-point detector is provided. The proposal is easier to analyse in term of false alarm rate and detection delay. A complete mathematical analysis of the proposal is provided. (This contribution gave rise to a paper at *ICML 2020* [5]).

1.5.3 Contribution n° 3: decentralized exploration in multi-armed bandits

The decentralized exploration problem for multi-armed bandit is introduced. This problem deals with a set of players aiming at collaborating in order to identify the best arm in an asynchronous fashion. A generic algorithm called Decentralized Elimination is provided with the guarantees that the privacy is ensured while the communication costs are low. (This contribution gave rise to a paper at *ICML 2019* [47]).

1.5.4 Contribution n° 4: LoRa network optimization via multi-armed bandits

The optimization of the LoRaWAN technology performances is demonstrated via multi-armed bandit algorithms where this class of learning algorithms are able to manage the trade-off between energy consumption and packet loss. The multi-armed bandit algorithms are used to select the communication parameters that are spreading factor and emission power. In order to highlight the benefit of using MAB in this king of industrial use-case, the comparison against the default strategy called ADR is performed. (This contribution gave rise to a paper at *ICT 2018* [83]).

1.5.5 Contribution n° 5: TSCH network optimization via multi-armed bandits

In the context of Industrial Internet of Things, an evaluation of 9 multi-armed bandit algorithms against the default TSCH operation is performed where the benefit of using multi-armed bandit strategies for the selection of high performance channels is highlighted. (This contribution gave rise to a paper at *MSWiM'18* [39]).

1.6 Thesis structure

The manuscript is organized as follows. The formal background and algorithms for the multi-armed bandits are presented in chapter 2. Chapter 3 presents our first contribution, and details the proposed hybridization between the Thompson Sampling algorithm and the Bayesian online change-point detector in order to solve the piece-wise stationary stochastic multi-armed bandit in both global and per-arm switch cases. Obviously, this hybridization can naturally be generalized to any stochastic bandit. We are thus building the family of memory bandits. Chapter 4 presents our second contributions, a variant of the Bayesian online change-point detector which compares favorably with the original algorithm. We also provide a mathematical analysis of its optimality in term of false alarm rate and detection delay. Then, chapter 5 introduces the decentralized exploration problem in the multi-armed bandit paradigm where a set of players collaborate to identify the best arm by asynchronously interacting with the same stochastic environment. We propose a first generic solution called decentralized elimination: which uses any best arm identification algorithm as a subroutine with the guarantee that the algorithm ensures privacy, with a low communication cost.

Then the rest of the manuscript is devoted to evaluate multi-armed bandit strategies in two different context of telecommunication networks.

Indeed, chapter 6 proposes to use multi-armed bandit algorithms instead of the default algorithm ADR (Adaptive Data Rate) in order to minimize the energy consumption and the packet losses of end-devices in LoRaWAN (Long Range Wide Area Network) context.

Moreover, in a IEEE 802.15.4-TSCH context, chapter 7 performs the evaluation of 9 multi-armed bandit algorithms in order to select the ones that choose high-performance channels, using data collected through the FIT IoT-LAB platform. The performance evaluation suggests that our proposal can significantly improve the packet delivery ratio compared to the default TSCH operation, thereby increasing the reliability and the energy efficiency of the transmissions.

Finally, chapter 8 concludes this manuscript by discussing the presented contributions, and proposing future research directions.

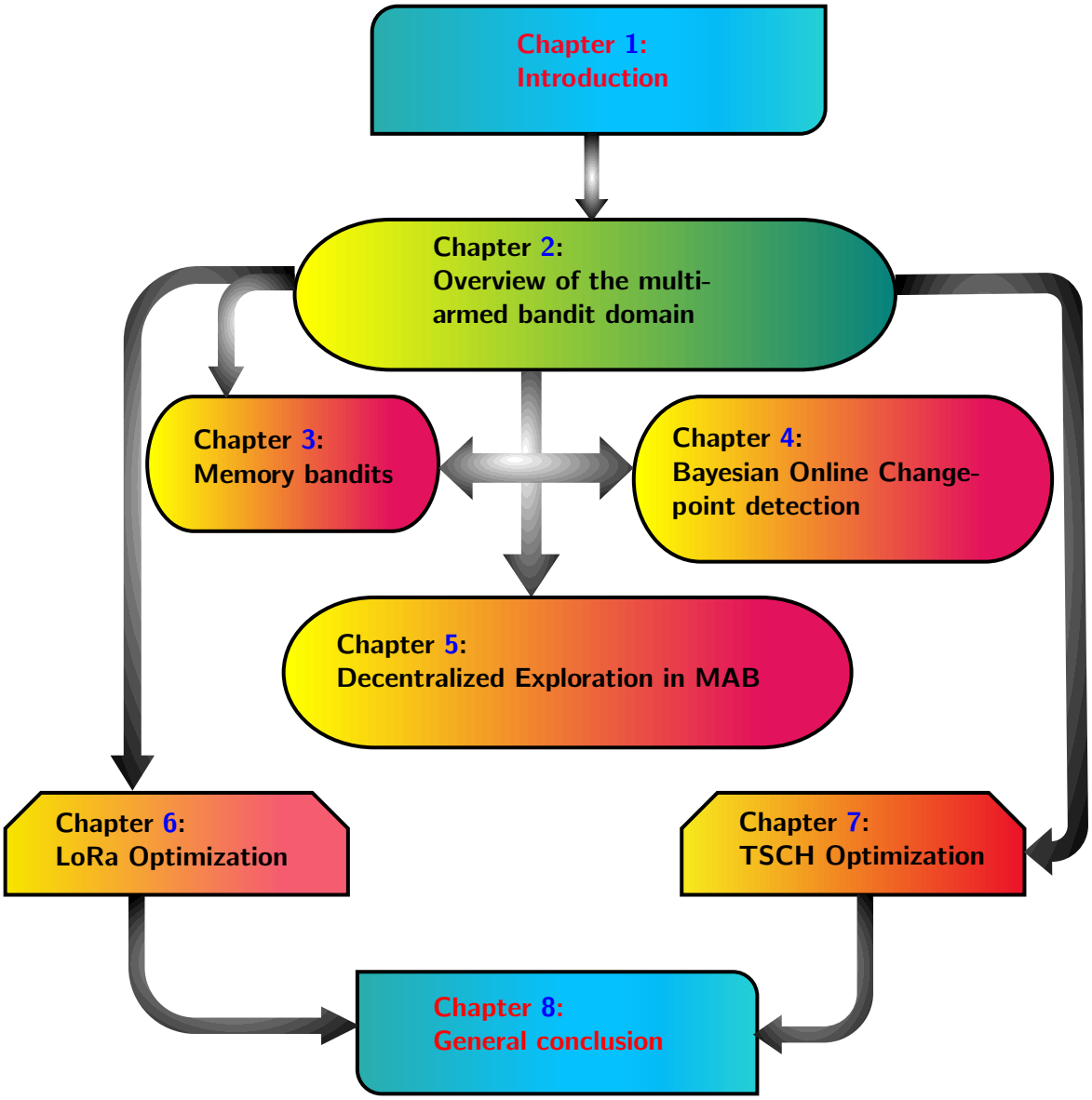


Figure 1.6.1: Thesis structure

Chapter 2

Multi-armed bandits: a general background

Overview

This chapter introduces the formal background of the multi-armed bandit problem (MAB). First, we introduce the overall overview of the multi-armed bandit world.

Then, after some definitions and notations, we present the different goals tackled by MAB settings together with the associated evaluation criteria and loss functions known as regret.

Finally, we present the whole state of the art of the stochastic and the adversarial bandit for the stationary and non-stationary setting.

2.1 An overview of the multi-armed bandit world

2.1.1 Multi-armed bandit model

A multi-armed bandit is formally equivalent to a one-step Markov Decision Process (MDP).

Definition 2.1: Multi-armed bandit
A MAB problem is a tuple $< \mathcal{A}, R >$ where:

- $\mathcal{A}$ is the set of A possible actions, called arms;
- $R := a \in \mathcal{A} \rightarrow R(a) \in \mathbb{R}$ is an unknown reward function depending on the chosen action a.

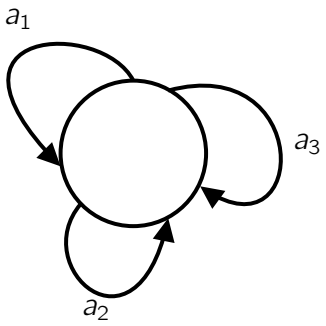


Figure 2.1.1: Formulation of a multi-armed bandit problem with 3 arms. When the agent selects an action, he receives a reward that depends on the chosen action and returns to the initial (unique) state.

2.1.2 Taxonomy of multi-armed bandits

Depending on the model of the reward function R, there are different variants of the MAB model. The taxonomy of MAB variants is summarized in Figure 2.1.2. Among the most important characteristics of a MAB, we have that a specific problem can be:

- *Stochastic vs Adversarial*: in a Stochastic Multi-Armed Bandit (S-MAB) [14], the rewards are generated from stochastic distributions. In an Adversarial Multi-Armed Bandit (A-MAB) [15], the rewards are generated by a process that cannot be treated as a stochastic distribution. In this latter setting, we say that the rewards are generated by an adversary, who may take advantage of those corner cases in which a bandit algorithm performs badly. As a consequence, adversarial algorithms must be robust to adversary choice of the rewards.
- *Stochastic Stationary vs Stochastic Non-Stationary*: in a Stationary Multi-Armed Bandit (ST-MAB), the rewards are generated from stochastic distributions which are stationary, i.e., they do not change over time. In a Non-Stationary Multi-Armed Bandit (NS-MAB), the stochastic distributions may change at each time.

Moreover, NS-MAB settings in which the mean reward of the arms changes at each round. Among them, we make the following distinction:

- *Abrupt changes vs Smooth changes*: we call Abruptly changing Non-Stationary Multi-Armed Bandit the setting in which the reward distributions change abruptly a number of times which is a function of the time horizon, but the magnitude of the change remains unbounded. We call Smoothly changing Non-Stationary Multi-Armed Bandit the setting in which the reward distribution parameters smoothly change at each time-step by less than a certain value.

This taxonomy is useful as it allows to correctly formalize a multi-armed bandit problem and solve it with the right tools. Indeed, it is often unrealistic to model the real world with stationary distributions and many problems can be modelled as non-stationary. We now categorize the applications of the MAB model presented in Chapter 1. In online recommendation systems, the designed algorithm must be able to adapt to the user having a particular interest on one day, and thus, this situation may be modeled as a non-stationary MAB. On the other hand, consider the task of sending packets between two nodes in a communication network, in which there are A possible channels and the decision-maker must choose one for

each channel in order to minimize the transmission cost. It is likely that the costs associated with each channel cannot be modeled with stationary stochastic distribution, thus this is a non-stationary MAB setting.

About Markovian bandits In addition to the stochastic and bandit, there is a third type of bandit model called "*Markovian*". A Markovian MAB model maps each arm a to a Markov chain [111] instead of a distribution (as for the stochastic bandit), and thus they are no longer stochastic. There is two different kinds of Markov model: the rested model and the restless one. For a bandit of A arms, each Markov chain has a finite number of states s, each corresponding to a (constant) reward that the agent obtains if he selects this arm while its Markov chain is in state s. Rested Markov models [134] means that only the state of the selected arm's Markov chain can change according to its Markov transition matrix. Restless models remove this hypothesis, making them harder to track and solve. Indeed in restless bandits [64], the state of the underlying Markov process controlling the evolution of the arm distributions is only sparsely revealed to the agent.

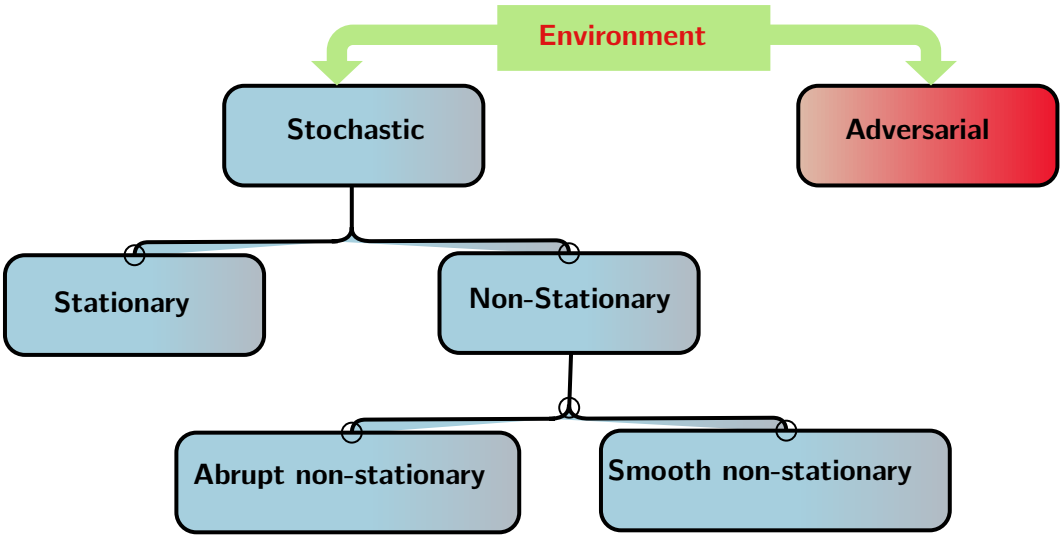


Figure 2.1.2: Taxonomy of different environments considered for the multi-armed bandit problem

2.2 Algorithms for Multi-armed bandits

2.2.1 Problem formulation

The multi-armed bandit problem (MAB) formalizes the fundamental exploration-exploitation dilemma that appears in decision making problems facing partial information, where decisions have to be taken over time (discrete turns) and impact both the rewards and the information withdrawn [92, 14, 28, 95]. Specifically, a set of A arms is available to the decision maker (player). At each turn, he has to choose one arm and receives a reward corresponding to the played arm, ignoring what the received reward would have been, if he had played another arm. The player faces the dilemma of *exploring*, that is playing an arm whose mean reward is loosely estimated in order to build a better estimate, or *exploiting*, that is playing a seemingly best arm based on current estimates in order to maximize its cumulative reward. The accuracy of the player policy at a given time horizon is typically measured in terms of *regret*, that is the difference between the cumulative rewards of the player and the one that

could have been acquired by a policy assumed to be optimal. The notion of optimality and hence the algorithms depend on the environment.

Notation 1 (Useful notations for the chapter)**.** *Let us introduce the notations which will be used in the rest of the document for stationary environments.*

- $\mathcal{A}$*: finite set of arms of whose cardinal is* $A > 0$*.*
- A_t*: arm chosen at time* t*.*
- X_t*: reward received at time* t *after playing arm* A_t*.*
- $\mathbf{P}_a$*: probability distribution of arm* $a \in \mathcal{A}$*.*
- μ_a*: expectation of the distribution* $\mathbf{P}_a$ *(interpreted as expected reward for arm* a*).*
- $\mu^\star$*: maximum expectation taken over all arms (i.e.* $\mu^\star = \max_{a \in \mathcal{A}} \mu_a$*).*
- Δ_a*: optimality gap of the* a^{th} *arm (i.e.* $\Delta_a = \mu^\star - \mu_a$*).*
- $N_{a,s} := \sum_{i=1}^{s} \mathbf{1}\{A_i = a\}$*: overall number of time-steps the* a^{th} *arm has been selected up to time* s*.*
- $\widehat{\mu}_{a,s} := \frac{1}{N_{a,s}} \sum_{i=1}^{s} X_i \times \mathbf{1}\{A_i = a\}$*: empirical mean of arm* a *up to time* s*.*
- $\widehat{\nu}_{a,s} := \frac{1}{N_{a,s}} \sum_{i=1}^{s} \left(X_i \times \mathbf{1}\{A_i = a\} - \widehat{\mu}_{a,s} \right)^2$*: empirical variance of arm* a *up to time* s*.*
- $T \in \mathbb{N}^\star$ *denotes the time horizon which might be finite or infinite.*
- $Y_{a,s}$*: the* s*-th selected reward drawn from distribution* $\mathbf{P}_a$*.*

2.2.2 Notion of regret for multi-armed bandits

Two definitions of regret are introduced in the cumulative gain maximization case, respectively referred to as regret and pseudo-regret. In the last case, the rewards associated to the oracle strategy are simply set to their expectation, $\mu^\star$.

Definition 2.2: Cumulative regret
With same notations as above, the `cumulative regret` of the agent π at time t is defined as:

$$\mathbf{R}_t^\pi \overset{\text{def}}{=} \max_{a \in \{1 \dots A\}} \sum_{s=1}^{t} Y_{a,s} - \sum_{s=1}^{t} X_s$$

and the expected cumulative regret takes the following form:

$$\mathbb{E}_{\mathrm{P}_1, \dots, \mathrm{P}_A}[\mathbf{R}_t^\pi] \overset{\text{def}}{=} \mathbb{E}\left[\max_{a \in \{1 \dots A\}} \sum_{s=1}^{t} Y_{a,s} - \sum_{s=1}^{t} X_s \right]$$

Definition 2.2 then tells us that the cumulative regret $\mathcal{R}_t^\pi$ of a MAB policy π at time t is the sum, up to the time t, of the differences between the rewards of the arm that would return the highest expected reward if always pulled, and the rewards of the arms selected by the policy π.

A different quantity is the cumulative pseudo-regret, in which we consider the expectation over the stochasticity of the reward process.

Definition 2.3: Cumulative pseudo regret
With same notations as above, the `cumulative pseudo-regret` of the MAB policy π at time t is defined as:

$$\mathcal{R}_t^\pi \overset{\text{def}}{=} \sum_{s=1}^{t} (\mu^\star - \mu_{A_s}) = t\mu^\star - \sum_{s=1}^{t} \mu_{A_s} = \sum_{a=1}^{A} \Delta_a N_{a,t}$$

and the expected cumulative pseudo-regret takes the following form:

$$\mathbb{E}\left[\mathcal{R}_t^\pi\right] = \sum_{a=1}^{A} \Delta_a \mathbb{E}\left[N_{a,t}\right]$$

Thus, definition 2.3 tells us that the cumulative pseudo-regret $\mathcal{R}_t^\pi$ of a MAB policy π at time t is the sum, up to round t, of the differences between the mean of the optimal arm $a^\star = \text{argmax}_{a \in \mathcal{A}} \mu_a$ and the expectations of the arms selected by the policy π.

Lower bounds on the expected cumulative pseudo-regret

In this section, we present the two main lower bounds of the stochastic bandit, namely the lower bound independent of the distributions (distribution free) [15] and the one distribution dependant [91].

Theorem 2.1: Distribution-free lower bound
Let `sup` denote the supremum of all stochastic bandits over the segment $[0, 1]$ and let `inf` denote the infimum over all the forecasters, the following bound holds true:

$$\texttt{infsup}\mathbb{E}\left[\mathcal{R}_t\right] \geqslant \frac{1}{20}\sqrt{tA} \tag{2.2.1}$$

Proof. See [15]. □

Before stating the distribution dependant lower bound in Theorem 2.2, we need to introduce the notion of Kullback-Leibler divergence in Definition 2.4.

Definition 2.4: Kullback Leibler divergence
Let $\mathscr{P}([0, 1])$ be the set of probability distributions over the segment $[0, 1]$. The Kullback-Leibler divergence between two distributions P and Q is defined as follows:

$$\mathbf{KL}\left(P, Q\right) = \begin{cases} \int_{[0,1]} \frac{dP}{dQ} \log \frac{dP}{dQ} dQ & \text{if } P \ll Q \\ +\infty & \text{otherwise} \end{cases} \tag{2.2.2}$$

Theorem 2.2: Distribution dependant lower bound
Let us consider a bandit strategy satisfying $\mathbb{E}[N_{a,t}] = o(t^r)$ for any sub-optimal arm a (i.e. arm a with $\Delta_a > 0$,) and any $r > 0$. Then, the following holds:

$$\liminf_{T\to\infty} \frac{\mathbb{E}[\mathcal{R}_T]}{\log T} \geqslant \sum_{a:\Delta_a>0} \frac{\Delta_a}{\mathscr{K}_{\inf}(\mathbf{P}_a, \mu^\star)} \tag{2.2.3}$$

where:

$$\mathscr{K}_{\inf}(\mathbf{P}_a, \mu^\star) = \inf\{\mathbf{KL}(\mathbf{P}_a, v) : \mathbb{E}_{X\sim v}[X] > \mu^\star\} \tag{2.2.4}$$

Proof. See [92]. □

2.2.3 Stationary stochastic multi-armed bandits (ST-MAB)

In this section, we introduce the common strategies that deal with the stochastic stationary settings. Stochastic multi-armed bandit problems, characterised by the presence of stochastic distributions, can be handled with three different reasoning methods:

- *Frequentist reasoning*: the parameters of the reward distribution are scalar unknown parameters. The strategy selects an arm at each time step based on the observed history. Some examples are the UCB1 [14], Upper Confidence Bound 1 Tuned (UCB1Tuned) [13] and Kullback Leibler Upper Confidence Bound (KL-UCB) [31] strategies.
- *Bayesian reasoning*: the parameters of the reward distribution are random variables with a prior distribution. The strategy selects an arm at each round based on the posterior distribution, built using both the past observed rewards and on the provided prior, which is updated as rewards (samples) are received. An example is the Thompson Sampling policy [81].
- *Hybrid frequentist and Bayesian reasoning*: there exists policies that combines both reasonings, for example BayesUCB [79].

Upper confidence bound

Upper confidence bound (UCB) is a suite of stationary MAB algorithms that use the frequentist approach [14]. Their strategy to cope with the exploration versus exploitation problem is to rely on the so-called optimism in face of uncertainty [92]: when the expected reward of an arm is uncertain and the probability of it being the optimal action is high enough, the policy favours the selection of that arm. As more samples are gathered, the estimates of the arms rewards become more and more accurate, lessening the effect of optimism and eventually choosing the action with highest mean reward. Following this heuristic, UCB policies consider an upper bound $\widehat{U}_{a,t}$ over the empirical mean $\widehat{\mu}_{a,t}$ of arm a and, at each time step, select the arm with the highest upper bound.

The bound used by such policies is designed such that, with high probability:

$$\widehat{U}_{a,t} := \widehat{\mu}_{a,t} + \widehat{B}_{a,t} \geqslant \mu_a$$

that is, the upper bound on the mean of arm a must be higher than the true mean of arm a with high probability. What is still left to define is $\widehat{B}_{a,t}$. In general, we want that:

- small $N_{a,t} \rightarrow$ large $\widehat{U}_{a,t}$, because the estimated value $\widehat{\mu}_{a,t}$ is uncertain;
- large $N_{a,t} \rightarrow$ small $\widehat{U}_{a,t}$, because the estimated value $\widehat{\mu}_{a,t}$ is accurate;

Definition 2.5: Chernoff-Hoeffding inequality
Let $X_1, ..., X_t$ be random variables with support in $[0, 1]$ and with identical mean $\mathbb{E}[X_i] = \mu$, then, for all $\varepsilon > 0$ the following holds:

$$\mathbb{P}\left(\left|\frac{1}{t}\sum_{i=1}^{t} X_i - \mu\right| \geqslant \varepsilon\right) \leqslant 2\exp\left(-2\varepsilon^2 \times t\right)$$

The pseudo-code of UCB 1 is described in Algorithm 2.1.

Algorithm 2.1 UCB 1 [14]

Require: T: Horizon, $\mathcal{A}$: Arm set
1: **Initialization:**
$\forall a \in \mathcal{A} \quad N_{a,0} = 0$
2: **Define:**

$$\mathfrak{I}_{a,t-1}^{\text{UCB1}} := \begin{cases} \infty & \text{if } N_{a,t-1} = 0 \\ \widehat{\mu}_{a,t-1} + \sqrt{\frac{2}{N_{a,t-1}}\log(t-1)} & \text{otherwise} \end{cases} \tag{2.2.5}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a\, \mathfrak{I}_{a,t-1}^{\text{UCB1}}$.
5: Observe X_t and update $\mathfrak{I}_{a,t}^{\text{UCB1}}$ according to Eq.(2.2.5).
6: **end for**

In term of theoretical guarantees, the regret upper bound of UCB 1 is given in Theorem 2.3.

Theorem 2.3: UCB 1 regret upper bound
For the stationary bandit $\Theta = (\mathbf{P}_1, ..., \mathbf{P}_A)$, the regret of UCB1 strategy after T rounds takes the following upper bound:

$$\mathbb{E}\left[\mathcal{R}_T^{\text{UCB1}}(\Theta)\right] \leqslant \left[8\sum_{a:\Delta_a>0}\left(\frac{\log T}{\Delta_a}\right)\right] + \left(1 + \frac{\pi^2}{3}\right)\left(\sum_{a=1}^{A}\Delta_a\right)$$

Proof. See [14]. □

From Theorem 2.3, the UCB 1 strategy is asymptotically order optimal; its cumulative pseudo-regret reaches the distribution dependant lower bound (Theorem 2.2) up to a multiplicative constant.

Variance estimates for Multi-Armed Bandit

UCB Tuned The UCB Tuned algorithm introduced in [13] exploits the idea of exploiting the empirical variance $\widehat{\nu}_{a,t}$ of the arm a to build its index at time t. Informally, everything else being equal, an arm with large variance should be explored more often than an arm with small variance. The pseudo-code of UCB Tuned is described in Algorithm 2.2.

Algorithm 2.2 UCB Tuned [13]

Require: T: Horizon, $\mathcal{A}$: Arm set, $p > 0$: hyper-parameter.

1: **Initialization:**
$\forall a \in \mathcal{A} \quad N_{a,0} = 0$

2: **Define:**

$$\mathfrak{B}'_{a,s,t} := \sqrt{\frac{2\widehat{\nu}_{a,s}\log(4t^p)}{s}} + \frac{16\log(4t^p)}{3s} \qquad \texttt{(Confidence sequences)}$$

$$\mathfrak{I}^{\text{UCBTuned}}_{a,t-1} := \begin{cases} \infty & \text{if } N_{a,t-1} = 0 \\ \widehat{\mu}_{a,t-1} + \mathfrak{B}'_{a,N_{a,t-1},t-1} & \text{otherwise} \end{cases} \tag{2.2.6}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a \min\{\mathfrak{I}^{\text{UCBTuned}}_{a,t-1}, 1\}$.
5: Observe X_t and update $\mathfrak{I}^{\text{UCBTuned}}_{a,t}$ according to Eq.(2.2.6).
6: **end for**

In term of theoretical guarantees, the regret upper bound of UCB Tuned is given in Theorem 2.4.

Theorem 2.4: UCB Tuned regret upper bound
For the stationary bandit $\Theta = (\mathbf{P}_1, ..., \mathbf{P}_A)$, if $p > 2$, then the regret of UCB Tuned strategy after T rounds takes the following upper bound:

$$\mathbb{E}\left[\mathcal{R}^{\text{UCBTuned}}_T(\Theta)\right] \leqslant 16\Big[\log(4) + p\log T\Big] \sum_{a:\Delta_a>0}\left(1 + \frac{\sigma_a^2}{2\Delta_a}\right) + 2\left(1 + \frac{1}{p-2}\right)\sum_{a:\Delta_a>0}\Delta_a$$

where: $\sigma_a^2 = \mathbb{E}_{X\sim \mathrm{P}_a}\left[\Big(X - \mathbb{E}_{X\sim \mathrm{P}_a}[X]\Big)^2\right]$ denotes the variance of the distribution $\mathbf{P}_a$.

Proof. See [13]. □

From Theorem 2.4, the UCB-Tuned strategy is asymptotically order optimal; its cumulative pseudo-regret reaches the distribution dependant lower bound (Theorem 2.2) up to a multiplicative constant.

UCB V Like UCB Tuned, the UCB V algorithm is also based on the use of empirical variance to construct a better estimate of the index of each arm. One can quite simply consider that the algorithm UCB V is a generalized version of the algorithm UCB Tuned, by using any non decreasing function $t \longrightarrow \mathcal{E}_t$ instead of the particular function $t \longrightarrow \log(4t^p)$. We describe the UCB V algorithm in Algorithm 2.3.

Algorithm 2.3 UCB V [12]

Require: $c \geqslant 0$, T: Horizon, $(\mathcal{E}_t)_{t\geqslant 0}$: $t \longrightarrow \mathcal{E}_t$ is non decreasing, $\mathcal{A}$: Arm set.

1: **Initialization:**
$\forall a \in \mathcal{A} \quad N_{a,0} = 0$

2: **Define:**

$$\forall a \in \mathcal{A} \quad \mathfrak{B}_{a,s,t} := \sqrt{\frac{2\widehat{\nu}_{a,s}\mathcal{E}_t}{s}} + c \times \frac{3b\mathcal{E}_t}{s} \qquad \texttt{(Bias sequences)}$$

$$\mathfrak{I}^{\text{UCBV}}_{a,t-1} := \begin{cases} \infty & \text{if } N_{a,t-1} = 0 \\ \widehat{\mu}_{a,t-1} + \mathfrak{B}_{a,N_{a,t-1},t-1} & \text{otherwise} \end{cases} \tag{2.2.7}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a \, \mathfrak{I}^{\text{UCBV}}_{a,t-1}$.
5: Observe X_t and update $\mathfrak{I}^{\text{UCBV}}_{a,t}$ according to Eq.(2.2.7).
6: **end for**

In term of theoretical guarantees, the regret upper bound of UCB Tuned is given in Theorem 2.5.

Theorem 2.5: UCB V regret upper bound
For the stationary bandit $\Theta = (\mathbf{P}_1, ..., \mathbf{P}_A)$, if $c = 1$ and $\mathcal{E}_t = \zeta \log t$ for $\zeta > 0$ Then, there exists a constant c_ζ that only depends on ζ such that the regret of UCBV strategy after T rounds takes the following upper bound:

$$\mathbb{E}\left[\mathcal{R}^{\text{UCBV}}_T(\Theta)\right] \leqslant c_\zeta \sum_{a:\Delta_a>0} \left(\frac{\sigma_a^2}{\Delta_a} + 2b\right) \log T$$

where: $\sigma_a^2 = \mathbb{E}_{X\sim \mathrm{P}_a}\left[\left(X - \mathbb{E}_{X\sim \mathrm{P}_a}[X]\right)^2\right]$ denotes the variance of the distribution $\mathbf{P}_a$.

Proof. See [12]. □

From Theorem 2.5, the UCB-V strategy is asymptotically order optimal; its cumulative pseudo-regret reaches the distribution dependant lower bound (Theorem 2.2) up to a multiplicative constant.

KL-UCB

The KL-UCB algorithm follows the same spirit as the UCB1 algorithm presented above, it is an index policy (see Algorithm 2.4), which also builds upper confidence bounds on the unknown mean of each arm.

The KL-UCB algorithm originates from the work of T.L. Lai, and was popularized under this name in [103] and later in [31]. The analysis of [31] is provided for Bernoulli and discrete distributions, while [103] considers also one-dimensional exponential families. A specific version called kl-UCB was introduced in [57] for bounded rewards

Instead of using a confidence bound based on Hoeffding's inequality, the kl-UCB algorithm rather uses Chernoff's concentration inequality for the considered exponential family (see equation 2.2.8).

We describe the KL-UCB algorithm in Algorithm 2.4.

Algorithm 2.4 KL-UCB [57]

Require: $f(t)$: ..., T: Horizon, $\mathcal{A}$: Arm set.

1: **Initialization:**
$\forall a \in \mathcal{A} \quad N_a(0) = 0$

2: **Define:**

$$\mathfrak{I}_{a,t-1}^{\text{KL-UCB}} := \begin{cases} \infty & \text{if } N_{a,t-1} = 0 \\ \max\left\{\tilde{\mu} \in [0,1] : \mathbf{KL}\left(\mathcal{B}\left(\widehat{\mu}_{a,t-1}\right), \mathcal{B}\left(\tilde{\mu}\right)\right) \leqslant \frac{\log f(t)}{N_{a,t-1}}\right\} & \text{otherwise} \end{cases} \tag{2.2.8}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a \, \mathfrak{I}_{a,t-1}^{\text{KL-UCB}}$.
5: Observe X_t and update $\mathfrak{I}_{a,t}^{\text{KL-UCB}}$ according to Eq.(2.2.8).
6: **end for**

In term of theoretical guarantees, the regret upper bound of KL-UCB for the Bernoulli bandit (where $\mathbf{P}_a$ the probability distribution of arm $a \in \mathcal{A}$ is a Bernoulli distribution (see Definition 2.6)) is given in Theorem 2.6.

Definition 2.6: Bernoulli distribution
A Bernoulli distribution ($\mathcal{B}(\theta)$) has support on $\{0, 1\}$ with one parameter $\theta \in [0, 1]$ with probability density function:

$$\mathcal{D}_{\mathcal{B}(\theta)}(x) = \theta^x \times (1-\theta)^{1-x}$$

Theorem 2.6: KL-UCB regret upper bound
For the stationary Bernoulli bandit $\Theta = (\mathcal{B}(\theta_1), ..., \mathcal{B}(\theta_A))$ and choosing the function $f(t) = \log t + c \log\log t$ and the parameter $c = 3$, the regret of KL-UCB after T rounds takes the following upper bound:

$$\forall \varepsilon > 0, \exists C_1 > 0, C_2(\varepsilon) > 0, C_3(\varepsilon) > 0 :$$
$$\mathbb{E}\left[\mathcal{R}_T^{\text{KL-UCB}}(\Theta)\right] \leqslant \sum_{a:\Delta_a>0} \Delta_a \left(\frac{(1+\varepsilon)}{\mathbf{KL}\left(\mathcal{B}(\theta_a), \mathcal{B}(\theta_\star)\right)} \log T + C_1 \log(\log T) + \frac{C_2(\varepsilon)}{T^{C_3(\varepsilon)}} \right)$$

Proof. See [31]. □

From Theorem 2.6, the KL-UCB strategy is asymptotically optimal; its cumulative pseudo-regret reaches the distribution dependant lower bound (Theorem 2.2).

Thompson Sampling

Thompson Sampling [81] is a multi-armed bandit strategy which uses the Bayesian approach. It needs a Bayesian prior for each arm as a starting point, which encodes the initial belief that a learner has on each reward distribution associated to each arm. Subsequently, at each round t, it samples from each arm distribution, chooses the action with highest sample and updates its distribution using the Bayes's rule with the received reward. The updated distribution is called the posterior distribution. The Bayes's rule states the following:

$$\mathbb{P}(H \mid D) = \frac{\mathbb{P}(H)\,\mathbb{P}(D \mid H)}{\mathbb{P}(D)}$$

In the formula we can distinguish:

- $\mathbb{P}(H)$ is the prior distribution;
- $\mathbb{P}(D \mid H)$ is the probability of generating data D from hypothesis H also called likelihood;
- $\mathbb{P}(H \mid D)$ is the posterior distribution, that is the prior updated with data D;
- $\mathbb{P}(D)$ is the normalizing constant, which is needed in order to obtain a probability distribution as posterior.

In general, it is not easy to compute the posterior distribution, except for specific choices of the prior distribution and the likelihood function such that the prior and the posterior distributions are from the same family: the prior and the posterior are then called conjugate distributions, and the prior is called conjugate prior for the likelihood function [50]. For example, the beta family is conjugate to itself with respect to a Bernoulli likelihood, and the Gaussian family is conjugate to itself with respect to a Gaussian likelihood.

We consider now the simple case of a MAB with Bernoulli rewards, i.e. arms that provide a reward of value one with probability μ and zero otherwise. As previously stated, if the prior is a Beta distribution and the Bernoulli samples are i.i.d. (i.e., independent and identically distributed), then the posterior distribution is a Beta distribution too. In the following, we recall the formal definition of the Beta distribution.

Definition 2.7: Beta distribution
A Beta distribution ($\text{Beta}(\alpha, \beta)$) has support on $[0, 1]$ with two parameters $\alpha > 0, \beta > 0$ with probability density function:

$$\mathcal{D}_{\text{Beta}(\alpha,\beta)}(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$$

where Γ is the Gamma function. For integers $x \geqslant 1, \Gamma(x) = (x-1)!$.
If x is a random variable distributed from the Beta distribution with parameters (α, β), we have $\mathbb{E}_{x\sim\text{Beta}(\alpha,\beta)}[x] = \frac{\alpha}{\alpha+\beta}$.

The Beta posterior distribution can be computed in the following way, where $X_t \in [0, 1]$ is the new reward received:

$$\begin{aligned}\underbrace{\mathbb{P}(\mu=\theta \mid X_t)}_{\text{posterior}} &\propto \underbrace{\mathbb{P}(X_t \mid \mu=\theta)}_{\text{likelihood}} \underbrace{\mathbb{P}(\mu=\theta)}_{\text{prior}} \\ &= \mathcal{D}_{\mathcal{B}(\theta)}(X_t) \times \mathcal{D}_{\text{Beta}(\alpha,\beta)}(\theta) \\ &= \theta^{X_t}(1-\theta)^{1-X_t}\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha-1}(1-\theta)^{\beta-1} \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\theta^{\alpha+X_t-1}(1-\theta)^{\beta-X_t} \\ &\propto \frac{\Gamma(\alpha+\beta+1)}{\Gamma(\alpha+X_t)\Gamma(\beta+1-X_t)}\theta^{\alpha+X_t-1}(1-\theta)^{\beta-X_t} \\ &= \mathcal{D}_{\text{Beta}(\alpha+X_t,\beta+1-X_t)}(\theta)\end{aligned} \tag{2.2.9}$$

Therefore, starting with a Beta with parameters $\alpha = 1$ and $\beta = 1$, i.e., a uniform distribution over $[0, 1]$, after observing n i.i.d. samples from a Bernoulli distribution, the posterior is a Beta $(\#(\text{reward} = 1) + 1, \#(\text{reward} = 0) + 1)$. Thus, when a new sample arrives, the policy just updates α or β depending on the new sample being a one or a zero. The Thompson Sampling algorithm for Bernoulli ST-MAB with Beta priors is defined in Algorithm 2.5.

Algorithm 2.5 Thompson Sampling with Beta prior [81]

Require: T: Horizon, $s_0, f_0 > 0$: Beta prior hyper-parameters, $\mathcal{A}$: Arm set.

1: **Initialization:**
$\forall a \in \mathcal{A} \quad S_{a,0} = s_0, F_{a,0} = f_0$

2: **Define:**

$$\mathfrak{I}^{\text{TS}}_{a,t-1} \sim \text{Beta}(S_{a,t-1}, F_{a,t-1}) \tag{2.2.10}$$

3: **for** $t = 1, \ldots, T$ **do**

4: Choose action $A_t \leftarrow \text{argmax}_a \mathfrak{I}^{\text{TS}}_{a,t-1}$.

5: Observe X_t and update $S_{a,t} = S_{a,t-1} + X_t, F_{a,t} = F_{a,t-1} + 1 - X_t$.

6: **end for**

In term of theoretical guarantees, the regret upper bound of Thompson Sampling is given in Theorem 2.7.

Theorem 2.7: Thompson Sampling regret upper bound
For the stationary Bernoulli bandit $\Theta = (\mathcal{B}(\theta_1), \ldots, \mathcal{B}(\theta_A))$, the regret of Thompson Sampling after T rounds takes the following upper bound:

$$\forall \varepsilon > 0 \; \exists c_{\varepsilon,\Theta} : \mathbb{E}\left[\mathcal{R}^{\text{TS}}_{T,\varepsilon}(\Theta)\right] \leqslant (1+\varepsilon)\sum_{a:\Delta_a>0}\frac{\Delta_a(\log T + \log\log T)}{\mathbf{KL}(\mathcal{B}(\theta_a), \mathcal{B}(\theta_\star))} + c_{\varepsilon,\Theta}$$

Proof. See [81]. □

From Theorem 2.7, the Thompson Sampling strategy is asymptotically optimal; its cumulative pseudo-regret reaches the distribution dependant lower bound (Theorem 2.2).

Bayes UCB

The Bayes UCB introduced in [79] also uses the Bayesian point of view, updated like for Thompson sampling, but used in a different way. Instead of sampling from the posteriors and playing he arms with maximum sample, Bayes-UCB computes the $(1 - 1/(t(\ln(t))^c)$ quantile of each rm, at time t, and plays the arm with the largest quantile.

Algorithm 2.6 Bayes UCB [79]

Require: T: Horizon, $s_0, f_0 > 0$: Beta prior hyper-parameters, c: parameter of the quantile, $\mathcal{A}$: Arm set.

1: **Initialization:**
$\forall a \in \mathcal{A} \quad S_{a,0} = s_0, F_{a,0} = f_0$

2: **Define:**

$$\mathfrak{I}_{a,t-1}^{\text{BayesUCB}} = \mathbf{Q}\Big(1 - \frac{1}{(t \log t)^c}, \text{Beta}\left(S_{a,t-1}, F_{a,t-1}\right)\Big) \tag{2.2.11}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a \, \mathfrak{I}_{a,t-1}^{\text{BayesUCB}}$.
5: Observe X_t and update $S_{a,t} = S_{a,t-1} + X_t, F_{a,t} = F_{a,t-1} + 1 - X_t$.
6: **end for**

In term of regret upper bound, Bayes UCB was proven to achieve in asymptotical optimal problem-dependent bound for binary (Bernoulli) bandits [79] as we can notice from Theorem 2.8.

Theorem 2.8: Bayes UCB regret upper bound
For the stationary Bernoulli bandit $\Theta = (\mathcal{B}(\theta_1), ..., \mathcal{B}(\theta_A))$ and choosing the parameter of the quantile $c \geqslant 5$, the regret of Bayes UCB after T rounds takes the following upper bound:

$$\forall \varepsilon > 0 : \mathbb{E}\left[\mathcal{R}_T^{\text{BayesUCB}}(\Theta)\right] \leqslant \sum_{a:\Delta_a>0} \frac{(1+\varepsilon)\,\Delta_a}{\mathbf{KL}\left(\mathcal{B}(\theta_a), \mathcal{B}(\theta_\star)\right)} \log T + o_{\varepsilon,c}\Big(\sum_{a:\Delta_a>0} \Delta_a \log T\Big)$$

Proof. See [79]. □

From Theorem 2.8, the Bayes UCB strategy is asymptotically optimal; its cumulative pseudo-regret reaches the distribution dependant lower bound (Theorem 2.2).

2.2.4 Non-stationary stochastic multi-armed bandits

A non stationary multi-armed bandit is a stochastic MAB problem in which the distribution of at least one arm changes before the horizon T. Formally:

Definition 2.8: Non-stationary multi-armed bandit
A non-stationary multi-armed bandit problem is a tuple $< \mathcal{A}, R >$ where:

- $\mathcal{A}$ is the set of possible actions;
- $R := a \in \mathcal{A} \rightarrow R(a, t) \in \mathbb{R}$ is an unknown reward distribution depending on the chosen action a at time t, where $\exists a \in \mathcal{A}, t_1, t_2$ s.t. $R(a, t_1) \neq R(a, t_2)$

Notation 2 (Useful notations for non-stationary bandits)**.** *For the non-stationary bandit, we will need to extend some notations introduced previously so that they are valid in the case of non-stationary distributions.*

- $\mathbf{P}_{a,t}$*: probability distribution of arm* $a \in \mathcal{A}$ *at time* t*.*
- $\mu_{a,t}$*: expectation of the distribution* $\mathbf{P}_{a,t}$ *(Expected reward for arm* a *at time* t*).*
- $\mu_t^\star$*: maximum expectation taken over all arms at time* t *(i.e.* $\mu_t^\star = \max_{a \in \mathcal{A}} \mu_{a,t}$*).*
- $\Delta_{a,t}$*: optimality gap of the* a^{th} *arm at round* t *(i.e.* $\Delta_{a,t} = \mu_t^\star - \mu_{a,t}$*).*

In the case on non-stationary multi-armed bandits, the regret definition becomes the following:

Definition 2.9: Dynamic regret
The dynamic pseudo-regret of a policy π on a non-stationary multi-armed bandit problem is defined as:

$$\mathbb{E}\left[\mathcal{R}_t^\pi\right] = \sum_{s=1}^{t} \mathbb{E}\left(\mu_s^\star - \mu_{A_s}\right)$$

Obviously, the expectation in Definition 2.9 is computed with regard to the stochasticity of the policy π.

In the literature, there are essentially two kinds of strategies for the non-stationary multi-armed bandit: passively adaptive policies and actively adaptive policies.

- *Passively adaptive strategies*

  In order to forget the past rewards, the first passively adaptive strategies propose to penalize the past rewards by multiplying them with a discount factor $\gamma \in (0, 1)$ such that the penalization is of γ^s if the arm was not seen since s time steps. The Discounted UCB (D-UCB) was first proposed by [87] and then it has been analyzed by [58]. Another popular mechanism to forget the past rewards is to use a sliding window of fixed size τ, where only the τ last rewards are used for the decision-maker. There are also other recent algorithms such as Discounted Thompson Sampling [119], Thompson Sampling with sliding window [136] and REXP3 [22] that use passively adaptive mechanisms.

- *Actively adaptive strategies*

  There is a large literature exploring the idea of monitoring the change in the reward distribution via online change-point detection and triggering the reset of the bandit algorithm. This kind of algorithm aims at localizing the change-point and hence

demonstrate better performances than the passive policies. The `Adapte-EvE` algorithm [66] uses the Page-Hinkley test to detect the change-point and hence restart the UCB1 strategy once an alarm is raised. Then, in [105], the authors provide a combination between the Bayesian online change-point detector [2] and Thompson Sampling. A recent and related work [97] uses CUSUM algorithm for change-point detection. Furthermore, the Monitored UCB algorithm (M-UCB) [30] also combines a CUSUM instance with UCB. However, the change-detection test is much easier and a forced exploration phase is also performed. Moreover, in [24] the authors propose a hybrid combination between KL-UCB algorithm and a Bernoulli Generalized Likelihood Ratio Test for change-point detection.

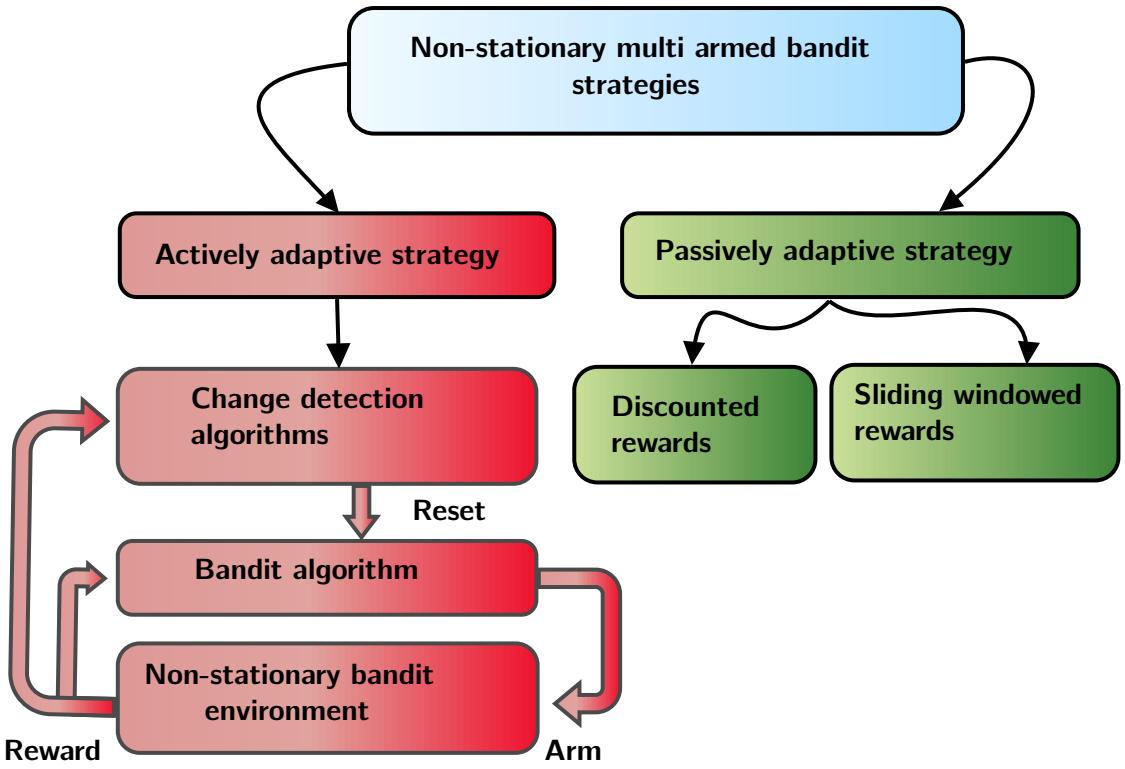


Figure 2.2.1: Taxonomy of non-stationary multi-armed bandits. There are two main families of algorithms for the non-stationary multi-armed bandit. The actively adaptive strategies which are adapted to the abrupt change of environment and the passively adaptive strategies which are more adapted to smooth changes.

Model of non-stationary environments: the switching model

Real decision-making scenarios take place in environments that change over time. We therefore want to design algorithms that assume the environment changes over time.

For the non-stationary bandits, we assume abrupt switching defined by a hazard function, $h(t)$, such that:

$$\mu_{a,t} = \begin{cases} \mu_{a,t-1} & \text{with probability } h(t) \\ \mu_{\text{new}} \in [0,1] & \text{with probability } 1 - h(t) \end{cases} \tag{2.2.12}$$

The algorithms are designed with two such models in mind. The first model is referred to as the Global Switching model. This model switches at a constant rate ($\rho \in [0, 1]$); when a change point happens all arms change their expected rewards. The second model will be referred to as Per-Arm Switching. In this model change points occur independently for each arm, such that the times when arms switch are uncorrelated from each other (see Figure 2.2.2 for a simple illustration of these two models).

In the following, we let Υ_T denoting the overall number of change-points up to time T.

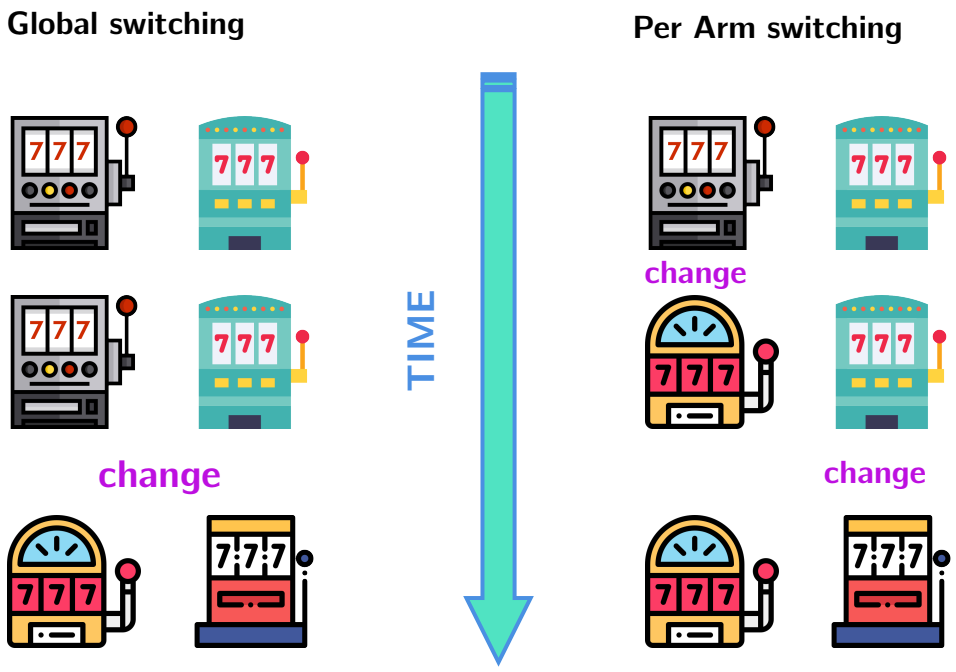


Figure 2.2.2: Global and per-arm switching models. In the global switching model, when a change point happens all arms change their expected rewards. In the per-arm switching model, change points occur independently for each arm, such that when the expected reward switches for one arm, it is uncorrelated to when all other arms switch.

Passively adaptive strategies

Discounted rewards The Discounted UCB (D-UCB) algorithm is an adaptation of the UCB algorithm, first introduced in [87]. It works by decreasing all the past rewards by a discount factor $\gamma \in (0, 1)$, when receiving a new reward from an arm, so that the recent rewards weight more in the (discounted) empirical means, that are used for the computation of its UCB indexes. Indeed, the D-UCB build the index $\mathfrak{I}_{a,t}^{\text{D-UCB}}$ of arm a at time t by exploiting the discounted empirical mean $\widehat{\mu}_{a,t}^{\gamma}$ of arm a at time t. The index $\mathfrak{I}_{a,t}^{\text{D-UCB}}$ takes the following form:

$$\mathfrak{I}_{a,t}^{\text{D-UCB}} = \widehat{\mu}_{a,t}^{\gamma} + \sqrt{\frac{2}{N_{a,t}^{\gamma}} \log\left(\sum_{a \in \mathcal{A}} N_{a,t}^{\gamma}\right)}$$

$$\text{where: } \widehat{\mu}_{a,t}^{\gamma} := \frac{1}{N_{a,t}^{\gamma}} \sum_{i=1}^{s} \gamma^{t-i} \mathbf{1}\{A_i = a\} X_i \quad \text{and: } N_{a,t}^{\gamma} := \sum_{i=1}^{t} \gamma^{t-i} \mathbf{1}\{A_i = a\}$$

We summarize the discounted UCB strategy in Algorithm 2.7.

Algorithm 2.7 Discounted UCB [106]

Require: $\gamma \in (0,1)$: discount factor, T: Horizon, $\mathcal{A}$: Arm set

1: **Initialization:**
$\forall a \in \mathcal{A} \quad N^{\gamma}_{a,0} = 0$

2: **Define:**

$$\mathfrak{I}^{\text{D-UCB}}_{a,t-1} := \begin{cases} \infty & \text{if } N^{\gamma}_{a,t-1} = 0 \\ \widehat{\mu}^{\gamma}_{a,t-1} + \sqrt{\frac{2}{N^{\gamma}_{a,t-1}} \log\left(\sum_{a \in \mathcal{A}} N^{\gamma}_{a,t-1}\right)} & \text{otherwise} \end{cases} \tag{2.2.13}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a \, \mathfrak{I}^{\text{D-UCB}}_{a,t-1}$.
5: Observe X_t and update $\mathfrak{I}^{\text{D-UCB}}_{a,t}$ according to Eq.(2.2.13).
6: **end for**

The regret of D-UCB was proven to be upper-bounded by $\mathcal{O}\left(\sqrt{\Upsilon_T T} \ln(T)\right)$ in [58], with a tuning $\gamma = 1 - \sqrt{\Upsilon_T / T}/4$, dependent on Υ_T.

Sliding windowed rewards The Sliding-Window UCB (SW-UCB), proposed by [58], uses a sliding window of a fixed size τ, to store only the most τ recent rewards for each arm. Indeed, the SW-UCB build the index $\mathfrak{I}^{\text{SW-UCB}}_{a,t}$ of arm a at time t by exploiting the sliding windowed empirical mean $\widehat{\mu}^{\tau}_{a,t}$ of arm a at time t. The index $\mathfrak{I}^{\text{SW-UCB}}_{a,t}$ takes the following form:

$$\mathfrak{I}^{\text{SW-UCB}}_{a,t} = \widehat{\mu}^{\tau}_{a,t} + \sqrt{\frac{2}{N^{\tau}_{a,t}} \log\left(\min\{\tau, t\}\right)}$$

$$\text{where: } \widehat{\mu}^{\tau}_{a,t} := \frac{1}{N^{\tau}_{a,t}} \sum_{i=t-\tau+1}^{t} \mathbf{1}\{A_i = a\} X_i \quad \text{and: } N^{\tau}_{a,t} := \sum_{i=t-\tau+1}^{t} \mathbf{1}\{A_i = a\}$$

We summarize the SW-UCB strategy in Algorithm 2.8.

Algorithm 2.8 Sliding window UCB [106]

Require: $\tau > 0$: sliding window size, T: Horizon, $\mathcal{A}$: Arm set

1: **Initialization:**
$\forall a \in \mathcal{A} \quad N^{\tau}_{a,0} \leftarrow 0$

2: **Define:**

$$\mathfrak{I}^{\text{SW-UCB}}_{a,t-1} := \begin{cases} \infty & \text{if } N^{\tau}_{a,t-1} = 0 \\ \widehat{\mu}^{\tau}_{a,t-1} + \sqrt{\frac{2}{N^{\tau}_{a,t-1}} \log\left(\min\{\tau, t-1\}\right)} & \text{otherwise} \end{cases} \tag{2.2.14}$$

3: **for** $t = 1, \ldots, T$ **do**
4: Choose action $A_t \leftarrow \text{argmax}_a \, \mathfrak{I}^{\text{SW-UCB}}_{a,t-1}$.
5: Observe X_t and update $\mathfrak{I}^{\text{SW-UCB}}_{a,t}$ according to Eq.(2.2.14).
6: **end for**

They prove that tuning its window-size to $\tau = 2\sqrt{T \ln(T)/\Upsilon_T}$, gives a bound on the regret of SW-UCB of the form $\mathcal{O}\left(\sqrt{\Upsilon_T T \ln(T)}\right)$.

Actively adaptive strategies

Adapte-Eve

Adapt-EvE [66] is an actively adaptive policy that uses UCB1-Tuned as sub-algorithm and employs a Page-Hinkley test (PHT) [70] to detect decreases in the mean of the optimal arm. Whenever a change-point is detected, a meta-bandit transient phase starts, whose goal is to choose between two options: reset the sub-algorithm or not. This policy, like other actively adaptive policies, can be generalised with any stationary sub-policy and any change-point detector (CDT) instead of the PHT. However, the PHT procedure provably minimizes the expected time before detection [100].

When the CDT signals an alarm, the algorithm enters a specific transient Exploration vs Exploitation strategy implemented as a meta-bandit. This transient phase is considered as another bandit problem where the two options are:

- restarting the sub-algorithm from scratch;
- discarding the change detection and keeping the same strategy as before.

In the meta-bandit phase, the policy selects at each round the old bandit or the new bandit and gradually decides if the result of the CDT was a false alarm or not. After M time-steps, the meta-bandit phase finishes and the best meta-option becomes the new core algorithm. This process is shown in the flowchart in Figure 4.1.

Adapt-EvE designers tried experimentally to use past samples after an alarm is raised to bootstrap the sub-policy [66]. They call this technique γ restart, which is implemented by decreasing the estimation effort (i.e., $N_{a,t}$) for all the actions after an alarm is triggered:

$$N_{a,t} \to \gamma N_{a,t} \quad \forall a \in \mathcal{A}, \gamma \in [0,1]$$

where γ represents the amount of "memory" of the sub-algorithm to be kept. If the alarm is a false alarm, a big γ would help in preserving inertia and

keep exploiting. On the other hand, a big γ would hinter the exploration if the CDT was right. Experimentally [66], it turns out that $\gamma = 0$ is the optimal setting, i.e. restarting the sub-algorithm from scratch.

There is no proven upper bound on the regret of Adapt-EvE up to now, thus the parameters of the algorithm must be decided with a trial and error procedure.

Monitored UCB

The Monitored UCB (M-UCB) algorithm was introduced in [30]. It uses a specific and simple instance de CUSUM as change-point detector. The CUSUM test is quite complicated and it is parametric: it uses the first w samples (for a fixed $w \in \mathbb{N}^\star$) from one arm to compute the index of arm a at time t denoted as $\mathfrak{I}_{a,t}^{\text{M-UCB}}$, and then builds two random walks, using the remaining observations for this arm. A change is detected when one of the random walks crosses a threshold b. It requires the tuning of three parameters, w and b as well as a drift correction parameter $\gamma \in (0,1)$. We describe the M-UCB strategy in Algorithm 2.9.

Algorithm 2.9 Monitored UCB [30]

Require: $\gamma \in (0,1)$, T: horizon, $\mathcal{A}$: arm set, A: number of arms, $w > 0$: window size, $b > 0$ change-point threshold.

procedure M-UCB($T, \mathcal{A}, A, \gamma, w, b$)
  Initialization:
  $\tau \leftarrow 0, \forall a \in \mathcal{A}\ N_a \leftarrow 0$
  for $t = 1...T$ **do** ▷ Interaction with environment
    $A_t \leftarrow \texttt{ArmSelection}(t, \tau, \mathcal{A}, A, \gamma)$
    Observe X_t and update $N_{A_t} \leftarrow N_{A_t} + 1$, $Z_{A_t,N_{A_t}} \leftarrow X_t$.
    if $\texttt{ChangeDetection}\left(w, b, Z_{A_t,N_{A_t}-w+1}, ..., Z_{A_t,N_{A_t}}\right) = \texttt{True}$ **then**
      $\tau \leftarrow t+1, \forall a \in \mathcal{A}\ N_a \leftarrow 0$
    end if
  end for
end procedure

---

procedure $\texttt{ChangeDetection}(w, b, z_1, ..., z_w)$
  if $\left|\sum_{i=1}^{w/2} z_i - \sum_{i=w/2+1}^{w} z_i\right| \geqslant b$ **then return** True
  else return False
  end if
end procedure

---

procedure $\texttt{ArmSelection}(t, \tau, \mathcal{A}, A, \gamma)$
  $A_t \leftarrow (t-\tau) \bmod \lfloor A/\gamma \rfloor$
  if $A_t > A$ **then**
    $\forall a \in \mathcal{A} \quad \mathfrak{I}_{a,t}^{\text{M-UCB}} := \frac{1}{N_a}\sum_{i=1}^{N_a} Z_{a,i} + \sqrt{\frac{2\log(t-\tau)}{N_a}}$
    $A_t \leftarrow \text{argmax}_{a\in\mathcal{A}}\, \mathfrak{I}_{a,t}^{\text{M-UCB}}$
  end if
  return A_t
end procedure

In term of regret, it has been shown that the Monitored UCB strategy achieves a cumulative pseudo-regret upper bound of the order of $O(\sqrt{\Upsilon_T AT \log T})$ for any fixed window size $w \in \mathbb{N}$, where υ_T denotes the number of change-points observed up to time T.

Switching Thompson Sampling

In what follows, we present a detailed version of the Switching Thompson Sampling algorithm introduced in [105] for the case of a global switch and a per-arm switch. Chapter 3 is dedicated to the detailed analysis of this algorithm.

Let $\mathbf{D}_{t-1} = \{X_1, ..., X_{t-1}\}$ denotes the sequence of observations from the starting time $t = 1$ until $t-1$. In order to extend the Thompson Sampling to the global switching environment setting, it is necessary to correct the sampling from $\mathbb{P}(\theta|\mathbf{D}_{t-1})$ which is the distribution of the arms given the data so far. In a switching setting, the arms model θ depends only the data observed since the last global change occurs. To characterize the occurrence of changes, we introduce the notion of "runlength" which is the overall time steps since the last change-point. We denote the length of the current run at time t by r_t. Since r_t is unknown, we can consider it as a random variable taking values in $\mathbb{N}$ and then

marginalise it out to compute $\mathbb{P}(\theta|\mathbf{D}_{t-1})$ as follow:

$$\mathbb{P}(\theta|\mathbf{D}_{t-1}) = \sum_{r_t} \mathbb{P}(\theta|\mathbf{D}_{t-1}, r_t) \times \mathbb{P}(r_t|\mathbf{D}_{t-1})$$

We denote by $\mathfrak{R}_T = \{r_t\} \subset \mathbb{N}$ the set of all possible runlengths at time t. So, sampling from $\mathbb{P}(\theta|\mathbf{D}_{t-1})$ is done in two steps. We start by sampling the runlength distribution $\mathbb{P}(r_t|\mathbf{D}_{t-1})$, and then given this runlength we sample θ from $\mathbb{P}(\theta|\mathbf{D}_{t-1}, r_t)$. The agent pulls then the arm maximizing the arm model θ.

Computation of $\mathbb{P}(r_t|\mathbf{D}_{t-1})$ Adams and MacKay [2007] have proposed an online recursive runlength estimation in order to calculate the runlength distribution $\mathbb{P}(r_t|\mathbf{D}_{t-1})$. To find

$$\mathbb{P}(r_t|\mathbf{D}_{t-1}) = \frac{\mathbb{P}(r_t, \mathbf{D}_{t-1})}{\mathbb{P}(\mathbf{D}_{t-1})} \tag{2.2.15}$$

we seek the joint distribution over the past estimated runlengths $\mathfrak{R}_{t-1}$:

$$\begin{aligned}
\mathbb{P}(r_t|\mathbf{D}_{t-1}) &= \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t, r_{t-1}, \mathbf{D}_{t-1}) && (2.2.16)\\
&= \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t, X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) && (2.2.17)\\
&= \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t|r_{t-1}) \times \mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) && (2.2.18)
\end{aligned}$$

So, given the previous runlength distribution $\mathbb{P}(r_{t-1}|\mathbf{D}_{t-2})$, one can thus construct a message-passing algorithm for the current runlength distribution $\mathbb{P}(r_t|\mathbf{D}_{t-1})$ by calculating:

1. the predictive distribution $\mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2})$
2. the changepoint prior $\mathbb{P}(r_t|r_{t-1})$

It should be noted that at each time t, the runlength either continues to grow and then $r_t = r_{t-1} + 1$ or a global switch occurs which implies that $r_t = 0$. By this way, the recursive runlength distribution estimation becomes:

- Growth probability:

$$\mathbb{P}(r_t = r_{t-1} + 1|\mathbf{D}_{t-1}) = \mathbb{P}(r_t|r_{t-1}) \times \mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) \tag{2.2.19}$$

- Changepoint probability:

$$\mathbb{P}(r_t = 0|\mathbf{D}_{t-1}) = \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t|r_{t-1}) \times \mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) \tag{2.2.20}$$

Changepoint Prior According to Eq.(4.3.4) and Eq.(4.3.5), the runlength distribution estimation need to compute the changepoint prior $\mathbb{P}(r_t|r_{t-1})$, which is done following 9.2.3:

$$\mathbb{P}(r_t|r_{t-1}) = \begin{cases} \rho & if\ r_t = 0 \\ 1 - \rho & if\ r_t = r_{t-1} + 1 \end{cases} \tag{2.2.21}$$

Then Eq.(4.3.4) and Eq.(4.3.5) become:

$$\mathbb{P}(r_t = r_{t-1} + 1|\mathbf{D}_{t-1}) = (1-\rho) \times \mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) \tag{2.2.22}$$

$$\mathbb{P}(r_t = 0|\mathbf{D}_{t-1}) = \sum_{r_t \in \mathfrak{R}_{t-1}} \rho \times \mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) \tag{2.2.23}$$

Predictive Distribution $\mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2})$ The predictive distribution $\mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2})$ is built on exponential family likelihoods. The inference is established by a finite number of sufficient statistics which are updated incrementally as new datum X_t arrives. In the case of Bernoulli rewards, and denoting $\mu_{A_t}(t)$ the expectation of the arm pulled at time t, and X_t the associated reward, one can easily verify that:

$$\mathbb{P}(X_t|\mu_{A_t}(t)) = \exp\left(X_t \log \mu_{A_t}(t) + (1 - X_t) \log\left(1 - \mu_{A_t}(t)\right)\right) \tag{2.2.24}$$

which is an exponential likelihood of natural parameter $\eta_{A_t}(t) = \langle \log \mu_{A_t}(t), \log\left(1 - \mu_{A_t}(t)\right)\rangle$ and sufficient statistics $t(X_t) = \langle X_t, 1 - X_t\rangle$.

If confusion isn't caused, we will write:

$$\mathbb{P}(X_t|\eta_{A_t}(t)) = \mathbb{P}(X_t|\mu_{A_t}(t)) = \exp\left(\eta_{A_t}(t) \times t(X_t)^\top\right) \tag{2.2.25}$$

Conjugate-exponential families allow us to write both the prior and the posterior as an exponential family distribution over the natural parameter $\eta_{A_t}(t)$ which is controlled by succinct hyperparameters $\mathscr{S}_{A_t}(t)$ and $\mathscr{F}_{A_t}(t)$.
In the case of Bernoulli reward, the posterior distribution of arm A_t is a Beta distribution of parameters $\mathscr{S}_{A_t}(t)$ and $\mathscr{F}_{A_t}(t)$ whose likelihood is written as:

$$\mathbb{P}\left(\eta_{A_t}(t) \mid \mathscr{S}_{A_t}(t), \mathscr{F}_{A_t}(t)\right) = \frac{\exp\left((\mathscr{S}_{A_t}(t) - 1) \log \mu_{A_t}(t) + (\mathscr{F}_{A_t}(t) - 1) \log\left(1 - \mu_{A_t}(t)\right)\right)}{\text{Beta}\left(\mathscr{S}_{A_t}(t), \mathscr{F}_{A_t}(t)\right)} \tag{2.2.26}$$

which is obviously an exponential likelihood of natural parameter $\chi_{A_t}(t) = \langle \mathscr{S}_{A_t}(t) - 1, \mathscr{F}_{A_t}(t) - 1\rangle$ and sufficient statistics $\eta_{A_t}(t)$. It should be noted that $\text{Beta}\left(\mathscr{S}_{A_t}(t), \mathscr{F}_{A_t}(t)\right)$ denotes the evaluation of Euler Beta function at the points $\mathscr{S}_{A_t}(t)$, $\mathscr{F}_{A_t}(t)$.

Because of the global switching behavior of the environment, the hyperparameters $\mathscr{S}_{A_t}$ and $\mathscr{F}_{A_t}$ and the natural parameter $\eta_{A_t}(t)$ depend only on the previous runlength r_{t-1} instead of the overall interaction t. So, the posterior distribution of arm A_t becomes:

$$\mathbb{P}\left(\eta_{A_t}(r_{t-1}) \mid \mathscr{S}_{A_t}(r_{t-1}), \mathscr{F}_{A_t}(r_{t-1})\right) = \frac{\exp\left(\chi_{A_t}(r_{t-1}) \times \eta_{A_t}(r_{t-1})\right)}{\text{Beta}\left(\mathscr{S}_{A_t}(r_{t-1}), \mathscr{F}_{A_t}(r_{t-1})\right))} \tag{2.2.27}$$

Knowing the previous runlength r_{t-1}, we can easily get $\mathscr{S}_{A_t}(r_{t-1})$ and $\mathscr{F}_{A_t}(r_{t-1})$. Without causing confusion, we write:

$$\mathbb{P}\left(\eta_{A_t}(r_{t-1})|r_{t-1}\right) = \mathbb{P}\left(\eta_{A_t}(r_{t-1})|\mathscr{S}_{A_t}(r_{t-1}), \mathscr{F}_{A_t}(r_{t-1})\right) \tag{2.2.28}$$

Then, the predictive distribution $\mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2})$ naturally comes by marginalizing out the natural parameter $\eta_{A_t}(r_{t-1})$ following 2.2.25 and 2.2.28:

$$\mathbb{P}(X_t|r_{t-1}, \mathbf{D}_{t-2}) = \int \mathbb{P}(X_t|\eta_{A_t}(r_{t-1})) \times \mathbb{P}\left(\eta_{A_t}(r_{t-1})|r_{t-1}\right) d\eta_{A_t}(r_{t-1}) \tag{2.2.29}$$

Finally, to establish the desired inference we follow the hyperparameters updating rule inspired by the classical Thompson Sampling. Indeed, the update rules are written as follows:

$$\begin{cases} \mathscr{S}_{A_t}(r_t = r_{t-1} + 1) & = \mathscr{S}_{A_t}(r_{t-1}) + \mathbb{I}(X_t = 1). \\ \mathscr{F}_{A_t}(r_t = r_{t-1} + 1) & = \mathscr{F}_{A_t}(r_{t-1}) + \mathbb{I}(X_t = 0). \\ \mathscr{S}_{A_t}(r_t = 0) & = s_0. \\ \mathscr{F}_{A_t}(r_t = 0) & = f_0. \end{cases}$$

We describe the detailed operation of the switching Thompson Sampling algorithm in the case of a global switching setting in Algorithm 2.10 (We use the notation $\mathfrak{p}_r^t = \mathbb{P}(r_t = r|\mathbf{D}_{t-1})$ to denote the runlength distribution at time t).

Algorithm 2.10 Global Switching Thompson sampling

Require: $\rho \in (0,1)$: switching rate, $s_0, f_0 > 0$: Beta prior hyper-parameters, T: horizon, $\mathcal{A}$: arm set.

procedure GLOBAL.S-TS($\rho, s_0, f_0, T, \mathcal{A}$)
  $t \leftarrow 1$ ▷ Initialize interaction
  $\{\mathfrak{p}\}^t \leftarrow \{\}$, $\mathfrak{p}_0^t \leftarrow 1$, $\{\mathfrak{p}\}^t \leftarrow \{\mathfrak{p}\}^t \cup \mathfrak{p}_0^t$ ▷ Initialize runlength distribution
  $\forall a \in \mathcal{A}\ \mathscr{S}_{0,a}^t \leftarrow s_0, \mathscr{F}_{0,a}^t \leftarrow f_0$ ▷ Initialize hyperparameters
  for $t \leqslant T$ **do** ▷ Interaction with environment
    $A_t \leftarrow \texttt{ArmSelection}(\{\mathfrak{p}\}^t, \{\mathscr{S}\}^t, \{\mathscr{F}\}^t)$
    $X_t \leftarrow \texttt{PlayArm}(A_t)$
    $\{\mathfrak{p}\}^{t+1} \leftarrow \texttt{RunlengthDistributionUpdate}(\{\mathfrak{p}\}^t, \{\mathscr{S}\}_{A_t}^t, \{\mathscr{F}\}_{A_t}^t, X_t, \rho)$
    $\{\mathscr{S}\}^{t+1}, \{\mathscr{F}\}^{t+1} \leftarrow \texttt{ArmsModelUpdate}(\{\mathscr{S}\}_{A_t}^t, \{\mathscr{F}\}_{A_t}^t, X_t)$
  end for
end procedure

---

procedure ArmSelection($\{\mathfrak{p}\}^t, \{\mathscr{S}\}^t, \{F\}^t$)
  Draw r with probability $\mathfrak{p}_r^t$
  $\forall a \in \mathcal{A} \quad \theta_a \sim \text{Beta}\left(\mathscr{S}_{r,a}^t, \mathscr{F}_{r,a}^t\right)$
  return $\text{argmax}_a\, \theta_a$
end procedure

---

procedure RunlengthDistributionUpdate($\{\mathfrak{p}\}^t, \{\mathscr{S}\}_{A_t}^t, \{F\}_{A_t}^t, X_t, \rho$)
  define:

$$\text{likelihood}_r^t := \begin{cases} \frac{\mathscr{S}_{r,A_t}^t}{\mathscr{S}_{r,A_t}^t + \mathscr{F}_{r,A_t}^t} & \text{if } X_t = 1 \\ \frac{\mathscr{F}_{r,A_t}^t}{\mathscr{S}_{r,A_t}^t + \mathscr{F}_{r,A_t}^t} & \text{otherwise} \end{cases}$$

  $\mathfrak{p}_{r+1}^{t+1} \leftarrow (1-\rho) \times \text{likelihood}_r^t \times \mathfrak{p}_r^t$
  $\mathfrak{p}_0^{t+1} \leftarrow \sum_r \rho \times \text{likelihood}_r^t \times \mathfrak{p}_r^t$
  Normalise $\{\mathfrak{p}\}^{t+1}$
  return $\{\mathfrak{p}\}^{t+1}$
end procedure

---

procedure ArmsModelUpdate($\{\mathscr{S}\}_{A_t}^t, \{\mathscr{F}\}_{A_t}^t, X_t$)
  $\mathscr{S}_{r+1,A_t}^{t+1} \leftarrow \mathscr{S}_{r,A_t}^t + X_t$
  $\mathscr{F}_{r+1,A_t}^{t+1} \leftarrow \mathscr{F}_{r,A_t}^t + 1 - X_t$
  $\forall a \in \mathcal{A}, \mathscr{S}_{0,a}^{t+1} \leftarrow s_0, \mathscr{F}_{0,a}^{t+1} \leftarrow f_0$
  return $\{\mathscr{S}\}^t, \{\mathscr{F}\}^t$
end procedure

Extension to the per-arm switching setting For the case of per-arm switching setting, the switching Thompson sampling algorithm can easily be extended in this case by considering a runlength distribution $\{\mathfrak{p}\}_a^t$ for each arm $a \in \mathcal{A}$ at time t.

Thus, in the per-arm switching setting, at a time-step t we update the runlength distribution model associated with the arm A_t that was pulled at t via the update equations Eq.(2.2.22) and Eq.(2.2.23).

The runlength distribution models associated with arms not pulled at t are updated differently

since the runlength for these arms is independent of the reward X_t received at time t for the arm actually pulled A_t. The reward likelihood term disappears in the update equations for the runlength distribution of unpulled arms. Indeed, the update rule in this case takes the following form:

$$\begin{aligned}
\mathbb{P}(r_t|\mathbf{D}_{t-1}) &= \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t, r_{t-1}, \mathbf{D}_{t-1}) \\
&= \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t, X_t|r_{t-1}, \mathbf{D}_{t-2}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) \\
&= \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t|r_{t-1}) \times \mathbb{P}(X_t) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2}) \\
&\propto \sum_{r_t \in \mathfrak{R}_{t-1}} \mathbb{P}(r_t|r_{t-1}) \times \mathbb{P}(r_{t-1}|\mathbf{D}_{t-2})
\end{aligned}$$

We describe the switching Thompson sampling for the per-arm switching setting in Algorithm 2.11.

Algorithm 2.11 Per Arm switching Thompson sampling

Require: $\forall a \in \mathcal{A}\ \rho_a \in (0,1)$: switching rate for each arm a, $s_0, f_0 > 0$: Beta prior hyperparameters, T: horizon, $\mathcal{A}$: arm set.

procedure PER ARM.S-TS($\rho, s_0, f_0, T, \mathcal{A}$)
  $t \leftarrow 1$ ▷ Initialize interaction
  $\forall a \in \mathcal{A}\ \{\mathfrak{p}\}_a^t \leftarrow \{\}$, $\mathfrak{p}_{0,a}^t \leftarrow 1$, $\{\mathfrak{p}\}_a^t \leftarrow \{\mathfrak{p}\}_a^t \cup \mathfrak{p}_{0,a}^t$ ▷ Initialize runlength distribution for each arm a.
  $\forall a \in \mathcal{A}, \mathscr{S}_{0,a}^t \leftarrow s_0, \mathscr{F}_{0,a}^t \leftarrow f_0$ ▷ Initialize hyperparameters
  for $t \leqslant T$ **do** ▷ Interaction with environment
    $A_t \leftarrow \texttt{ArmSelection}(\{\mathfrak{p}\}^t, \{\mathscr{S}\}^t, \{\mathscr{F}\}^t)$
    $X_t \leftarrow \texttt{PlayArm}(A_t)$
    $\{\mathfrak{p}\}^{t+1} \leftarrow \texttt{RunlengthDistributionUpdate}(\{\mathfrak{p}\}^t, \{\mathscr{S}\}_{A_t}^t, \{\mathscr{F}\}_{A_t}^t, X_t, \rho)$
    $\{\mathscr{S}\}^{t+1}, \{\mathscr{F}\}^{t+1} \leftarrow \texttt{ArmsModelUpdate}(\{\mathscr{S}\}_{A_t}^t, \{\mathscr{F}\}_{A_t}^t, X_t)$
    $t \leftarrow t+1$
  end for
end procedure

---

procedure $\texttt{ArmSelection}(\{\mathfrak{p}\}^t, \{\mathscr{S}\}^t, \{F\}^t)$
  $\forall a \in \mathcal{A}$ draw r_a with probability $\mathfrak{p}_{r,a}^t$. Then draw $\theta_a \sim \text{Beta}\left(\mathscr{S}_{r_a,a}^t, \mathscr{F}_{r_a,a}^t\right)$
  return $\text{argmax}_a\, \theta_a$
end procedure

---

procedure $\texttt{RunlengthDistributionUpdate}(\{\mathfrak{p}\}^t, \{\mathscr{S}\}_{A_t}^t, \{F\}_{A_t}^t, X_t, \rho)$
  define:

$$\text{likelihood}_r^t := \begin{cases} \frac{\mathscr{S}_{r,A_t}^t}{\mathscr{S}_{r,A_t}^t + \mathscr{F}_{r,A_t}^t} & \text{if } X_t = 1 \\ \frac{\mathscr{F}_{r,A_t}^t}{\mathscr{S}_{r,A_t}^t + \mathscr{F}_{r,A_t}^t} & \text{otherwise} \end{cases}$$

  $\mathfrak{p}_{r+1,A_t}^{t+1} \leftarrow (1-\rho) \times \text{likelihood}_r^t \times \mathfrak{p}_{r,A_t}^t$ ▷ Increase size of runlength
  $\mathfrak{p}_{0,A_t}^{t+1} \leftarrow \sum_r \rho \times \text{likelihood}_r^t \times \mathfrak{p}_{r,A_t}^t$ ▷ Set runlength size to zero
  $\mathfrak{p}_{r+1,a}^{t+1} \leftarrow (1-\rho) \times \mathfrak{p}_{r,a}^t \quad \forall a \neq A_t$ ▷ Increase size of runlength
  $\mathfrak{p}_{0,A_t}^{t+1} \leftarrow \sum_r \rho \times \mathfrak{p}_{r,a}^t \quad \forall a \neq A_t$ ▷ Set runlength size to zero
  $\forall a \in \mathcal{A} \quad \texttt{Normalise}\,\{\mathfrak{p}\}_a^{t+1}$
  return $\{\mathfrak{p}\}^{t+1}$
end procedure

---

procedure $\texttt{ArmsModelUpdate}(\{\mathscr{S}\}_{A_t}^t, \{\mathscr{F}\}_{A_t}^t, X_t)$
  $\mathscr{S}_{r+1,A_t}^{t+1} \leftarrow \mathscr{S}_{r,A_t}^t + X_t$
  $\mathscr{F}_{r+1,A_t}^{t+1} \leftarrow \mathscr{F}_{r,A_t}^t + 1 - X_t$
  $\forall a \in \mathcal{A}, \mathscr{S}_{0,a}^{t+1} \leftarrow s_0, \mathscr{F}_{0,a}^{t+1} \leftarrow f_0$
  return $\{\mathscr{S}\}^t, \{\mathscr{F}\}^t$
end procedure

2.2.5 Adversarial multi-armed bandits

Another important family of bandits algorithms are the algorithms proposed for the adversarial setting [16].

Definition 2.10: Adversarial bandits
An adversarial MAB problem is specified by the number of possible actions A and a reward vector $\mathbf{X}_t = (X_{1,t}, \ldots, X_{A,t})$ for each round t, where $X_{a,t} \in [0,1]$.

In an Adversarial multi-armed bandit, the rewards are generated by a process that cannot be considered as stochastic. For instance, an adversary may generate arbitrary rewards that make the agent's policy achieve as much regret as possible. As a consequence, deterministic algorithms cannot be used (e.g., UCB1), because the adversary would always know the chosen arm and would assign it a low reward. The Adversarial multi-armed bandit problem can be formalised as a game between a player choosing an action and a player choosing the rewards.

The first algorithm that has been designed for adversarial bandits is the Exp3 strategy, which stands for "Exponential weights for Exploitation and Exploration".

Exp3: Exponential weights for Exploration and Exploitation

The Exp3 policy maintains weights $\varpi_a(t)$ on the A arms, and at time t it samples an arm from the distribution which is a mixture of $[\varpi_1(t), ..., \varpi_A(t)]$ and the uniform distribution, i.e. $\mathbb{P}(A_t = a) = (1-\gamma)\frac{\varpi_a(1)}{\sum_{a\in\mathcal{A}}\varpi_a(t)} + \gamma \times \frac{1}{A}$.

The weights are initially uniform and then at each time after observing a reward X_t from arm A_t, Exp3 updates the weight of the chosen arm multiplicatively, using $\varpi_a(t+1) = \varpi_a(t)\exp\left(\gamma\tilde{Y}_{a,t}/A\right)$. The quantity $\tilde{Y}_{a,t}$ denotes an unbiased estimation of the reward X_t which is $X_t/\mathbb{P}(A_t = a)$.

We summarize the Exp3 strategy in Algorithm 2.12.

Algorithm 2.12 Exp3 [16]

Require: $\gamma \in (0,1)$: exploration rate, T: horizon, A: number of arms, $\mathcal{A}$: arm set.

Initialization:
  $\forall a \in \mathcal{A}, \varpi_a(1) = 1$
for $t = 1, \ldots, T$ **do**
  $\xi_a(t) = (1-\gamma)\frac{\varpi_a(1)}{\sum_{a\in\mathcal{A}}\varpi_a(t)} + \gamma \times \frac{1}{A}$
  Play arm A_t such that: $\forall a \in \mathcal{A} \quad \mathbb{P}(A_t = a) = \xi_a(t)$.
  Observe $X_t \in [0,1]$.
  Define:

$$\tilde{Y}_{a,t} := \begin{cases} \frac{X_t}{\xi_{A_t}(t)} & \text{if } a = A_t \\ 0 & \text{otherwise} \end{cases}$$

  Update arm weight: $\varpi_a(t+1) = \varpi_a(t)\exp\left(\gamma\tilde{Y}_{a,t}/A\right)$
end for

It is proven in [16] that the Exp3 strategy with a constant parameter γ achieves $\mathcal{R}_T^{\text{Exp3}} = \mathcal{O}(\sqrt{AT\ln(A)})$ problem-independent regret, and using a decreasing sequence, such as $\gamma_t = \mathcal{O}(1/\sqrt{t})$ gives an order-optimal regret upper-bound of the same order $\mathcal{O}(\sqrt{AT\ln(A)})$.

Then, we have two more algorithms for adversarial bandits namely Exp3.P strategy and Exp3.S policy.

Exp3.P

Exp3.P [16] is a variant of the Exp3 strategy where the variance is well controlled by using estimations based on upper-confidence bounds instead of estimations with the correct expectation. This modification allows us to get guarantees in high probability.

We summarize the Exp3.P strategy in Algorithm 2.13.

Algorithm 2.13 Exp3.P [16]

Require: $\gamma \in (0,1)$: exploration rate, $\alpha > 0$, T: horizon, A: number of arms, $\mathcal{A}$: arm set.

Initialization:

$\forall a \in \mathcal{A}, \varpi_a(1) = \exp\left(\frac{\alpha\gamma}{3}\sqrt{\frac{T}{A}}\right)$

for $t = 1, \dots, T$ **do**

$\xi_a(t) = (1-\gamma)\frac{\varpi_a(1)}{\sum_{a\in\mathcal{A}}\varpi_a(t)} + \gamma \times \frac{1}{A}$

Play arm A_t such that: $\forall a \in \mathcal{A} \quad \mathbb{P}(A_t = a) = \xi_a(t)$.

Observe $X_t \in [0,1]$.

Define:

$$\tilde{Y}_{a,t} := \begin{cases} \frac{X_{A_t}}{\xi_{A_t}(t)} & \text{if } a = A_t \\ 0 & \text{otherwise} \end{cases}$$

Update arm weight: $\varpi_a(t+1) = \varpi_a(t)\exp\left(\frac{\gamma}{3A}\left(\tilde{Y}_{a,t} + \frac{\alpha}{\xi_a(t)\sqrt{KT}}\right)\right)$

end for

Exp3.S

The Exp3.S strategy [16] is a variant of Exp3 where its regret depends on the *hardness* of a sequence of actions $(A_1, \dots, A_T)$ defined as:

$$\mathrm{H}(A_1, \dots, A_T) \overset{\text{def}}{=} 1 + |\{1 \leqslant \ell < T : A_\ell \neq A_{\ell+1}\}| \tag{2.2.30}$$

Algorithm 2.14 Exp3.S [16]

Require: $\gamma \in (0,1)$: exploration rate, T: horizon, A: number of arms, $\mathcal{A}$: arm set.

Initialization:

$\forall a \in \mathcal{A}, \varpi_a(1) = 1$ ▷ Initial weight given to each arm a

for $t = 1, \dots, T$ **do**

$\xi_a(t) = (1-\gamma)\frac{\varpi_a(1)}{\sum_{a\in\mathcal{A}}\varpi_a(t)} + \gamma \times \frac{1}{A}$

Play arm A_t such that: $\forall a \in \mathcal{A} \quad \mathbb{P}(A_t = a) = \xi_a(t)$.

Observe $X_t \in [0,1]$.

Define:

$$\tilde{Y}_{a,t} := \begin{cases} \frac{X_t}{\xi_{A_t}(t)} & \text{if } a = A_t \\ 0 & \text{otherwise} \end{cases}$$

Update arm weight: $\varpi_a(t+1) = \varpi_a(t)\exp\left(\gamma\tilde{Y}_{a,t}/A\right) + \frac{e\alpha}{A}\sum_{a=1}^{A}\varpi_a(t)$

end for

It is proven in [16] that Exp3.S strategy achieves $\mathcal{R}_T^{\text{Exp3.S}} = \mathcal{O}(\sqrt{SAT\ln(AT)})$ for any sequence of actions $(A_1, \dots, A_T)$ where $\mathrm{H}(A_1, \dots, A_T) \leqslant S$.

Part II

Contributions in the multi-armed bandit and statistical learning theory communities

Chapter 3

Memory Bandits for the piece-wise stationary stochastic multi-armed bandit problem

Overview

The Thompson Sampling exhibits excellent results in practice and it has been shown to be asymptotically optimal. The extension of Thompson Sampling algorithm to the Switching Multi-Armed Bandit problem, proposed in [105], is a Thompson Sampling equiped with a Bayesian online change point detector [2]. In this chapter, we propose another extension of this approach based on a Bayesian aggregation framework. Experiments provide some evidences that in practice, the proposed algorithm compares favorably with the previous version of Thompson Sampling for the Switching Multi-Armed Bandit Problem, while it outperforms clearly other algorithms of the state-of-the-art.

Publication. This chapter is mainly based on our article [4].

3.1 Introduction

We consider the non-stationary multi-armed bandit problem with a set $\mathcal{A}$ of A independent arms. At each round t, the agent chooses an action $A_t \in \mathcal{A}$ and observes a reward x_{A_t}. In a stationary environment, the agent has to explore to find the best arm and to exploit it to maximize his gain. Efficient algorithms [14, 81] have been proposed for handling the so-called exploration-exploitation dilemma. In an evolving environment, the best arm can change during time and hence the agent has to explore more. The adversarial bandits handles non-stationary environments by considering that a sequence of deterministic rewards is chosen in advance by an oblivious adversary. Rate optimal algorithms have been proposed to find the best arm or the best sequence of arms of the run in [16, 109, 7]. Another approach for handling non-stationary environment is to consider that the rewards are generated by an unknown stochastic process that evolves during time. In the switching bandit problem [66, 58, 6], the mean rewards of arms change abruptly. In comparison to the adversarial approach, the advantage of non-stationary stochastic approach is that with some mild assumptions, stochastic algorithms, which are more efficient in practice than adversarial algorithms, can be used.

For practical and theoretical reasons, the recent years have seen an increasingly interest for the oldest bandit algorithm, the Thompson Sampling [135]. It exhibits excellent results

in practice [36], while it is asymptotically optimal [81]. In [105], the authors propose an adaptation of the Thompson Sampling to the switching bandit problem. To be consistent with the Bayesian approach, rather than using a frequentist drift detector used in [66, 6], the authors use a Bayesian online change point detection [2] combined with the Thompson Sampling algorithm. In this paper, we propose a similar approach of [105] which is based on a Bayesian aggregation of a growing number of experts seen as learners. Our approach compares favorably with the one of [105].

3.2 Problem formulation

Let us consider an agent facing a non-stationary multi-armed bandit problem with a set $\mathcal{A} = \{1, ..., A\}$ of A independent arms. At each round $t \in [\![1, T]\!]$, the agent chooses to observe one of the A possible actions. When playing the arm A_t at time t, a reward x_{A_t} is received, where $x_{A_t} \sim \mathcal{B}(\mu_{A_t,t})$ is a random variable drawn from a Bernoulli distribution of expectation $\mu_{a,t}$. Let $\mu_t^\star = \max_{a \in \mathcal{A}} \{\mu_{a,t}\}$ denotes the best expected reward at round t, $A_t^\star = \text{argmax}_{a \in \mathcal{A}} \{\mu_{a,t}\}$ the best arm at round t, k_t the action chosen by the decision-maker at time t and x_{A_t} the reward obtained at the same time.

Changes in the Bernoulli distributions expectations It should be noted that $\mu_{a,t}$, i.e. the reward mean of arm a at time t changes over time according to a global abrupt switching model parametrized with an unknown hazard function $h(t) \in [0, 1]$ assumed to be a constant $h(t) = \rho$ such that:

$$\mu_{a,t} = \begin{cases} \mu_{a,t-1} & \text{with probability } \ 1 - \rho \\ \mu_{new} \sim \mathcal{U}(0, 1) & \text{with probability } \rho \end{cases} \tag{3.2.1}$$

When the behavior's environment is modeled by equation (3.3.3) for all $k \in \mathcal{K}$, the problem setting is called a *Global Switching Multi-Armed Bandit* (GS-MAB), i.e. when a switch happens *all* arms change their expected rewards. Where changes occur independently for each arm k (i.e. arms change points are independent from an arm to another), the problem setting is called a *Per-arm Switching Multi-Armed Bandit*. For the sake of clarity, in the following we will focus on GS-MAB.

Sequence of change points It should be noted that for each GS-MAB, it exists a non-decreasing change points sequence of length Υ_T denoted by $(\tau_\kappa)_{\kappa \in [\![1, \Upsilon_T + 1]\!]} \in \mathbb{N}^{\Upsilon_T + 1}$ where:

$$\begin{cases} \forall\ \kappa \in [\![1, \Upsilon_T]\!],\ \forall\ t \in \mathcal{T}_\kappa = [\![\tau_\kappa + 1, \tau_{\kappa+1}]\!],\ \forall\ a \in \mathcal{A},\ \ \mu_{a,t} = \mu_{a,[\kappa]} \\ \tau_1 = 1 < \tau_2 < ... < \tau_{\Upsilon_T + 1} = T \end{cases}$$

In this case, $\mu_{[\kappa]}^\star = \max_a \left\{\mu_{a,[\kappa]}\right\}$ denotes the highest expected reward at epoch $\mathcal{T}_\kappa$.

Pseudo Cumulative Regret for the switching environment In a switching environment, the pseudo cumulative regret $\mathcal{R}(T)$ up to time T is defined as the expected difference between the rewards obtained by our policy and those received by the *oracle* which always plays the best arm $k_{[\kappa]}^\star$ at each epoch $\mathcal{T}_\kappa$ such as:

$$\mathcal{R}(T) = \sum_{t=1}^{T} \mu_t^\star - \mathbb{E}\left[\sum_{t=1}^{T} x_{A_t}\right] = \sum_{\kappa=1}^{\Upsilon_T} \left(|\mathcal{T}_\kappa|\, \mu_{[\kappa]}^\star - \mathbb{E}\left[\sum_{t=\tau_\kappa+1}^{\tau_{\kappa+1}} x_{A_t}\right]\right)$$

3.3 Global Switching Thompson Sampling with Bayesian Aggregation

3.3.1 The Thompson Sampling algorithm

Unlike optimistic algorithms belonging to the UCB family [14], which are often based on confidence intervals, the Thompson Sampling deals with Bayesian tools by assuming a Beta prior distribution $\pi_{a,t=1} = \text{Beta}(s_0, f_0)$ on each arm for some $s_0, f_0 > 0$. Based on the rewards observed x_t, the posterior distribution $\pi_{a,t}$ is updated such as: $\pi_{a,t} = \text{Beta}\left(S_{a,t} = \sum_{s=1}^{t} X_s \times \mathbf{1}\{A_s = a\} + s_0, F_{a,t} = \sum_{s=1}^{t} (1 - X_s) \times \mathbf{1}\{A_s = a\} + f_0\right)$. At each time, the agent takes a sample $\theta_{a,t}$ from each $\pi_{a,t}$ and then plays the arm $A_t = \text{argmax}_a \theta_{a,t}$. Formally, by denoting $\mathbf{D}_{t-1} = \bigcup_{i=1}^{t-1} x_i$ the history of past rewards we write: $\theta_t = (\theta_{1,t}, ..., \theta_{a,t}) \sim \mathbb{P}(\theta_t | \mathbf{D}_{t-1}) = \prod_{a=1}^{K} \pi_{a,t}$. Recently, the Thompson Sampling has been shown to be asymptotically optimal [81], i.e. the expectation of the pseudo-cumulative regret reaches the Lai and Robbins lower bound on regret in the stochastic Bernoulli multi-armed bandit setting [92].

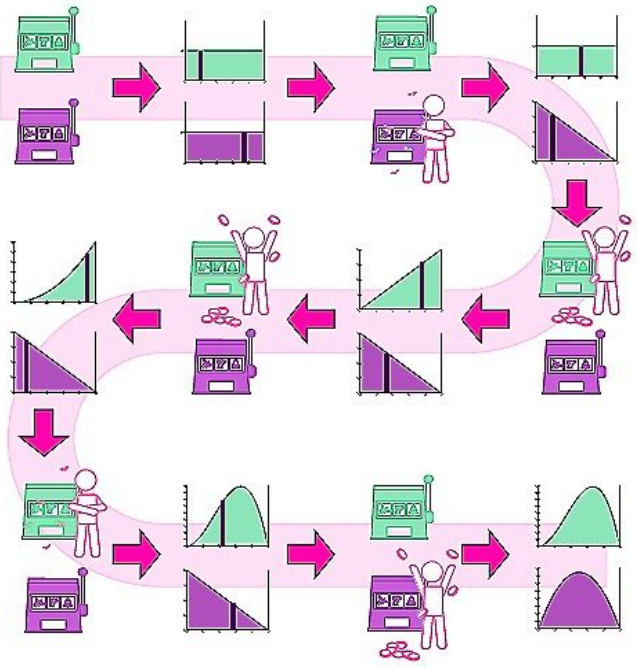


Figure 3.3.1: Evolution of the posterior Beta distributions for a two armed Bernoulli bandit problem.

The agent starts with no knowledge of the arm means and so the distributions are uniform (Beta(1, 1)). As more observations are made the distributions become more concentrated so that it becomes more likely that a sample estimate is drawn close to the true mean of a given arm.

3.3.2 Decision making based on Bayesian aggregation

Best achievable performance: the Thompson Sampling oracle Let TS* denotes the oracle that knows exactly the change points τ_κ. It simply restarts a Thompson Sampling at these change points. Assume that Υ_T is the overall number of change points observed

until T, then TS* runs successively Υ_T Thompson Sampling processes starting at $\tau_\kappa + 1$ and ending at $\tau_{\kappa+1}$.

Notion of expert Let $t \in \mathbb{N}^\star$ and $i \in [1, t]$. An expert $\mathfrak{E}_{i,t}$ is a Thompson Sampling procedure which has started at time i. The expert $\mathfrak{E}_{i,t}$ observes exactly $t - i$ rewards from the environment. Formally, the expert $\mathfrak{E}_{i,t}$ is written as follows:

$$\mathfrak{E}_{i,t} = \Big\{ \text{Beta}\left(S_{1,i,t}, F_{1,i,t}\right), \text{Beta}\left(S_{2,i,t}, F_{2,i,t}\right), ..., \text{Beta}\left(S_{a,i,t}, F_{a,i,t}\right) \Big\}$$

where: $\forall a \in \mathcal{A}, S_{a,i,t} = \sum_{s=i}^{t} X_s \times \mathbf{1}\{A_s = a\} + s_0$ and $F_{a,i,t} = \sum_{s=i}^{t} (1 - X_s) \times \mathbf{1}\{A_s = a\} + f_0$.

Bayesian Aggregation Like [105], to characterize the occurrence of changes, we use the expert $\mathfrak{E}_{i,t}$ as an index to access in the memory the parameters of the model created at time i. The computation of $\mathbb{P}(\theta_t || \mathbf{D}_{t-1})$ is done by taking into account the distribution $w_{i,t} = \mathbb{P}(\mathfrak{E}_{i,t} | \mathbf{D}_{t-1})$ of the expert $\mathfrak{E}_{i,t}$ such as:

$$\mathbb{P}(\theta_t || \mathbf{D}_{t-1}) = \sum_{i=1}^{t} \mathbb{P}(\theta_t | \mathbf{D}_{t-1}, \mathfrak{E}_{i,t}) \mathbb{P}(\mathfrak{E}_{i,t} | \mathbf{D}_{t-1}) \tag{3.3.1}$$

Then, unlike the sampling procedure used in [105], we build the index $\theta_{a,t}$ of arm k at time t by launching a Bayesian aggregation of a growing number of experts. The estimation of the expert distribution is done recursively according to the work of [2] where:

$$\underbrace{\mathbb{P}(\mathfrak{E}_{i,t} | \mathbf{D}_{t-1})}_{\texttt{Expert distribution at } t} \propto \sum_{i=1}^{t-1} \overbrace{\mathbb{P}(\mathfrak{E}_{i,t} | \mathfrak{E}_{i,t-1})}^{\texttt{change point prior}} \underbrace{\mathbb{P}(x_t | \mathfrak{E}_{i,t-1}, \mathbf{D}_{t-2})}_{\texttt{Instantaneous gain}} \overbrace{\mathbb{P}(\mathfrak{E}_{i,t-1} | \mathbf{D}_{t-2})}^{\texttt{Expert distribution at } t-1} \tag{3.3.2}$$

The change point prior $\mathbb{P}(\mathfrak{E}_{i,t} | \mathfrak{E}_{i,t-1})$ is naturally computed following equation (3.3.3):

$$\mathbb{P}(\mathfrak{E}_{i,t} | \mathfrak{E}_{i,t-1}) = (1 - \rho)\, \mathbf{1}\{i < t\} + \rho \mathbf{1}\{i = t\} \tag{3.3.3}$$

Thus, the inference model takes the following form (Up to a normalization factor):

- *Growth probability:*
$$\mathbb{P}(\mathfrak{E}_{i,t} | \mathbf{D}_{t-1}) \propto (1 - \rho) \mathbb{P}(x_t | \mathfrak{E}_{i,t-1}, \mathbf{D}_{t-2}) \mathbb{P}(\mathfrak{E}_{i,t-1} | \mathbf{D}_{t-2})$$

- *Change-point probability:*
$$\mathbb{P}(\mathfrak{E}_{t,t} | \mathbf{D}_{t-1}) \propto \rho \sum_{i=1}^{t-1} \mathbb{P}(x_t | \mathfrak{E}_{i,t-1}, \mathbf{D}_{t-2}) \mathbb{P}(\mathfrak{E}_{i,t-1} | \mathbf{D}_{t-2}) \tag{3.3.4}$$

It should be noted that $\mathbb{P}(x_t | \mathfrak{E}_{i,t-1}, \mathbf{D}_{t-2})$ is a Bernoulli distribution of expectation $\frac{S_{A_t,i,t-1}}{S_{A_t,i,t-1} + F_{A_t,i,t-1}}$, where $S_{A_t,i,t-1}$ and $\mathfrak{E}_{A_t,i,t-1}$ are the hyper-parameters of the arm A_t learned by the expert $\mathfrak{E}_{i,t-1}$. Let $l_{i,t-1}$ denotes the instantaneous logarithmic loss associated to the forecaster $\mathfrak{E}_{i,t-1}$ such as: $\mathbb{P}(x_t | \mathfrak{E}_{i,t-1}, \mathbf{D}_{t-2}) = \exp(-l_{i,t-1})$, then Algorithm 3.1 provides us an index prediction of each arm k at time t based on a Bayesian aggregation of the available experts.

Algorithm 3.1 Bayesian Aggregation with a growing number of experts

for $t = 2, \ldots$ **do**

-1- Update the previous experts:

$$\forall i \in \{1, \ldots, t-1\} \quad w_{i,t} = (1-\rho)\exp\left(-l_{i,t-1}\right) w_{i,t-1}$$

-2- Create new expert starting at t:

$$w_{t,t} = \rho \sum_{i=1}^{t-1} w_{i,t-1} \exp\left(-l_{i,t-1}\right)$$

-3- Compute arm index:

$$\forall a \in \mathcal{A} \; \theta_{a,t} = \frac{\sum_{i=1}^{t} \theta_{a,i,t} w_{i,t}}{\sum_{j=1}^{t} w_{j,t}} \text{ where: } \theta_{a,i,t} \sim \text{Beta}\left(S_{a,i,t}, F_{a,i,t}\right)$$

end for

Finally, by plugging the Bayesian aggregation into the formalism of [105], we get the *Global Switching Thompson Sampling with Bayesian Aggregation* (Global-STS-BA), described in Algorithm 3.2.

Algorithm 3.2 Global Switching **T**hompson **S**ampling with **B**ayesian **A**ggregation

procedure Global-STS-BA($\mathcal{A}, T, s_0, f_0, \rho$)
  $t \leftarrow 1, w_{1,t} \leftarrow 1$, and $\forall a \in \mathcal{A}\ S_{a,1,t} \leftarrow s_0,\ F_{a,1,t} \leftarrow f_0$ ▷ Initializations
  for $t \leqslant T$ **do** ▷ Interaction with environment
    $A_t \leftarrow$ `ChooseArm` $(\{w\}^t, \{S\}^t, \{F\}^t)$
    $x_t \leftarrow$ `PlayArm`(A_t) ▷ Bernoulli trial
    $\{w\}_{t+1} \leftarrow$ `UpdateExpertWeight` $(\{w\}_t, \{S\}_{a_t,t}, \{F\}_{a_t,t}, x_t, \rho)$
    $\{S\}_{t+1}, \{F\}_{t+1} \leftarrow$ `UpdateArmModel`$(\{S\}_t, \{F\}_t, x_t, A_t)$
  end for
end procedure

---

procedure `ChooseArm`$(\{w\}_t, \{S\}_t, \{F\}_t)$
  $\forall a \in \mathcal{A}\ \forall i \in [1, t] \quad \theta_{a,i,t} \sim \text{Beta}(S_{a,i,t}, F_{a,i,t})$
  return $\text{argmax}_a \sum_{i=1}^{t} \frac{w_{i,t}}{\sum_{j=1}^{t} w_{j,t}} \theta_{a,i,t}$ ▷ Bayesian aggregation
end procedure

---

procedure `UpdateExpertWeight`$(\{w\}_t, \{S\}_{a_t,t}, \{F\}_{a_t,t}, x_t, \rho)$
  $l_{i,t} \leftarrow -x_t \log\left(\frac{S_{A_t,i,t}}{S_{A_t,i,t}+F_{A_t,i,t}}\right) - (1-x_t)\log\left(\frac{F_{A_t,i,t}}{S_{A_t,i,t}+F_{A_t,i,t}}\right) \forall i \in [1, t]$
  $w_{i,t+1} \leftarrow (1-\rho)\exp(-l_{i,t})\, w_{i,t} \forall i \in [1, t]$ ▷ Increasing the size of expert $\mathfrak{E}_{i,t}$
  $w_{t+1,t+1} \leftarrow \rho \sum_i \exp(-l_{i,t})\, w_{i,t}$ ▷ Creating new expert starting at $t+1$
  `Normalise` $\{w\}_{t+1}$
  return $\{w\}_{t+1}$
end procedure

---

procedure `UpdateArmModel`$(\{S\}_t, \{F\}_t, x_t, A_t)$
  $S_{A_t,i,t+1} \leftarrow S_{A_t,i,t} + \mathbf{1}\{x_t = 1\}\ \forall i \in [1, t]$
  $F_{A_t,i,t+1} \leftarrow F_{A_t,i,t} + \mathbf{1}\{x_t = 0\}\ \forall i \in [1, t]$
  $S_{a,t+1,t+1} \leftarrow s_0, \quad F_{a,t+1,t+1} \leftarrow f_0 \quad \forall a \in \mathcal{A}$ ▷ Initializing new expert
  return $\{S\}_{t+1}, \{F\}_{t+1}$
end procedure

Discussion 3.3.1 (Global-STS-BA)**.** *In the global switching setting, when a switch occurs, all arms change their expected pay-off at the same time. It should be noted that the data from all plays collaborate in the posterior of the expert distribution estimation (*`UpdateExpertWeight`*). The experts tell us how much previous observed data can be used in the arm indexes prediction, i.e. when a switch occurs all past data have not to be taken into account in the arm characterizations because of their obsolete information. The change-point detection concept is based on a tracking of the optimal expert (see section 3.5.1). Indeed, the total mass of the expert distribution $w_{i,t}$ tends to focus around the optimal expert $\mathfrak{E}_{\tau_\kappa,t}$ i.e. the expert starting at the most recent change point τ_κ and corresponds to the most appropriate characterization of the environment. The Bayesian aggregation exploits this concentration to highlight the contribution of the most appropriate experts (starting around the most recent change point τ_κ) in the arm indexes prediction. The concentration around $\mathfrak{E}_{\tau_\kappa,t}$ is possible thanks to the inference model of equation (3.3.4). In fact, when a change occurs the instantaneous gain $\mathbb{P}(x_t \mid \mathfrak{E}_{i,t-1}, \mathbf{D}_{t-2})$ of all experts starting before the change point suddenly fall down because of their wrong estimation of the environment, giving the advantage to the experts newly created while annihilating the former ones (see*

figures 3.5.1a, 3.5.1b). At this point, we deviate from [105] and instead of sampling the expert distribution and then sampling the arms, we index each arm k by launching an overall Bayesian aggregation of samples taken from the posterior distribution of arm k related to the hyper-parameters $\{S\}_{a,t}, \{F\}_{a,t}$ (`ChooseArm`). Finally, the agent chooses to pull the arm with highest index. This process allows us to avoid the sampling noise induced by Global-STS and by this way the model chosen at time t tends to better fit the unknown environment (figure 3.4.1).

3.4 Experiments

In all the experiments, we consider a GS-MAB of three arms observing three change points occurring at each 1000 rounds. Experiments are run 100 times. The parameters of the state-of-the art algorithms are chosen to be experimentally optimal. Exp3, Exp3P and Exp3S [16] are launched with an exploration rate ($\gamma = 5\%$). We run Exp3R [6], Rexp3 [22] and SW-UCB [58] respectively with $H = 1000$, $\Delta_T = 1000$ and $\tau = 500$. Global-STS [105] and Global-STS-BA are outperforming the well parametrized state of art non stationary MAB algorithms (figure 3.4.1).
Replacing the expert distribution sampling used in [105] with the Bayesian Aggregation allows us to obtain performances challenging those of the Thompson Sampling Oracle (figure 3.4.1).

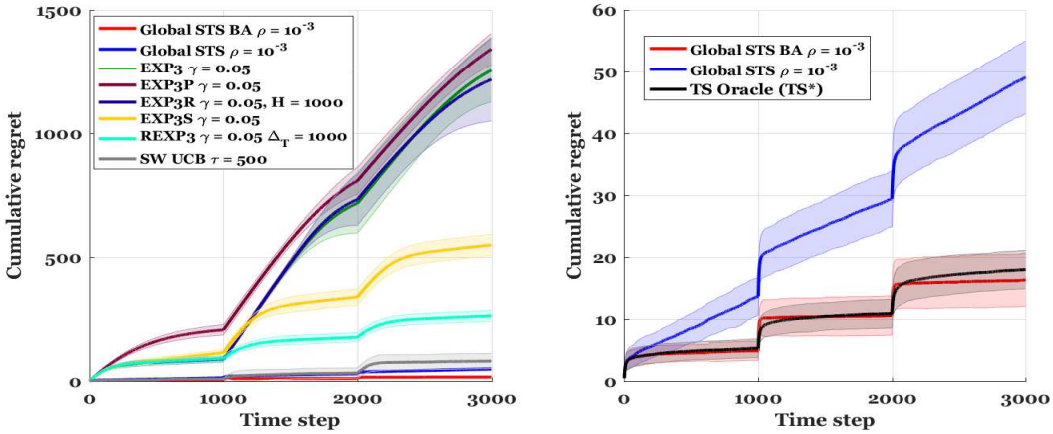


Figure 3.4.1: Overall comparison with the non-stationary state of art and the TS Oracle.

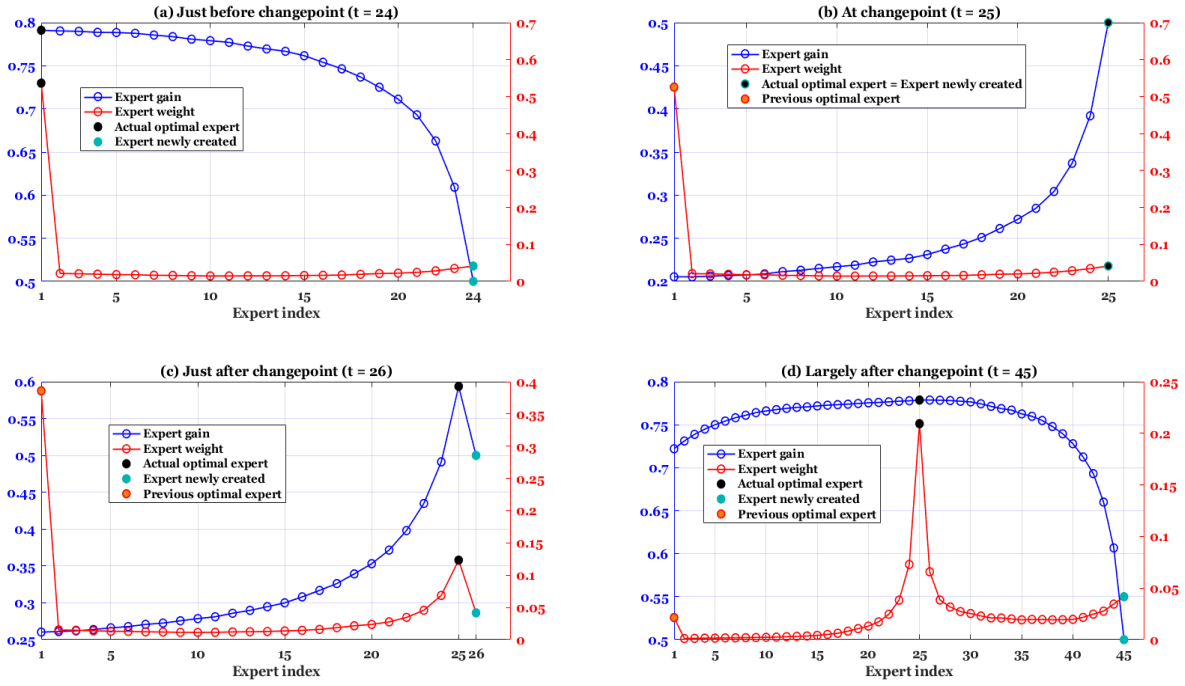
3.5 Numerical Illustrations in the global switching setting

3.5.1 Tracking the optimal expert in the global switching setting

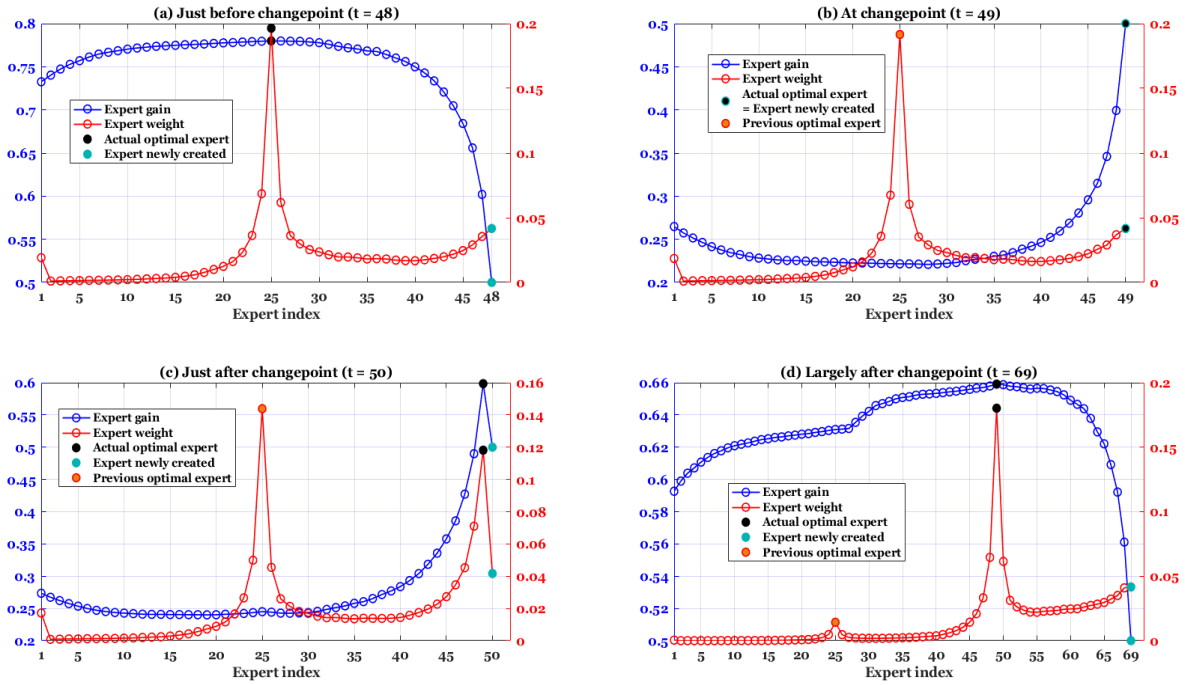
Optimal expert Let $t > 0$ and let $\tau_{[t]}$ denotes the most recent change point before time t. At each time t, a set of exactly t experts is available (each expert has started at $i \in [1, t]$). The optimal expert is the one which has started exactly at $t = \tau_{[t]}$. It is the expert which gives the best description of the environment.

Tracking the optimal expert Before the decision step, Global-STS-BA characterizes each arm k by aggregating all the contributions of the available experts. To do well, Global-STS-BA needs to highlight the contribution of the optimal expert because of its optimal description of the environment. This task is called *the track of the optimal expert*.

The estimation of the expert distribution (expert weight) is built according to the instantaneous gain of each expert. This gain takes into account the history of past rewards $\mathbf{D}_{t-1}$. When a switch occurs, the instantaneous gains of all experts fall down because of their obsolete estimation giving the advantage to the expert newly created which will become the optimal expert during the next rounds.
Let us consider a GS-MAB of three arms observing two change points at time $t = 25$ and $t = 49$. Figures 3.5.1a and 3.5.1b show the behavior of the expert gain and the expert weight when a switch occurs. It should be noted that the experts are indexed by their starting time i.e. the most recent expert is the one with highest index and inversely.



(a) Tracking the optimal expert during the first switch.



(b) Tracking the optimal expert during the second switch.

Discussion 3.5.1 (Tracking the optimal expert)**.** • *Just before the change point, the optimal expert (index* = 1 *for the first switch and index* = 25 *for the second one)*

is given the highest weight. The lower the index, the higher the expert gain. This behavior is expected as much as the higher the number of the observations the higher the expert gain.

- *When the switch occurs, all the experts created before the change point see their gains dropping sharply because of the abrupt change of the environment i.e. the instantaneous gain doesn't match anymore with the current environment, except the expert newly created which is given a gain based on the prior of a Thompson Sampling* $\left(\frac{s_0}{s_0+f_0}\right)$.
- *Just after the change point, the new optimal expert see its weight starting to grow up while the weight of the previous optimal expert starts to fall down.*
- *Largely after the change point, the optimal expert is given the highest weight, because of its well fit of the current environment. This is corresponding to the expected behavior of the Global-STS-BA:* **the optimal expert is well tracked.**

3.5.2 Behavior of Global-STS-BA with respect to the memory size

Regarding the implementation of the Global-STS-BA, we should notice that memory space and time requirements of the experts inference model grow linearly at each time step because of the creation of a new expert. This makes the size of the support set of the expert distribution $\{w\}_t$ increasing by one. Thus, for computational reasons, we propose to restrict the number of experts by fixing a maximum memory size M. So, at each round, after launching the inference model, we delete the worst expert $\mathfrak{E}_t^{\min} = \arg\min_i w_{i,t}$. In all the experiments, we consider a GS-MAB of three arms. Changes occur at each 1000 rounds. We variate the memory size from $M = 15$ to $M = 3000$. Experiments are run 100 times.

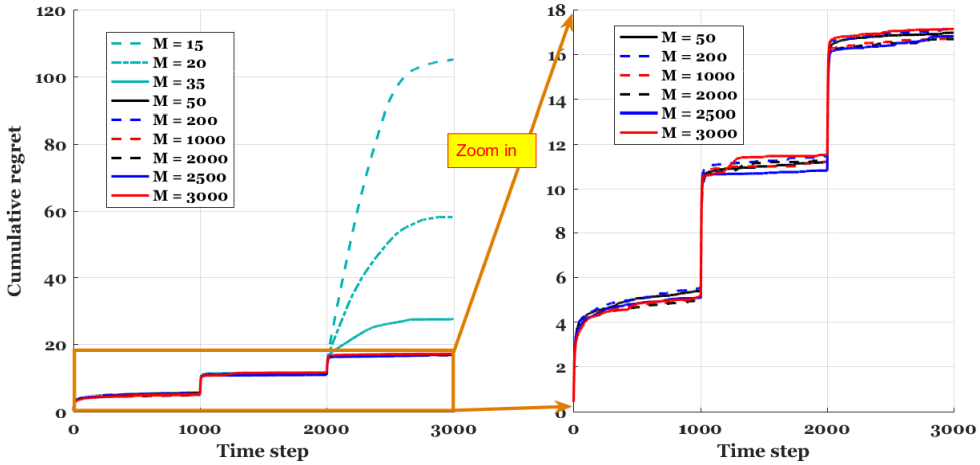


Figure 3.5.2: Behavior with respect to the switching rate ρ.

For a not very poor memory size value ($M \geqslant 50$), the performances of the Global-STS-BA remains stable.

3.5.3 Behavior of Global-STS-BA with respect to the switching rate ρ

We should also notice that the inference model of the experts needs the knowledge of the true switching rate (ρ_{true}). In real life, this value is unknown to the agent. [138] has proposed

to learn the switching rate from the data via a gradient descent. This method appears to not perform particularly well if the switching rate has to be adapted at every time step. Figure 3.5.3 shows the behavior of the Global-STS-BA for different values of the switching rate.

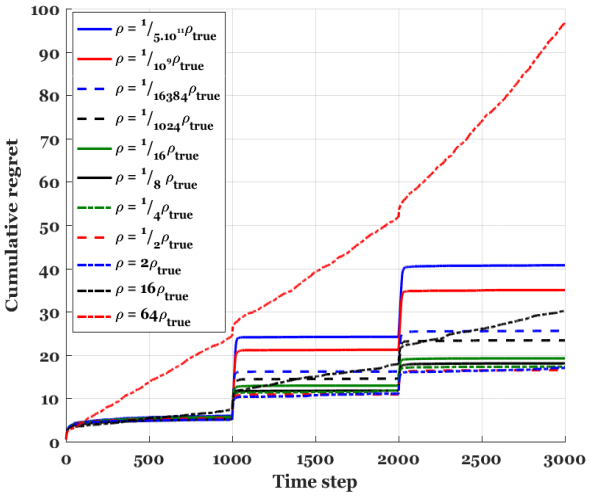


Figure 3.5.3: Cumulative regret versus the value of the switching rate.

The performances of the Global-STS-BA remains stable even if the switching rate is quite far from the true one.

3.6 Extension to the per-arm switching setting

3.6.1 Per-Arm Switching Thompson Sampling with Bayesian Aggregation (Per-Arm-STS-BA)

In the per-arm switching setting, since the changes occur independently from an arm to another, each arm k will have its own expert distribution denoted by $\{w\}_a^t$. To estimate $\{w\}_a^t$ at each time step t, we need to build some recursive message-passing algorithm. The main difference between the global and the per-arm version of the message-passing is that the instantaneous gain will be used in the update of the expert distribution associated to the pulled arm A_t. For the other arms not pulled, the instantaneous gain won't intervene in the expert distribution [105]. More formally, if we denote by $\mathfrak{E}_{a,i,t}$ the expert associated to the arm k at time t which has been introduced at time i, then the expert distributions are updated as follow:

$$\mathbb{P}(\mathfrak{E}_{a,i,t}|\mathbf{D}_{t-1}) \propto \begin{cases} \sum_{i=1}^{t-1} \mathbb{P}(\mathfrak{E}_{a,i,t}|\mathfrak{E}_{a,i,t-1})\mathbb{P}(x_t|\mathfrak{E}_{a,i,t-1},\mathbf{D}_{t-2})\mathbb{P}(\mathfrak{E}_{a,i,t-1}|\mathbf{D}_{t-2}) & \text{if } A_t = a \\ \sum_{i=1}^{t-1} \mathbb{P}(\mathfrak{E}_{a,i,t}|\mathfrak{E}_{a,i,t-1})\mathbb{P}(\mathfrak{E}_{a,i,t-1}|\mathbf{D}_{t-2}) & \text{otherwise} \end{cases}$$

Then, we model the changes of the arm a with a constant switching rate γ_a such that:

$$\mathbb{P}(\mathfrak{E}_{a,i,t}|\mathfrak{E}_{a,i,t-1}) = \begin{cases} 1-\rho_a & \text{if } i < t \\ \rho_a & \text{if } i = t \end{cases}$$

Then, following the inference model used in the global switch, we deduce the following expert distributions update rules (Up to a normalization factor):

- *Growth probability:*

$$\mathbb{P}(\mathfrak{E}_{a,i,t}) \propto \begin{cases} (1-\rho_k)\mathbb{P}(x_t|\mathfrak{E}_{a,i,t-1})\mathbb{P}(\mathfrak{E}_{a,i,t-1}) & \text{if } A_t = a \\ (1-\rho_k)\mathbb{P}(\mathfrak{E}_{a,i,t-1}) & \text{otherwise} \end{cases}$$

- *Change-point probability:*

$$\mathbb{P}(\mathfrak{E}_{a,i,t}) \propto \begin{cases} \rho_k \sum_{i=1}^{t-1} \mathbb{P}(x_t|\mathfrak{E}_{a,i,t-1})\mathbb{P}(\mathfrak{E}_{a,i,t-1}) & \text{if } A_t = a \\ \rho_k \sum_{i=1}^{t-1} \mathbb{P}(\mathfrak{E}_{a,i,t-1}) & \text{otherwise} \end{cases}$$

Notice that: $\mathbb{P}(x_t|\mathfrak{E}_{a,i,t-1})$ still a Bernoulli distribution of expectation $\frac{S_{A_t,i,t-1}}{S_{A_t,i,t-1}+F_{A_t,i,t-1}}$, where $S_{A_t,i,t-1}, F_{A_t,i,t-1}$ are the hyper-parameters of arm A_t at time $t-1$ learned by the expert $\mathfrak{E}_{A_t,i,t-1}$ which is associated to the change point model of arm A_t. We define:

$$\mathbb{P}(x_t|\mathfrak{E}_{A_t,i,t-1}) = \exp\left(-l_{i,t-1}\right)$$

where: $l_{i,t-1}$ denotes the instantaneous logarithmic loss of the pulled arm A_t associated to the forecaster $\mathfrak{E}_{A_t,i,t-1}$. For convenience, we write: $w_{a,i,t} = \mathbb{P}(\mathfrak{E}_{a,i,t}|\mathbf{D}_{t-1})$. Then, we naturally extend the Bayesian aggregation used in the global switching setting to the per-arm Bayesian Aggregation (Algorithm 3.3).

Algorithm 3.3 Per-arm Bayesian Aggregation

for $t = 2, \ldots$ **do**

-1- Update the previous experts:

$$\forall i \in [1, t-1] \;\; \forall a \in \mathcal{A} \quad w_{a,i,t} = (1-\rho_k)\, w_{a,i,t-1} \exp\left(-l_{i,t-1}\right)^{1\{a=A_t\}}$$

-2- Create new expert starting at t:

$$\forall a \in \mathcal{A} \quad w_{a,t,t} = \rho_k \sum_{i=1}^{t-1} w_{a,i,t-1} \exp\left(-l_{i,t-1}\right)^{1\{a=A_t\}}$$

-3- Predict arm index:

$$\forall a \in \mathcal{A} \quad \theta_{a,t} = \frac{\sum_{i=1}^{t} w_{a,i,t}\theta_{a,i,t}}{\sum_{j=1}^{t} w_{a,j,t}} \text{ where: } \theta_{a,i,t} \sim \text{Beta}\left(S_{a,i,t}, F_{a,i,t}\right)$$

end for

Then, by plugging the per-arm Bayesian Aggregation into the formalism of the Global-STS-BA we get the *Per-arm Switching Thompson Sampling with Bayesian Aggregation* (Per-arm-STS-BA) described in Algorithm 3.4.

Algorithm 3.4 Per-Arm Switching Thompson Sampling with Bayesian Aggregation

procedure Per-Arm-STS-BA($\mathcal{A}, T, s_0, f_0, \rho$)
  $t \leftarrow 1$, and $\forall a \in \mathcal{A}\ w_{a,1,t} \leftarrow 1, S_{a,1,t} \leftarrow s_0,\ F_{a,1,t} \leftarrow f_0$ ▷ Initializations
  for $t \leqslant T$ **do** ▷ Interaction with environment
    $A_t \leftarrow \text{ChooseArm}\ (\{w\}^t, \{S\}^t, \{F\}^t)$
    $x_t \leftarrow \text{PlayArm}(A_t)$ ▷ Bernoulli trial
    $\{w\}_{t+1} \leftarrow \text{UpdateExpertWeight}\ (\{w\}_t, \{S\}_{a_t,t}, \{F\}_{a_t,t}, x_t, \rho)$
    $\{S\}_{t+1}, \{F\}_{t+1} \leftarrow \text{UpdateArmModel}(\{S\}_t, \{F\}_t, x_t, A_t)$
  end for
end procedure

---

procedure ChooseArm($\{w\}_t, \{S\}_t, \{F\}_t$)
  $\forall a \in \mathcal{A}, \forall i \in [1,t] \quad \theta_{a,i,t} \sim \text{Beta}\,(S_{a,i,t}, F_{a,i,t})$
  return $\text{argmax}_a \sum_{i\in[1,t]} \frac{w_{a,i,t}}{\sum_{j\in[1,t]} w_{a,j,t}}\, \theta_{a,i,t}$ ▷ Bayesian aggregation
end procedure

---

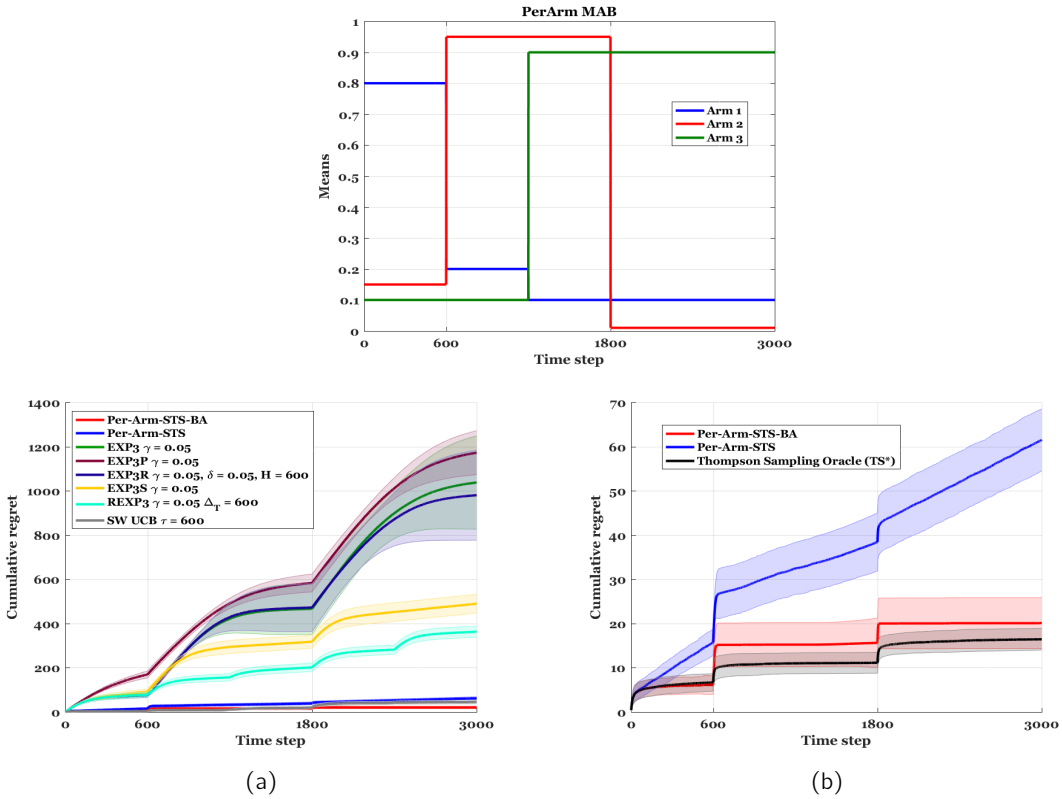
procedure UpdateExpertWeight($\{w\}_t, \{S\}_{a_t,t}, \{F\}_{a_t,t}, x_t, \rho$)
  $l_{i,t} \leftarrow -x_t \log\left(\frac{S_{A_t,i,t}}{S_{A_t,i,t}+F_{A_t,i,t}}\right) - (1-x_t)\log\left(\frac{F_{A_t,i,t}}{S_{A_t,i,t}+F_{A_t,i,t}}\right) \forall i \in [1,t]$
  $w_{A_t,i,t+1} \leftarrow (1-\rho)\, w_{A_t,i,t} \exp(-l_{i,t}) \forall i \in [1,t]$ ▷ Increasing the size of expert $\mathfrak{E}_{A_t,i,t}$
  $w_{A_t,t+1,t+1} \leftarrow \rho \sum_i w_{A_t,i,t} \exp(-l_{i,t})$ ▷ Creating new expert for arm A_t starting at $t+1$
  $w_{a,i,t+1} \leftarrow (1-\rho)\, w_{a,i,t}\ \forall i \in [1,t], \forall a \neq A_t$ ▷ Increasing the size of expert $\mathfrak{E}_{a,i,t}$
  $w_{a,t+1,t+1} \leftarrow \rho \sum_i w_{a,i,t}\ \forall a \neq A_t$ ▷ Creating new expert for arm k starting at $t+1$
  `Normalise` $\{w\}_{t+1}$
  return $\{w\}_{t+1}$
end procedure

---

procedure UpdateArmModel($\{S\}_t, \{F\}_t, x_t, A_t$)
  $S_{A_t,i,t+1} \leftarrow S_{A_t,i,t} + \mathbf{1}\{x_t = 1\}\ \forall i \in [1,t]$
  $F_{A_t,i,t+1} \leftarrow F_{A_t,i,t} + \mathbf{1}\{x_t = 0\}\ \forall i \in [1,t]$
  $S_{a,t+1,t+1} \leftarrow s_0, \quad F_{a,t+1,t+1} \leftarrow f_0 \quad \forall a \in \mathcal{A}$ ▷ Initializing new expert
  return $\{S\}_{t+1}, \{F\}_{t+1}$
end procedure

3.6.2 Experiments

In all the experiments, we consider a Per-Arm Switch MAB of three arms observing several change points. Experiments are run 100 times. Per-Arm-STS [105] and Per-Arm-STS-BA are outperforming the well parametrized state-of-art non stationary MAB algorithms (figure 3.6.1). Exp3, Exp3P and Exp3S [16] are launched with an exploration rate ($\gamma = 5\%$). We run Exp3R [6], Rexp3 [22] and SW-UCB [58] respectively with $H = 600$, $\Delta_T = 600$ and $\tau = 600$. Replacing the expert distribution sampling used in STS with the Bayesian Aggregation allows us to obtain performances challenging those of the Thompson Sampling Oracle (figure 3.6.1).



(a) (b)

Figure 3.6.1: Overall comparison with the state of art and the oracle.

3.7 Conclusion and future works

We have proposed Global-STS-BA: an extension of the Thompson Sampling for the Switching Bandit Problem based on a Bayesian aggregation framework. From the experiments, the proposed algorithm compares favorably with the previous version of the Global Switching Thompson Sampling [105], outperforming clearly other algorithms of the state-of-the-art. It is worth noting that Global-STS-BA challenges the Thompson sampling oracle, an oracle which already knows the change points. These results arise from the fact that Global-STS-BA is based on the Bayesian concept of tracking the best experts which allows us to catch efficiently the change points. Moreover, we have also extended the proposed algorithm to the *Per-arm Switching Multi-Armed Bandit* by allowing an expert distribution per arm. Obviously, we can easily extend the hybridization of Thompson Sampling and the Bayesian change point detector to the whole family of the stochastic bandit namely KL-UCB, UCB, Bayes UCB, UCB V, ... so we can form the family of memory bandits (figure 3.7.1). The next step of this work is to analyze the Global-STS-BA (or more generally the memory bandits) in term of regret.

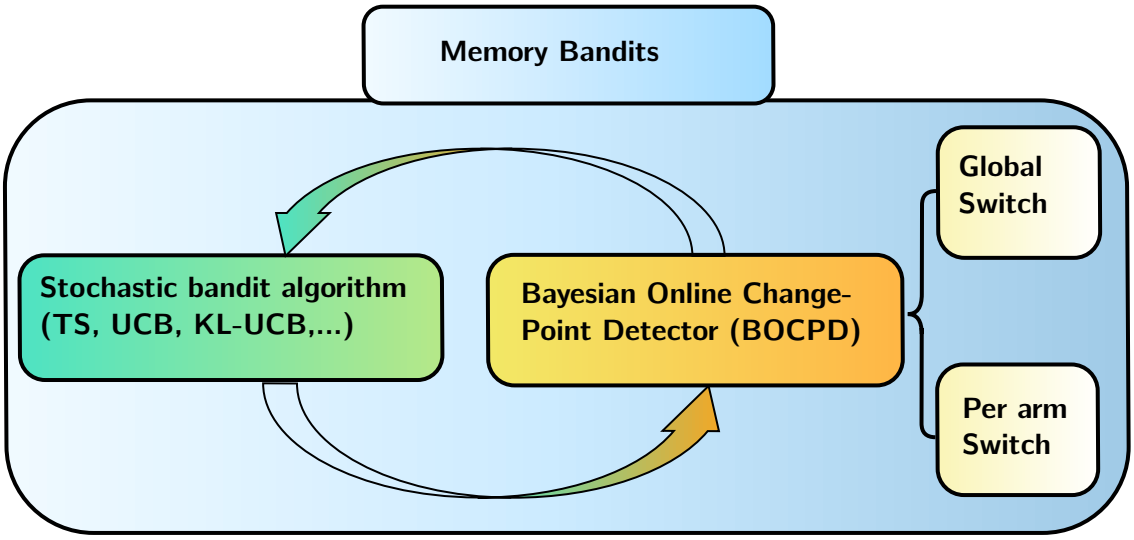


Figure 3.7.1: The memory bandits family: a hybridization between a stochastic bandit and the Bayesian change point detector

Chapter 4

Restarted Bayesian Online Change-point detection strategy

Overview of the chapter

In this chapter, we consider the problem of sequential change-point detection where both the change-points and the distributions before and after the change are assumed to be unknown. For this problem of primary importance in statistical and sequential learning theory, we derive a variant of the Bayesian Online Change Point Detector proposed by [46] which is easier to analyze than the original version while keeping its powerful message-passing algorithm. We provide a non-asymptotic analysis of the false-alarm rate and the detection delay that matches the existing lower-bound. We further provide the first explicit high-probability control of the detection delay for such approach. Experiments on synthetic and real-world data show that this proposal outperforms the state-of-art change-point detection strategy, namely the Improved Generalized Likelihood Ratio (Improved GLR) while compares favorably with the original Bayesian Online Change Point Detection strategy.

Publication. This chapter is mainly based on our article [5].

4.1 Introduction

The problem of online detecting abrupt variations (change-points) in the generative parameters of a sequence of observations $x_1, \dots x_n$, where observations are received one by one, is considered. Addressing this problem is useful in a number of real-world applications including finance [126], genetics [63], cybersecurity [118], robotics [61, 25, 88], speech recognition [113], climate modeling [107]. The online change-point detection problem has received a lot of attention from various areas of mathematical statistics, information theory and computer science over the past century. We refer the interested reader to the recent survey [10] on the large amount of methods developed for time series change point detection, and to [20, 27, 75, 132, 37, 146] for classical textbooks on change-points. As noticed in [10], performance guarantees are still lacking for many such methods, especially in terms of finite time guarantee on the detection delay and estimation of the change-gap, both important features for the practitioner. Amongst the many methods, the celebrated CUSUM strategy from [112] and its extension called Generalized Likelihood Ratio (GLR), that are following a frequentist approach based on likelihood ratio thresholding have been analyzed recently first in [93] and then in [102], where a fully explicit parameter tuning is also provided, together with fully non-asymptotic guarantees.

In this chapter, we turn to Bayesian approaches. In the seminal paper of [46], the authors introduced the Bayesian Online Changepoint Detection (BOCPD) strategy to infer the most recent change-point, by computing the probability distribution of the elapsed time since the last change-point (runlength). Although the algorithm has been used extensively (including in non-stationary multi-armed bandits, [105, 4, 83, 39] and other change-point context [2, 138, 147, 145, 124, 32, 110, 137, 123, 84, 85]), up to our knowledge, no formal analysis of its performance in terms of false-alarm or detection delay has been performed except the work in [85], where the authors has built a robust BOCPD version to reduce false discovery rates.

Note that although BOCPD stands for Bayesian Online Change Point Detection, the algorithm performs no detection at all; rather, it maintains weights to estimate the elapsed time since the last change-point. Following this work, we provide a modification of the BOCPD strategy that we analyze. In particular we provide a non-asymptotic guarantees related to the false-alarm (that is, detecting a change point while there was no change) in Theorem 4.10 and related to the detection delay (the number of steps after a change-point occurs before we declare detection) in Theorem 4.11.

In Section 4.2, we formally introduce the times-series model with abrupt changes, as well as notations. We provide in Section 4.3 a new formulation of the BOCPD strategy from [2], that we reinterpret from the standpoint of aggregation of forecasters, leading to a compact formulation presented in Algorithm 4.1. We then present a simple way to make use of this strategy to effectively detect changes, instead of just estimating the time since the last change. We note that the analysis of BOCPD involves dealing with a combinatorial number of terms, and propose a simplification of this strategy in order to derive performance guarantees. We call the resulting strategy R-BOCPD for *Restarted Bayesian Online Change Point Detection*. Then, we provide in Section 4.5 the two theoretical guarantees of this strategy: namely the false alarm rate control and detection delay (Theorem 4.10, Theorem 4.11). Finally, we show numerically that this strategy outperforms its previous version BOCPD and its compares favorably with the Improved GLR strategy introduced by [102].

4.2 Sequential change-point detection setting

Sequential change-point detection, which is rooted in classical statistical sequential analysis [20], aims to detect the change in underlying distributions of a sequence of observations as quickly as possible.

In this chapter, we study the online change point detection problem, where a sequence of independent univariate random variables with common fluctuation upper bound are collected, and the mean may change at one or multiple time points. Indeed, we consider an agent aiming at detecting changes in the generation of an online stream. At each time step t, the agent observes the datum $x_t \sim \mathcal{B}(\mu_t)$: a random variable following the Bernoulli distribution of mean μ_t and need to decide whether or not there is a change in the generation of the stream. Alternatively, the agent may compute at each time step t, an estimation $\widehat{\tau}_t$ of the last change-point.

Definition 4.1: Piece-wise stationary Bernoulli process
Let T denote the time horizon of the game (stream length) and C_T the overall number of change-points observed until time T. We assume that the observations $x_t \sim \mathcal{B}(\mu_t)$ are generated by a piece-wise Bernoulli process such that there exists a non-decreasing change-points sequence $(\tau_c)_{c\in[1,C_T]} \in \mathbb{N}^{C_T}$ verifying:

$$\begin{cases} \forall c \in [1, C_T]\,,\ \forall t \in \mathcal{T}_c = [\tau_c, \tau_{c+1}) \quad \mu_t = \theta_c, \\ \tau_1 = 1 < \tau_2 < \ldots < \tau_{C_T+1} = T + 1. \end{cases} \tag{4.2.1}$$

Remark 1 (Interests in the Bernoulli case)**.** *The interests in working on the Bernoulli distributions are not as restrictive as it seems. On the first hand, from a concentration point of view, Bernoulli distributions can be seen as a worst case of bounded distributions. Moreover, Bernoulli distributions are crucially used in many widespread applications of machine learning. For instance:*

- *modelling the collisions in cognitive radio,*
- *monitoring the performances of statistical models,*
- *monitoring events in probes for network supervision,*
- *the multi armed bandit problem,*
- *experiments in clinical trials and recommender systems.*

Notation 3. *In the following, we denote by* $\mathbf{x}_{s:t} := (x_s, \ldots, x_t)$ *the sequence of observations from time* s *up to time* $t \geqslant s$*. Furthermore, the length of the sequence* $\mathbf{x}_{s:t}$ *is denoted by* $n_{s:t} := t - s + 1$ *and the empirical mean over the sequence* $\mathbf{x}_{s:t}$ *is denoted by* $\widehat{\mu}_{s:t} := \frac{1}{n_{s:t}} \sum_{i=s}^{t} x_i$*.*

Definition 4.2: Online change-point detection strategy
An online change-point strategy $\mathcal{A}$ takes as input a sequence $\mathbf{x}_{r:t}$ and output a binary scalar such that:

$$\mathcal{A}(\mathbf{x}_{r:t}) = \begin{cases} 1 & \text{if a change in the generation of the sequence } \mathbf{x}_{r:t} \text{ is detected,} \\ 0 & \text{else.} \end{cases}$$

A strategy is said to be **anytime** if it does not depend on the time horizon T which denotes the stream length.

Performance assessment Let: $\mathbf{x}_{r:\tau_c-1} \sim \mathcal{B}(\theta_1)^{\otimes n_{r:\tau_c-1}}$, $\mathbf{x}_{\tau_c:t} \sim \mathcal{B}(\theta_2)^{\otimes n_{\tau_c:t}}$, τ_c the change-point to detect and r the starting time. The performance of an algorithm that aims at detecting the change-point $\tau_c \in [r, t]$ in the sequence $\mathbf{x}_{r:t}$ is assessed using two notions.

- **False alarm rate**: the probability of detecting a change at some instant $s \in [r, \tau_c)$ where there is no change. Usually, the false alarm rate is expressed as: $\mathbb{P}\Big(\exists s \in [r, \tau_c) : \mathcal{A}(\mathbf{x}_{r:s}) = 1\Big)$.

- **Detection delay**: the number of time steps needed to detect a change. It is formally defined for a strategy $\mathcal{A}$ as: $\widehat{\tau}_{\mathcal{A}}(\mathbf{x}_{r:t}) := \min\{s \in [r,t] : \mathcal{A}(\mathbf{x}_{r:s}) = 1\}$. Thus, the detection delay is expressed as: $\mathfrak{D}_{|\theta_2-\theta_1|,r,\tau_c} := (\widehat{\tau}_{\mathcal{A}}(\mathbf{x}_{r:t}) - \tau_c) \times \mathbf{1}\{\widehat{\tau}_{\mathcal{A}}(\mathbf{x}_{r:t}) > \tau_c\}$, where $\mathbf{1}\{\bullet\}$ denotes the indicator function.

Currently, the literature provides us with an interesting asymptotic lower bound on the **expected** detection delay. Theorem 4.9 gives the lower bound. (It is a reformulation of Theorem 3.1 in [93]).

Theorem 4.9: Asymptotic lower bound on the expected detection delay
Let: $\mathbf{x}_{r:\tau_c-1} \sim \mathcal{B}(\theta_1)^{\otimes n_{r:\tau_c-1}}$, $\mathbf{x}_{\tau_c:t} \sim \mathcal{B}(\theta_2)^{\otimes n_{\tau_c:t}}$, $\mathcal{A}$ an online change-point detection strategy, τ_c the change-point to detect and r the starting time. Assuming that the false alarm rate is controlled such that: $\mathbb{P}_{\theta_1}(\exists s \in [r,\tau_c) : \mathcal{A}(\mathbf{x}_{r:s}) = 1) \leqslant \delta$, then as the quantity $\frac{n_{r:\tau_c}}{|\log\delta|} \underset{\delta\to 0}{\to} \infty$, the expected detection delay $\mathbb{E}_{\theta_1,\theta_2}[\widehat{\tau}_{\mathcal{A}}(\mathbf{x}_{r:t}) - \tau_c]$ is lower bounded as follows:

$$\mathbb{E}_{\theta_1,\theta_2}[\widehat{\tau}_{\mathcal{A}}(\mathbf{x}_{r:t}) - \tau_c] \geqslant \left(\frac{\mathbb{P}_{\theta_1}(\widehat{\tau}_{\mathcal{A}}(\mathbf{x}_{r:t}) > \tau_c)}{\mathbf{kl}(\theta_2,\theta_1)}\right) \log\frac{1}{\delta}$$

where $\mathbf{kl}(\bullet,\bullet)$ stands for the Kullback-Leibler divergence for Bernoulli distributions.

Proof. See [93]. □

4.3 The original Bayesian Online Change Point Detector (BOCPD)

In this section, we describe the original version of the Bayesian Online Change Point Detector introduced in [46] and then revisited in [2]. Then,we reformulate it in term of a learning procedure using a growing number of forecaster. By this way, we highlight the difficulty of its analysis.

4.3.1 Learning using the runlength inference

Notion of runlength. In order to deal with the non-stationary behavior of the environment, the notion of runlength has been introduced by [2]. It represents the overall number of time steps since the last change-point. We denote the length of the current run at time $t \geqslant 1$ by r_t. Since r_t is unknown, we can consider the runlength as a random variable taking values in $\mathcal{R}_t = [0, t-1]$. Thereby, let $p(r_t|\mathbf{x}_{1:t})$ denotes the distribution of r_t given the sequence of observations $\mathbf{x}_{1:t}$.

Computation of $p(r_t|\mathbf{x}_{1:t})$ based on a message passing algorithm. The authors of [2] have proposed an online recursive runlength estimation in order to calculate the runlength distribution $p(r_t|\mathbf{x}_{1:t})$. More specifically to find:

$$p(r_t|\mathbf{x}_{1:t}) = \frac{p(r_t,\mathbf{x}_{1:t})}{p(\mathbf{x}_{1:t})}. \tag{4.3.1}$$

We seek the joint distribution over the past estimated runlengths r_{t-1} as follows:

$$\begin{aligned} p\left(r_t, \mathbf{x}_{1:t}\right) &\overset{(a)}{=} \sum_{r_{t-1} \in \mathcal{R}_{t-1}} p\left(r_t, \mathbf{x}_{1:t}, r_{t-1}\right) \\ &\overset{(b)}{=} \sum_{r_{t-1} \in \mathcal{R}_{t-1}} p\left(r_t, x_t | r_{t-1}, \mathbf{x}_{1:t-1}\right) p\left(r_{t-1}, \mathbf{x}_{1:t-1}\right) \\ &\overset{(c)}{=} \sum_{r_{t-1} \in \mathcal{R}_{t-1}} p\left(x_t | \cancel{r_t}, r_{t-1}, \mathbf{x}_{1:t-1}\right) p\left(r_t | r_{t-1}, \cancel{\mathbf{x}_{1:t-1}}\right) p\left(r_{t-1}, \mathbf{x}_{1:t-1}\right) \\ &\overset{(d)}{=} \sum_{r_{t-1} \in \mathcal{R}_{t-1}} p\left(r_t | r_{t-1}\right) p\left(x_t | r_{t-1}, \mathbf{x}_{1:t-1}\right) p\left(r_{t-1}, \mathbf{x}_{1:t-1}\right). \end{aligned} \tag{4.3.2}$$

where (a) holds true using a marginalization, (b) and (c) hold true using two chain rules, (d) holds true thanks to the fact that r_t does not depend on $\mathbf{x}_{1:t-1}$ and x_t does not depend on r_t.

Thus, combining Equation (4.3.1) and Equation (4.3.2) we get:

$$p\left(r_t | \mathbf{x}_{1:t}\right) \propto \sum_{r_{t-1} \in \mathcal{R}_{t-1}} \underbrace{p\left(r_t | r_{t-1}\right)}_{\text{hazard}} \underbrace{p\left(x_t | r_{t-1}, \mathbf{x}_{1:t-1}\right)}_{\text{UPM}} p\left(r_{t-1} | \mathbf{x}_{1:t-1}\right). \tag{4.3.3}$$

So, given the previous runlength distribution $p\left(r_{t-1} | \mathbf{x}_{1:t-1}\right)$, one can thus build a *message-passing algorithm* (see figure 4.3.1) for the current run-length distribution $p(r_t | \mathbf{x}_{1:t})$ by calculating:

1. the underlying predictive model (UPM) $p\left(x_t | r_{t-1}, \mathbf{x}_{1:t-1}\right)$,
2. the hazard function $p\left(r_t | r_{t-1}\right)$.

It should be noted that at each time t, the runlength R_t either continues to grow (which corresponds to the event $\left\{r_t = r_{t-1}+1\right\}$) or a change occurs which corresponds to $\left\{r_t = 0\right\}$. Thus, from equation (4.3.3), we get the following recursive runlength distribution estimation:

- Growth probability:

$$p\left(r_t = r_{t-1} + 1 | \mathbf{x}_{1:t}\right) \propto p\left(r_t | r_{t-1}\right) p\left(x_t | r_{t-1}, \mathbf{x}_{1:t-1}\right) p\left(r_{t-1} | \mathbf{x}_{1:t-1}\right). \tag{4.3.4}$$

- Change-point probability:

$$p\left(r_t = 0 | \mathbf{x}_{1:t}\right) \propto \sum_{r_{t-1} \in \mathcal{R}_{t-1}} p\left(r_t | r_{t-1}\right) p\left(x_t | r_{t-1}, \mathbf{x}_{1:t-1}\right) p\left(r_{t-1} | \mathbf{x}_{1:t-1}\right). \tag{4.3.5}$$

Hazard function. According to Equation (4.3.4) and Equation (4.3.5), the runlength distribution estimation need to compute the change-point prior $p\left(r_t | r_{t-1}\right)$, which is done following the model in Equation 4.3.6:

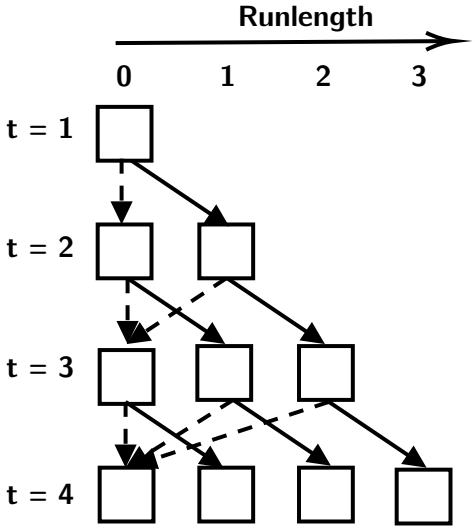


Figure 4.3.1: Message passing for runlength inference.
Boxes represent runlengths in the runlength distribution, and arrows represent messages that are passed between them in the update algorithm. At the top of the picture the runlength distribution is shown starting as a single runlength (here we initialise the algorithm such that we assume a change-point has happened, so the runlength distribution just contains the runlength of zero). After one time step the runlength distribution grows to two runlengths (zero and one). Either a change-point occurs and so the runlength remains at zero (shown by a dashed arrow representing the message from zero to zero) or a change didn't happen and thus the runlength increments to one (the message is represented by a full arrow). The picture shows how the runlength grows by one after each time step. The update step has a cost (both in time and space) linear in the number of possible runlengths.

$$p\left(r_t|r_{t-1}\right) = \begin{cases} H\left(r_{t-1}\right) & \text{if } r_t = 0 \\ 1 - H\left(r_{t-1}\right) & \text{if } r_t = r_{t-1} + 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.3.6}$$

with: $H(s) = \frac{\mathsf{P}_{\text{change}}\,(s+1)}{\sum_{t=s+1}^{\infty} \mathsf{P}_{\text{change}}(t)}$ and $\mathbf{P}_{\text{change}}$ denotes the probability distribution over the interval between changepoints.

A simple example of BOCPD would be to use a constant hazard function $h \in (0, 1)$ in the sense that $p(r_t = 0|r_{t-1})$ is independent of r_{t-1} and is constant, giving rise, a priori, to geometric inter-arrival times for change points ($\mathbf{P}_{\text{change}}(s + 1) = h\,(1 - h)^s$). Thus, the recursive runlength distribution computation becomes:

$$\begin{aligned} p(r_t \neq 0|\mathbf{x}_{1:t}) &\propto (1 - h)\, p(x_t|r_{t-1}, \mathbf{x}_{1:t-1}) p(r_{t-1}|\mathbf{x}_{1:t-1}) \\ p(r_t = 0|\mathbf{x}_{1:t}) &\propto h \sum_{r_{t-1} \in \mathcal{R}_{t-1}} p(x_t|r_{t-1}, \mathbf{x}_{1:t-1}) p(r_{t-1}|\mathbf{x}_{1:t-1}) \end{aligned}$$

Then, for Bernoulli observations ($x_t \sim \mathcal{B}\left(\mu_t\right)$) the underlying predictive distribution (UPM) can be set to the Laplace predictor.

Definition 4.3: Laplace predictor
The Laplace predictor $\mathrm{Lp}\left(x_{t+1}|\mathbf{x}_{s:t}\right)$ takes as input a sequence $\mathbf{x}_{s:t} \in \{0,1\}^{n_{s:t}}$ and predicts the value of the next observation $x_{t+1} \in \{0,1\}$ as follows:

$$\mathrm{Lp}\left(x_{t+1}|\mathbf{x}_{s:t}\right) := \begin{cases} \frac{\sum_{i=s}^{t} x_i + 1}{n_{s:t}+2} & \text{if } x_{t+1} = 1, \\ \frac{\sum_{i=s}^{t} (1-x_i) + 1}{n_{s:t}+2} & \text{if } x_{t+1} = 0, \end{cases}$$

where $\forall x \in \{0,1\}$ $\mathrm{Lp}\left(x|\emptyset\right) = \frac{1}{2}$ corresponds to the uniform prior given to the process generating θ_c.

Remark 2. *Laplace predictor is used as the estimator of the maximum likelihood with a uniform prior. It originates from the classical literature on universal codes and has standard robustness properties and Bayesian interpretation that make it of especial interest. Another variant is the Krichesky-Trofimov estimate [33].*

Although BOCPD algorithm is very efficient in practice, its analysis in term of false alarm rate and detection delay is still an open problem. As a first step of the analysis of the Bayesian Online Change Point Detection, in this section we reformulate it in terms of learning strategy based on a growing number of forecasters.

In the following, to simplify the derivations (especially for Lemmas 4.1 and 4.2) we assume that the hazard function for BOCPD is constant ($H\left(r_{t-1}\right) = h$). Otherwise, the statement of Lemmas 4.1 and 4.2 becomes cumbersome to write and difficult to understand.

4.3.2 Learning with a growing number of forecasters

Notion of forecaster.

Let $t \in \mathbb{N}^\star$ and $s \in [1, t]$. A forecaster s is a successive product of $(t-s)$ Laplace predictors $(\mathrm{Lp}\left(x_{t+1}|\mathbf{x}_{s:t}\right) \times \mathrm{Lp}\left(x_t|\mathbf{x}_{s:t-1}\right) \times \ldots \times \mathrm{Lp}\left(x_s|\emptyset\right))$ (see Definition 4.3), created at time s with some initial weight. At each time t, the forecaster s observes exactly the sequence $\mathbf{x}_{s:t}$ from the environment.

At each time t, each possible value of the runlength $r_t \in [0, t-1]$ corresponds to a specific forecaster. More specifically, the forecaster starting at time s corresponds at time t to the $t-s$ value of the runlength r_t.

Forecaster loss.

Using the Laplace predictor, the instantaneous loss of the forecaster s at time t is given by:

$$\begin{aligned} l_{s,t} &:= -\log \mathrm{Lp}\left(x_t|\mathbf{x}_{s:t-1}\right) \\ &= -x_t \log \mathrm{Lp}\left(1|\mathbf{x}_{s:t-1}\right) - (1-x_t) \log \mathrm{Lp}\left(0|\mathbf{x}_{s:t-1}\right). \end{aligned}$$

Then, let $\widehat{L}_{s:t} := \sum_{s'=s}^{t} l_{s':t}$ denotes the cumulative loss incurred by the forecaster s from time s until time t which takes the following crude expression:

$$\widehat{L}_{s:t} := \sum_{s'=s}^{t} -\log \mathrm{Lp}\left(x_t|\mathbf{x}_{s':t-1}\right) \tag{4.3.7}$$

Forecaster weights.

Instead of dealing with the posterior distribution of the runlength r_t, we propose to give to each forecaster s a weight $v_{s,t} := p(r_t = t-s|\mathbf{x}_{s:t})$ according to its sequence of observations $\mathbf{x}_{s:t}$ (see Figure 4.3.2). By this way, we describe the novel formulation of the Bayesian Online Change Point Detector in Algorithm 4.1. Notice that in line 5, Algorithm 4.1 performs a change-point detection, which was not present in [2].

Algorithm 4.1 BOCPD [46]

Require: $h \in (0, 1)$

1: $v_{1,1} \leftarrow 1$
2: **for** $t = 1, \ldots$ **do**
3: Observe $x_t \sim \mathcal{B}(\mu_t)$
4: Define for each forecaster s up to time t:

$$v_{s,t} \leftarrow \begin{cases} (1-h)\exp\left(-l_{s,t}\right) v_{s,t-1} & \forall s < t, \\ h\sum_{i=1}^{t-1}\exp\left(-l_{i,t}\right) v_{i,t-1} & s = t\,. \end{cases} \tag{4.3.8}$$

5: Estimate the last change-point $\widehat{\tau}_t$ such that: $\widehat{\tau}_t \leftarrow \underset{s\in[1,t]}{\operatorname{argmax}}\, v_{s,t}$.
6: **end for**

Equation (4.3.8) defines the weights $v_{s,t}$ recursively. Lemma 4.1 expands the expression of $v_{s,t}$ for a better way to handle these quantities.

Lemma 4.1: From recursive to closed-form expressions
Let: $V_t = \sum_{s=1}^{t} v_{s,t}$. Then, by noticing that $V_t = \sum_{s=1}^{t-1}\exp\left(-l_{s,t}\right) v_{s,t-1}$, the quantities $v_{s,t}$ take the following alternative closed-form expression:

$$v_{s,t} = \begin{cases} (1-h)^{t-s+1}\, h^{1\{s\neq 1\}} \exp\left(-\widehat{L}_{s:t}\right) V_s & \forall s < t, \\ hV_t & s = t. \end{cases}$$

Proof. You only need to see that:

$$\begin{aligned} V_t &= \sum_{s=1}^{t} v_{s,t} \\ &= \sum_{s=1}^{t-1} v_{s,t} + v_{t,t} \\ &= (1-h)\sum_{s=1}^{t-1}\exp\left(-l_{s,t}\right) v_{s,t-1} + h\sum_{s=1}^{t-1}\exp\left(-l_{s,t}\right) v_{s,t-1} \\ &= \sum_{s=1}^{t-1}\exp\left(-l_{s,t}\right) v_{s,t-1}. \end{aligned}$$

□

First, from Lemma 4.1 one should notice that the quantity V_t plays the role of an initial weight that is given to the forecaster newly created at time t. Thus, in order to control the quantities $v_{t,s}$, we need to explicitly expand the expression of V_t.

The expression for V_t is given iteratively (see Lemma 4.1). Making it explicit reveals the power of the strategy introduced by [46], that combines the updates of exponentially many forecasters into a simple iterative scheme. Indeed, Lemma 4.2 gives us the explicit expression of V_t.

Lemma 4.2: Computing the initial weight V_t
The initial weight V_t takes the following form:

$$V_t = (1-h)^{t-2} \sum_{k=1}^{t-1} \left(\frac{h}{1-h}\right)^{k-1} \tilde{V}_{k:t} \quad \text{where:}$$

$$\tilde{V}_{k:t} = \sum_{i_1=1}^{t-k} \sum_{i_2=i_1+1}^{t-(k-1)} \cdots \sum_{i_{k-1}=i_{k-2}+1}^{t-2} \underbrace{\exp\left(-\widehat{L}_{1:i_1}\right)}_{} \times \prod_{j=1}^{k-2} \exp\left(-\widehat{L}_{i_j+1:i_{j+1}}\right) \exp\left(-\widehat{L}_{i_{k-1}+1:t-1}\right).$$

Proof. First, for all $t \geqslant 2$, we have:

$$V_t = \sum_{i=1}^{t} v_{i,t}$$

$$V_t = v_{1,t} + \sum_{i=2}^{t-1} v_{i,t} + v_{t,t}$$

$$V_t = (1-h)^{t-1} \exp\left(-\widehat{L}_{1:t}\right) V_1 + \sum_{i=2}^{t-1} (1-h)^{t-i} \exp\left(-\widehat{L}_{i:t}\right) h V_i + h V_t. \quad \text{(using Lemma 4.1)}$$

$$\Leftrightarrow V_t = \sum_{i=1}^{t} \underbrace{(1-h)^{t-i} \exp\left(-\widehat{L}_{i:t}\right) h^{1(i\neq 1)}}_{\alpha_{t,i}} V_i \text{ with convention: } \widehat{L}_{i,j} = 0 \Leftrightarrow i > j.$$

$$\Leftrightarrow V_t = \sum_{i=1}^{t} \alpha_{t,i} V_i.$$

$$\Leftrightarrow (1-\alpha_{t,t})\, V_t = \sum_{i=1}^{t-1} \alpha_{t,i} V_i.$$

Finally, by letting $\beta_{t,i} = \frac{\alpha_{t,i}}{1-h}$, we obtain the following expression of V_t (using the classical induction procedure and using $V_1 = 1$):

$$\begin{aligned}
&\forall t \geqslant 4,\\
&V_t = \left(\beta_{t,1} + \sum_{i_1=1}^{t-2} \beta_{t,t-i_1}\beta_{t-i_1,1} + \sum_{k=3}^{t-1}\sum_{i_1=1}^{t-k}\sum_{i_2=i_1+1}^{t-(k-1)} \cdots \sum_{i_{k-1}=i_{k-2}+1}^{t-2} \beta_{t,t-i_1}\beta_{t-i_1,t-i_2}\cdots\beta_{t-i_{k-1},1}\right)V_1\\
&\quad = \beta_{t,1} + \sum_{i_1=1}^{t-2} \beta_{t,t-i_1}\beta_{t-i_1,1} + \sum_{k=3}^{t-1}\sum_{i_1=1}^{t-k}\sum_{i_2=i_1+1}^{t-(k-1)} \cdots \sum_{i_{k-1}=i_{k-2}+1}^{t-2} \beta_{t,t-i_1}\beta_{t-i_1,t-i_2}\cdots\beta_{t-i_{k-1},1}.\\
&V_3 = \beta_{3,1} + \beta_{3,2}\beta_{2,1}.\\
&V_2 = \beta_{2,1}.
\end{aligned}$$

which can be concatenated in the following form:

$$V_t = (1-h)^{t-2}\sum_{k=1}^{t-1}\left(\frac{h}{1-h}\right)^{k-1}\tilde{V}_{k:t}, \quad \text{where:}$$

$$\tilde{V}_{k:t} = \overbrace{\sum_{i_1=1}^{t-k}\sum_{i_2=i_1+1}^{t-(k-1)} \cdots \sum_{i_{k-1}=i_{k-2}+1}^{t-2}}^{\binom{t-2}{k-1}} \exp\left(-\widehat{L}_{1:i_1}\right) \times \prod_{j=1}^{k-2}\exp\left(-\widehat{L}_{i_j+1:i_{j+1}}\right)\exp\left(-\widehat{L}_{i_{k-1}+1:t-1}\right),$$

$$\text{and } (1-h)^{t-2}\sum_{k=1}^{t-1}\left(\frac{h}{1-h}\right)^{k-1}\binom{t-2}{k-1} = 1.$$

□

On the other hand, dealing with the explicit expression of V_t is challenging from a theoretical standpoint (proving performance guarantees), since we need to control a combinatorial number of cumulative losses. Indeed, notice that:

$$\sum_{i_1=1}^{t-k}\sum_{i_2=i_1+1}^{t-(k-1)} \cdots \sum_{i_{k-1}=i_{k-2}+1}^{t-2} 1 = \binom{t-2}{k-1},$$

where $\binom{\bullet}{\bullet}$ stands for the binomial operator.

4.4 The Restarted Bayesian Online Change Point Detector algorithm (R-BOCPD)

In this section, we introduce a pruning version of the original BOCPD which is built on a *novel initial weight function*, a *restart procedure to prune the useless experts* and a *well tuned hyper-parameter* instead of the hazard function.

4.4.1 Introducing a simple initial weight.

In order to avoid the difficulty mentioned in Lemma 4.2, we propose to use a much simpler initial weight that takes the following form:

$$\mathcal{V}_{r:s-1} := \exp\left(-\widehat{L}_{r:s-1}\right) \text{ for some starting time } r.$$

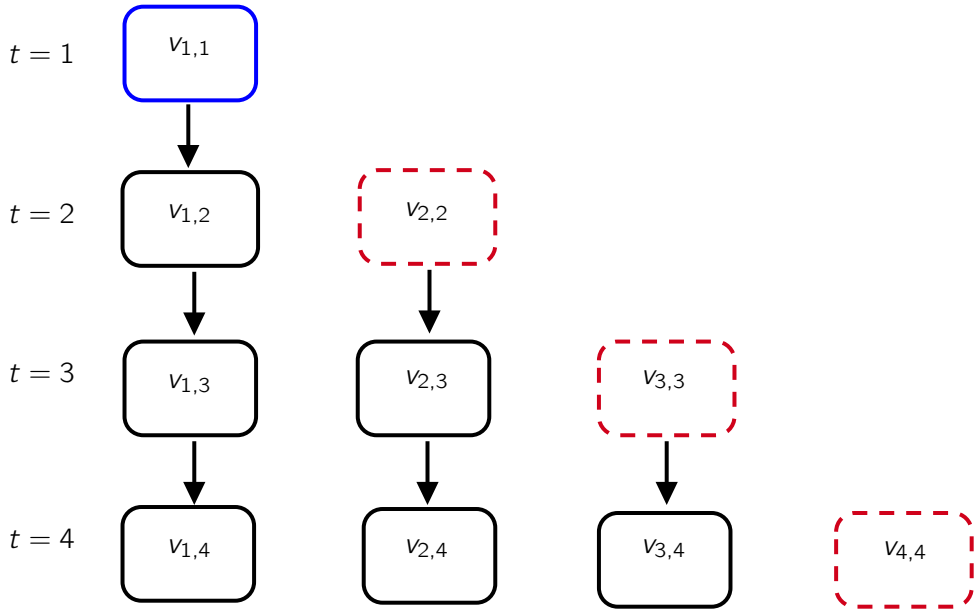


Figure 4.3.2: Updating the forecaster distribution.
Rounded rectangles represent the forecasters in the forecaster distribution, arrows represent the recursive forecaster distribution update formulated in Equation 4.3.8 and red dashed rounded rectangles represent the forecaster newly created at each time step t.

Notice that the initial weight $\mathcal{V}_{r:s-1}$ is a restricted version of the original one V_s by forgetting the contribution of all forecasters but the one launched at the starting time r (underlined term in Lemma 4.2). Thereby, the control of the initial weight is made easier: instead of dealing with a combinatorial number of cumulative losses, we only need to control **one** cumulative loss ($\widehat{L}_{r:s-1}$). Thus, for some starting time r, we denote by $\vartheta_{r,s,t}$ the novel weight given to the forecaster $s \geqslant r$ at time $t \geqslant s$. Then, one should also notice that BOCPD (Algorithm 4.1) produces an estimation of the last change point at each time step. In order to analyze this algorithm in term of detection delay, we propose to introduce a change-point decision rule (restart procedure).

4.4.2 Introducing a restart procedure.

For any starting time $r \leqslant t$, the change-point criterion is written as follows:

$$\texttt{Restart}_{r:t} = \mathbf{1}\{\exists s \in (r, t] : \vartheta_{r,s,t} > \vartheta_{r,r,t}\} \tag{4.4.1}$$

where $\vartheta_{r,s,t}$ denotes the weight of the forecaster s created with the initial weight $\mathcal{V}_{r:t_{s-1}}$ at time t (see Algorithm 4.2 line 4). The intuition behind the criterion $\texttt{Restart}_{r:t}$ is that at each time $t < \tau$ where there is no change, the forecaster distribution tends to be concentrated around the forecaster launched at the starting time r. So, if the distribution $\vartheta_{r,s,t}$ undergoes a change then it can be seen as a certain change-point that has appeared. Thereby when $\texttt{Restart}_{r:t} = 1$, a change-point is detected and thus we **restart** a new forecaster at time $r = t + 1$ and **delete** all previous launched forecasters ($s < r$). This can be seen as a sophisticated pruning out procedure to reduce the number of launched forecasters.

Finally, by using the hyper-parameter $\eta_{r,s,t}$ (instead of the constant value h) and plugging the initial weight $\mathcal{V}_{r:t-1}$ and the decision rule $\texttt{Restart}_{r:t}$ into the formalism of BOCPD (Equation

4.3.8), we obtain a restarted version of the Bayesian Online Change Point Detector which is described in Algorithm 4.2.

Algorithm 4.2 R-BOCPD

Require: $(\eta_{r,s,t})_{r\geqslant 1, s\geqslant 1, t\geqslant 1} \in (0,1)$

$r \leftarrow 1$, $\vartheta_{r,1,1} \leftarrow 1$, $\eta_{r,1,1} \leftarrow 1$.

for $t = 1, \ldots$ **do**

  Observe $x_t \sim \mathcal{B}(\mu_t)$

  Define for each forecaster s from time r to time t:

$$\vartheta_{r,s,t} \leftarrow \begin{cases} \frac{\eta_{r,s,t}}{\eta_{r,s,t-1}} \exp(-l_{s,t})\, \vartheta_{r,s,t-1} & \forall s < t, \\ \eta_{r,t,t} \times \mathcal{V}_{r:t-1} & s = t\,. \end{cases} \tag{4.4.2}$$

  if $\texttt{Restart}_{r:t} = 1$ **then** $r \leftarrow t+1$, $\vartheta_{r,r,r} \leftarrow 1$, $\eta_{r,r,r} \leftarrow 1$.

  Estimate the last change-point: $\widehat{\tau}_t \leftarrow r$.

end for

4.4.3 Discussion about R-BOCPD

The main difference between the R-BOCPD and its previous version lie primarily in the use of the test $\texttt{Restart}_{r:t} = 1$ for detecting the change-points. The second difference is the use of a simple initial weight $\mathcal{V}_{r:t-1}$ instead of the quantity V_t standing for the sum of the forecaster weights at time t. This is essentially done for theoretical reasons (see Lemma 4.2). The third difference is the use of a hyper-parameter $\eta_{r,s,t}$ instead of the hazard function h. The quantity $\frac{\eta_{r,s,t}}{\eta_{r,s,t-1}} \underset{t\to\infty}{\to} 1$ is used in updating the forecaster distribution $\vartheta_{r,s,t}$ at time t instead of the quantity $(1-h)$ whose asymptotic behavior is the same for an harazd rate very small. Indeed, $1 - h \approx 1$. Finally, unlike [2, 46] where the hazard function is assumed to be known (see section 4.3.1), the function $\eta_{r,s,t}$ will be specified thanks to the analysis in section 4.5.1.

4.5 Performance guarantees

In this section, we build the two main guarantees for the R-BOCPD algorithm, namely: the false alarm rate (Theorem 4.10) and the detection delay control (Theorem 4.11). Then, we provide the reader with some useful tools to build these guarantees.

4.5.1 Non-asymptotic analysis of R-BOCPD

Let r denotes the time of the last restart and τ_c the change-point just coming after r.

Theorem 4.10 states the condition on $\eta_{r,s,t}$ where R-BOCPD (algorithm 4.2) does not make any false alarm with high probability. It is without reminding that the false alarm corresponds to the event where $\texttt{Restart}_{r:t} = 1$ during the period $[r, \tau_c)$.

Theorem 4.10: False alarm rate
Assume that $\mathbf{x}_{r:t} \sim \mathcal{B}(\theta)^{\otimes n_{r:t}}$. Let: $\alpha > 1$. If $\eta_{r,s,t}$ is small enough such that:

$$\forall t \in [r, \tau_c), s \in (r, t] : \eta_{r,s,t} < \frac{\sqrt{n_{r:s-1} \times n_{s:t}}}{10\,(n_{r:t}+1)} \times \left(\frac{\log(4\alpha+2)\delta^2}{4 n_{r:t} \log((\alpha+3)\, n_{r:t})} \right)^{\alpha}$$

then, with probability higher than $1-\delta$, no false alarm occurs on the interval $[r, \tau_c)$:

$$\mathbb{P}_{\theta}\Big(\exists\ t \in [r, \tau_c) : \texttt{Restart}_{r:t} = 1 \Big) \leqslant \delta.$$

Proof. The complete proof is given in section 9.5. □

Before stating the control of the detection delay, we need to introduce the notion of relative gap $\Delta_{r,s,t}$.

Definition 4.4: Relative gap $\Delta_{r,s,t}$
Let $\Delta \in [0,1]$. The relative gap $\Delta_{r,s,t}$ for the forecaster s at time t takes the following form (depending on the position of s):

$$\Delta_{r,s,t} = \left(\frac{n_{r:\tau_c-1}}{n_{r:s-1}} \mathbf{1}\{\tau_c \leqslant s \leqslant t\} + \frac{n_{\tau_c:t}}{n_{s:t}} \mathbf{1}\{s < \tau_c\} \right) \Delta.$$

Theorem 4.11 states the detection delay under some condition on the quantity $\eta_{r,s,t}$.

Theorem 4.11: Detection delay
Let $\mathbf{x}_{r:\tau_c-1} \sim \mathcal{B}(\theta_1)^{\otimes n_{r:\tau_c-1}}$, $\mathbf{x}_{\tau_c:t} \sim \mathcal{B}(\theta_2)^{\otimes n_{\tau_c:t}}$ and $\Delta = |\theta_1 - \theta_2|$: the change-point gap. Then, let: $f_{r,s,t} = \log n_{r:s} + \log n_{s:t+1} - \frac{1}{2}\log n_{r:t} + \frac{9}{8}$.
If $\eta_{r,s,t}$ is large enough such that:

$$\eta_{r,s,t} > \exp\big(-2 n_{r,s-1} \left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right)^2 + f_{r,s,t} \big),$$

then, the change-point τ_c is detected (with a probability at least $1-\delta$) with a delay not exceeding $\mathfrak{D}_{\Delta,r,\tau_c}$, such that:

$$\mathfrak{D}_{\Delta,r,\tau_c} = \min\left\{ d \in \mathbb{N}^{\star} : d > \frac{\left(1 - \frac{\mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta}}{\Delta}\right)^{-2}}{2\Delta^2} \times \frac{-\log \eta_{r,\tau_c,d+\tau_c-1} + f_{r,\tau_c,d+\tau_c-1}}{1 + \frac{\log \eta_{r,\tau_c,d+\tau_c-1} - f_{r,\tau_c,d+\tau_c-1}}{2 n_{r,\tau_c-1}(\Delta - \mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta})^2}} \right\}, \tag{4.5.1}$$

where:

$$\mathcal{C}_{r,s,t,\delta} = \frac{\sqrt{2}}{2}\left(\sqrt{\frac{1+\frac{1}{n_{r:s-1}}}{n_{r:s-1}} \log\left(\frac{2\sqrt{n_{r:s}}}{\delta}\right)} + \sqrt{\frac{1+\frac{1}{n_{s:t}}}{n_{s:t}} \log\left(\frac{2 n_{r:t}\sqrt{n_{s:t}+1}\log^2(n_{r:t})}{\log(2)\delta}\right)} \right). \tag{4.5.2}$$

Proof. The complete proof is given in section 9.6. □

Discussion about the detection delay $\mathfrak{D}_{\Delta,r,\tau_c}$ From Eq.(4.5.1), for a fixed r and τ_c we notice that the larger the change-point gap Δ, the smaller the detection delay $\mathfrak{D}_{\Delta,r,\tau_c}$ and vice versa. Moreover for a fixed gap Δ, the larger n_{r,τ_c-1}: the number of observations before the change-point τ_c, the smaller the detection delay $\mathfrak{D}_{\Delta,r,\tau_c}$ (cf figure 4.5.1).

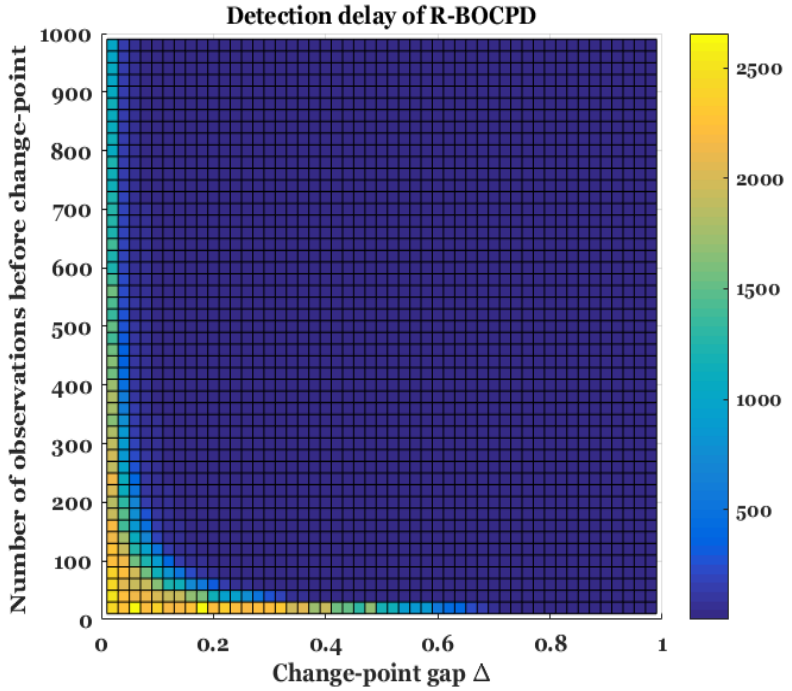


Figure 4.5.1: Variation of the R-BOCPD detection delay $\mathfrak{D}_{\Delta,r,\tau_c}$ as a function of the change point gap Δ (x-axis) and the number of observations before the change-point n_{r,τ_c-1} (y-axis). For this plot, we choose $\eta_{r,s,t} = \frac{1}{n_{r:t}}$ for R-BOCPD.

Remark 3 (Minimum detectable gap $\Delta_{\min}(r,\tau_c,t)$)**.** *Instead of imposing a condition on the lower-bound of $\eta_{r,s,t}$, we can discuss the detectability of the change-point τ_c according to the magnitude of the gap Δ. Thus, if the gap Δ is of magnitude at least $\Delta_{min}(r,\tau_c,t) = \sqrt{\frac{-\log \eta_{r,\tau_c,t} + f_{r,\tau_c,t}}{n_{r:\tau_c-1}}} + \mathcal{C}_{r,\tau_c,t,\delta}$, then the change-point τ_c is detected (with a probability at least $1-\delta$) with a finite delay not exceeding $\mathfrak{D}_{\Delta,r,\tau_c}$.*

Discussion about the asymptotic optimality We compare the result of Theorem 4.11 with the existing lower-bound on the detection delay (see [93]). The asymptotic regime corresponds to the case where the elapsed time between the last restart r and the new change point τ_c tends to infinity, while the probability of false alarm δ tends to zero. More formally, the asymptotic regime is reached when $\frac{n_{r:\tau_c-1}}{\log(1/\delta)} \to \infty$, and $\log n_{r:\tau_c-1} = o\left(\log \frac{1}{\delta}\right)$ when $\delta \to 0$. We obtain that:

$$\mathfrak{D}_{\Delta,r,\tau_c} \underset{\tau_c\to\infty}{\to} \frac{-\log \eta_{r,\tau_c,d+\tau_c-1} + o\left(\log \frac{1}{\delta}\right)}{2\left|\theta_2 - \theta_1\right|^2}. \tag{4.5.3}$$

Thus, following Theorem 4.9, the detection delay $\mathfrak{D}_{\Delta,r,\tau_c}$ is **asymptotically order optimal** (up to the Pinsker inequality tightness relating $|\theta_2-\theta_1|^2$ to $\mathbf{kl}(\theta_2,\theta_1)$). Moreover, the smaller $\eta_{r,s,t}$, the larger the detection delay and vice-versa. Also, following Remark 3, the smaller $\eta_{r,s,t}$, the larger the minimum detectable gap $\Delta_{\min}(r,\tau_c,t)$.

Remark 4 (Main difficulty to get optimality with the Kullback-Leibler divergence)**.** *The result of Equation 4.5.3 shows the Euclidean distance $|\theta_2-\theta_1|^2$ instead of the Kullback-Leibler divergence $\mathbf{kl}(\theta_2,\theta_1)$ as expected from Theorem 4.9. Indeed, in the analysis of the detection delay (see section 9), the quantity $n_{r:s-1}\mathbf{kl}(\widehat{\mu}_{r:s-1},\widehat{\mu}_{r:t})+n_{s:t}\mathbf{kl}(\widehat{\mu}_{s:t},\widehat{\mu}_{r:t})$ appears (it naturally comes from Lemma 4.3). Building a lower bound of $n_{r:s-1}\mathbf{kl}(\widehat{\mu}_{r:s-1},\widehat{\mu}_{r:t})+n_{s:t}\mathbf{kl}(\widehat{\mu}_{s:t},\widehat{\mu}_{r:t})$ showing the quantity $\mathbf{kl}(\theta_2,\theta_1)$ (in the case where there is change point $\tau\in[r,t]$) does not seem trivial. Thus, we have opted to use the Pinsker inequality which has slightly reduced the optimality of our result.*

Discussion about the choice of $\eta_{r,s,t}$. Choosing $\eta_{r,s,t}\approx\frac{1}{n_{r:t}}$ seems to be a wise choice since it allows us to verify the conditions of Theorem 4.10 (for some $\alpha\to 1$) and Theorem 4.11. Thus, by plugging $\eta_{r,s,t}\approx\frac{1}{n_{r:t}}$ into the asymptotic expression of the detection delay (Equation 4.5.3), we get (in the asymptotic regime $\frac{n_{r:\tau_c-1}}{\log(1/\delta)}\to\infty$, and $\log n_{r:\tau_c-1}=o\Big(\log\frac{1}{\delta}\Big)$ when $\delta\to 0$):

$$\mathfrak{D}_{|\theta_2-\theta_1|,r,\tau_c}\underset{\tau_c\to\infty}{\to}\frac{o\Big(\log\frac{1}{\delta}\Big)}{2\,|\theta_2-\theta_1|^2}.$$

By this way, the detection delay is **asymptotically optimal** in the sense of Theorem 4.9 (up to the Pinsker inequality tightness relating $|\theta_2-\theta_1|^2$ to $\mathbf{kl}(\theta_2,\theta_1)$).

4.5.2 Sketch of proof for the false alarm rate control and the detection delay

In this section, we provide the key elements and the essential intuitions to build the false alarm guarantee and the detection delay control. The key element in building the false alarm guarantee and the detection delay of R-BOCPD lies in controlling *efficiently* the quantities $\log\vartheta_{r,s,t}$. Indeed using Equation (4.4.2), we get (for some starting time r):

$$\log\vartheta_{r,s,t}=\log\eta_{r,s,t}\times\mathbf{1}\{s\neq r\}-\widehat{L}_{r:s-1}-\widehat{L}_{s:t}. \tag{4.5.4}$$

Then, it is clear that controlling the quantity $\log\vartheta_{r,s,t}$ requires an efficient control of $\widehat{L}_{s:t}$. Using the crude expression of $\widehat{L}_{s:t}$ (see Equation (4.3.7)) seems to be very naive in the sense that we need to control each instantaneous loss $l_{s,t}$ independently without taking into account the dependencies between $l_{s,t}$ and $l_{s,t-1}$. A smarter way to deal with the quantity $\widehat{L}_{s:t}$ lies in writing it as follows:

Lemma 4.3: Rewriting the cumulative loss
Based on the Laplace predictor, the cumulative loss $\widehat{L}_{s:t}$ takes the following closed-form expression:

$$\forall \mathbf{x}_{s:t} \in \{0,1\}^{n_{s:t}} \quad \widehat{L}_{s:t} = \log\left(n_{s:t}+1\right) + \log\binom{n_{s:t}}{\sum_{i=s}^{t} x_i}.$$

Proof. The complete proof is given in section 9.1. □

Remark 5. *The main idea of Lemma 4.3 is taken from the book "Prediction, Learning and Games" by [34]. Notice that Lemma 4.3 is valid for any binary sequence $\mathbf{x}_{s:t} \in \{0,1\}^{n_{s:t}}$. No assumption on the intrinsic distribution of the sequence $\mathbf{x}_{s:t}$ is required.*

Lemma 4.4: Cumulative loss control before a change-point
Assume that we observe a sequence $\mathbf{x}_{s:t} \sim \mathcal{B}(\theta)^{\otimes n_{s:t}}$. Then, the control of the quantity $\widehat{L}_{s:t}$ is done as follows:

- Upper bound:

$$\widehat{L}_{s:t} \leqslant \log n_{s:t+1} - \sum_{i=s}^{t} x_i \log\theta - \sum_{i=s}^{t}(1-x_i)\log\left(1-\theta\right),$$

- Lower bound:

$$\widehat{L}_{s:t} \geqslant -\sum_{i=s}^{t} x_i \log\theta - \sum_{i=s}^{t}(1-x_i)\log\left(1-\theta\right) + \log\frac{n_{s:t+1}}{\sqrt{n_{s:t}}} - n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t},\theta\right) - \frac{9}{8}.$$

Proof. The complete proof is given in section 9.2. □

Remark 6. *Notice how tight are the upper and lower bound of the loss $\widehat{L}_{s:t}$. The control in Lemma 4.4 represents an essential element to provide the false alarm guarantee in Theorem 4.10.*

Finally, one should notice that the lower bound of the cumulative loss $\widehat{L}_{s:t}$ involves the Kullback-Leibler divergence $\mathbf{kl}\left(\widehat{\mu}_{s:t},\theta\right)$. For very tight control of the cumulative loss, we need to efficiently control the quantity $\mathbf{kl}\left(\widehat{\mu}_{s:t},\theta\right)$. This is done uniformly in Lemma 4.5 and Lemma 4.6.

Lemma 4.5: Time uniform $\mathbf{kl}(\bullet,\bullet)$ concentration
Let: $\widehat{\mu}_t$ denotes the empirical mean over the sequence $x_1,\ldots,x_t \sim \mathcal{B}(\theta)^{\otimes n_{1:t}}$, then for all $(\delta,\alpha) \in (0,1)\times(1,\infty)$ we have:

$$\mathbb{P}_\theta\left(\exists t \in \mathbb{N}^\star : \mathbf{kl}\left(\widehat{\mu}_t,\theta\right) \geqslant \frac{\alpha}{t}\log\frac{\log(\alpha t)\log(t)}{\log^2(\alpha)\delta}\right) < \delta.$$

Proof. The interested reader can refer for more details on the proof of Lemma 4.5 to the manuscript untitled "Mathematics of Statistical Sequential Decision Making" https://pdfs.semanticscholar.org/9099/c0f71185adce7705beb78d595abc817c33d6.pdf. □

Lemma 4.6: Doubly-time uniform kl $(\bullet, \bullet)$ concentration
Let $\widehat{\mu}_{s:t}$ denotes the empirical mean over the sequence $(x_s, ..., x_t) \sim \mathcal{B}(\theta)^{\otimes n_{s:t}}$, then for all $(\delta, \alpha) \in (0,1) \times (1, \infty)$ we have:

$$\mathbb{P}_\theta \left(\exists t \in \mathbb{N}^\star, \exists s \in (r, t] : \mathbf{kl}\left(\widehat{\mu}_{s:t}, \theta\right) \geqslant \frac{\alpha}{n_{s:t}} \log \frac{n_{r:t} \log^2(n_{r:t}) \log((\alpha+1)\, n_{s:t})}{\log(2) \log^2(\alpha) \delta} \right) < \delta.$$

Proof. The interested reader can refer for more details on the proof of Lemma 4.6 to the manuscript untitled "Mathematics of Statistical Sequential Decision Making" https://pdfs.semanticscholar.org/9099/c0f71185adce7705beb78d595abc817c33d6.pdf. □

Then, in order to build the detection delay guarantee, we will need to efficiently control the quantity $|\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}|$ (which is related to $\widehat{L}_{r:s-1} + \widehat{L}_{s:t} - \widehat{L}_{r:t}$ via Pinsker inequality). This is done thanks to Lemma 4.7.

Lemma 4.7: Doubly-time uniform concentration for scan statistics
Let: $(x_r, ...x_t)$ be a sequence of independent binary random variables sampled from a Bernoulli distribution and $\widehat{\mu}_{i:j}$ the empirical mean over the sequence $\mathbf{x}_{i:j}$. Then, for all $(r, \delta) \in \mathbb{N}^\star \times (0,1)$, we get the following control:

$$\mathbb{P}\Big(\exists\ t > r, s \in [r, t) : \left| \widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t} - \mathbb{E}\left[\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t} \right] \right| \geqslant \mathcal{C}_{r,s,t,\delta} \Big) \leqslant \delta,$$

where: $\mathcal{C}_{r,s,t,\delta}$ is defined in Equation (4.5.2).

Proof. The complete proof is given in chapter 9. It is mainly built using the Laplace method of integration for optimization). □

4.6 Numerical illustrations of R-BOCPD against the state-of-art

In this section, we provide numerical comparisons between the proposed strategy R-BOCPD and two state-of-art strategies: BOCPD and the Improved GLR [102] in two different schemes, a first comparison on synthetic data and a second comparison on real world data.

4.6.1 Synthetic data

Comparison with BOCPD

Highlighting the use of the function $\mathcal{V}_{r:t-1}$ In order to highlight the use of the function $\mathcal{V}_{r:t-1}$ as initial weight given to the forecaster newly created at time t (instead of the original one V_t), we compare the R-BOCPD strategy against its previous version BOCPD

in the following experimental setting. We generate 2500 trajectories (sequences) of length $T = 5000$ where we vary the number of observation before the change-point from 10 to 1000 and we vary the change-point gap Δ from 0.01 to 1.

Then, we run R-BOCPD and BOCPD strategy on the same sequence 600 times. Finally, we plot the mean detection delay difference between R-BOCPD and BOCPD. Each square corresponds to a detection event for a change-point τ_c. The y coordinate corresponds to the number of observations both R-BOCPD and BOCPD algorithms received before the change-point, the x coordinate is the gap of the change-point. From figure 4.6.1, we see that the detection delay of R-BOCPD is slightly smaller than the one of BOCPD. Indeed, the detection delay difference is negative over all the experiments. By the way, using the function $\mathcal{V}_{r:t-1}$ instead of V_t as an initial weight given to the forecaster newly created allows us to speed up the change-point detection.

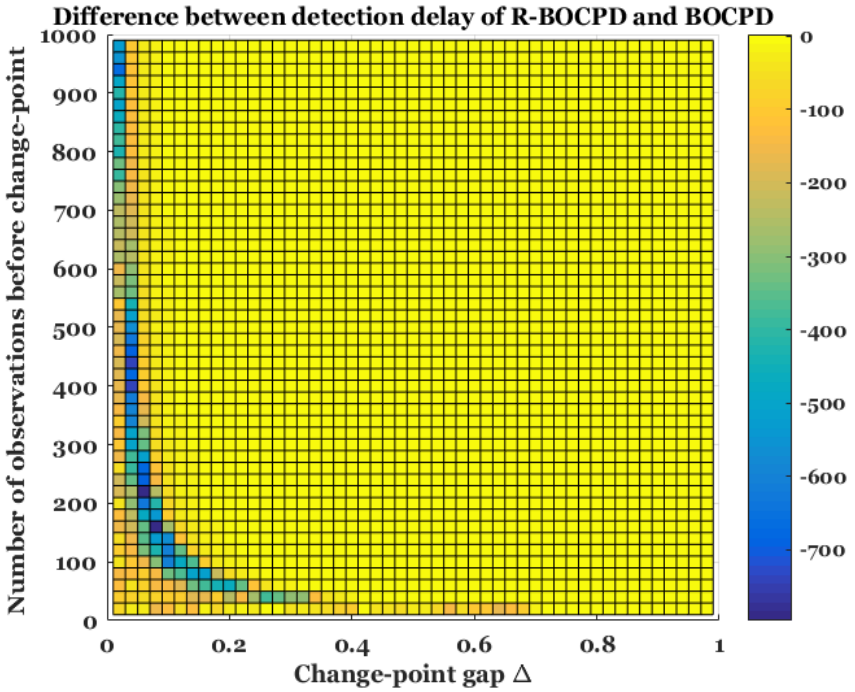


Figure 4.6.1: Difference between detection delays of R-BOCPD and BOCPD. In these experiments, we choose $\eta_{r,s,t} = \frac{1}{n_{r:t}}$ for R-BOCPD and $h = 1/T$ for BOCPD.

Highlighting the use of the restart procedure $\texttt{Restart}_{r:t}$ In order to highlight the benefit of using the restart procedure $\texttt{Restart}_{r:t}$ in R-BOCPD, we compare R-BOCPD strategy against BOCPD in the following setting. We generate a piece-wise stationary Bernoulli process with four change-points observed at time $(\tau_1 = 1, \tau_2 = 301, \tau_3 = 701, \tau_4 = 1051)$, then we run R-BOCPD and BOCPD on this environment and finally we plot (in figure 4.6.2) the change-point estimation $\widehat{\tau}_t$ for both R-BOCPD and BOCPD.

From figure 4.6.2, the curves of R-BOCPD and BOCPD are almost the same meaning that there is no significant difference in terms of false alarm and detection delays for both algorithms. Thus, restart procedure $\texttt{Restart}_{r:t}$ doesn't affect the performances of the Bayesian online change-point detector.

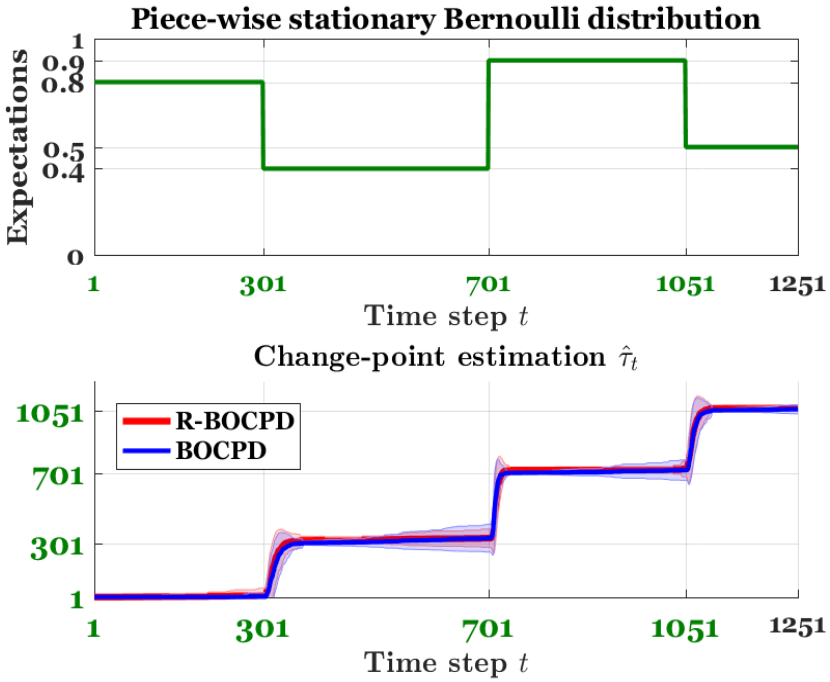


Figure 4.6.2: In all the experiment, we choose $\eta_{r,s,t} = \frac{1}{n_{r:t}}$ for R-BOCPD and $h = 3/1200$ for BOCPD. The curves are averaged over 300 runs. (Their error bars are also plotted).

Comparison with the Improved GLR

Recently, the classical Generalized Likelihood Ratio (GLR) strategy has been improved by [102] by well-tuning the decision threshold. It used a novel sharp concentration inequality based on the Laplace method for scan-statistics which holds doubly uniformly in time (see Lemma 4.7). The final formulation of the Improved GLR strategy for Bernoulli processes takes the following form:

$$\mathrm{GLR}_{r:t} = \mathbf{1}\{\exists\, s \in [r, t) : \left|\widehat{\mu}_{r:s} - \widehat{\mu}_{s+1:t}\right| \geqslant \mathcal{C}_{r,s,t,\delta}\}$$

where $\mathcal{C}_{r,s,t,\delta}$ is defined in Equation (4.5.2).

Lemma 4.8: Guarantees of the Improved GLR test

Let: $x_r, ..., x_{\tau-1} \overset{iid}{\sim} \mathcal{B}(\theta_1)^{\otimes n_{r:\tau-1}}$ be a sequence of $\tau - r$ i.i.d binary random variables following a Bernoulli distribution of expectation θ_1 and $x_\tau, ..., x_t \overset{iid}{\sim} \mathcal{B}(\theta_2)^{\otimes n_{\tau:t}}$ be a sequence of $t-\tau+1$ i.i.d binary random variables following a Bernoulli distribution of mean θ_2. Let: $\Delta = |\theta_1 - \theta_2|$ denotes the gap. Then, $\forall\, \delta \in (0,1)$, the GLR test started at time r and using the threshold $\mathcal{C}_{r,s,t,\delta}$ for each $t \geqslant r$ presents the three following guarantees:

1. **False alarm:** $\mathbb{P}_{\theta_1}\Big(\exists\, t \in [r,\tau) : \mathrm{GLR}_{r:t} = 1\Big) \leqslant \delta$
2. **Detection delay:**
   $\forall\, \varepsilon \in [0,1]\,,\ \forall t\ \in [\tau, \tau + \ell_{\varepsilon,\delta}(\tau - r, \Delta)]\ \ \mathbb{P}_{\theta_1,\theta_2}\Big(\mathrm{GLR}_{r:t} = 0\Big) \leqslant \delta_\varepsilon(t-r)$
   where:
   $$\begin{cases} \ell_{\varepsilon,\delta}(n,\Delta) = \min\left\{ d \in \mathbb{N}^\star : d > \dfrac{\frac{1}{2}(1+\varepsilon)^2 \times \frac{n+1}{n} \log(z_d/\delta)}{\left[\Delta^2 - \frac{1/2(1+\varepsilon)^2}{n} \log(z_d/\delta)\right]^+} \right\} \\ z_d = (d+\tau-r-1)\sqrt{d+\tau-r+1} \text{ and } [x]^+ = \max\{x, 0\} \\ \delta_\varepsilon(n) = 2n\left(\frac{\delta}{2n\sqrt{n+2}}\right)^{\varepsilon^2 \times \frac{n+2}{n+1}} \text{ and } \delta_1(n) = \delta \end{cases}$$
3. **Maximum non-detectability gap:**
   Let τ_{-1} denotes the change-point coming before τ and τ_{+1} the one coming after τ. If the change-point τ_c is undetectable then the magnitude of the gap Δ must satisfy the following condition:
   $$\Delta \leqslant \Delta^\dagger_{\tau_{-1},\tau,\tau_{+1},\delta} \quad \text{where } \Delta^\dagger_{r,\tau,t,\delta} = 2\mathcal{C}_{r,\tau,t-1,\delta}$$

Proof. Lemma 4.8 is a direct application of Theorem 6 in [102] for the Bernoulli case where the sequence of observations $\mathbf{x}_{r:t}$ has $\frac{1}{2}$-sub-Gaussian noise. □

The Improved GLR strategy has been proven to be asymptotically order optimal, in the sense of Theorem 4.9 (see Theorem 6 in [102]). Therefore, comparing R-BOCPD against the Improved GLR strategy is a wise choice since GLR is considered as a very good baseline for the setting of the paper. Thus in Figure 4.6.3, we generate 2500 trajectories (sequences) of length $T = 2500$ where we vary the number of observation before the change-point from 10 to 500 and we vary the change-point gap Δ from 0.01 to 1. Then, we run R-BOCPD and Improved GLR strategy on the same sequence 360 times. Finally, we plot the mean detection delay difference between R-BOCPD and Improved GLR.

Figure 4.6.3 highlights the benefit of the R-BOCPD algorithm over the Improved GLR strategy. Indeed, the detection delay of R-BOCPD is slightly smaller than the one of the Improved GLR strategy. The while square means that Improved GLR isn't able to perform a detection while R-BOCPD does. Thus, for the small gap case, R-BOCPD is more robust than the Improved GLR strategy.

4.6.2 Experiments on Well-log data (Real world data)

These data have been studied in the context of change-point detection by [45] and has become a benchmark data set for uni-variate change-point detection. They consist on 4050 measurements $(y_1, ...y_{4050} \in [6.42 \times 10^4, 1.04 \times 10^5])$ of nuclear magnetic response taken

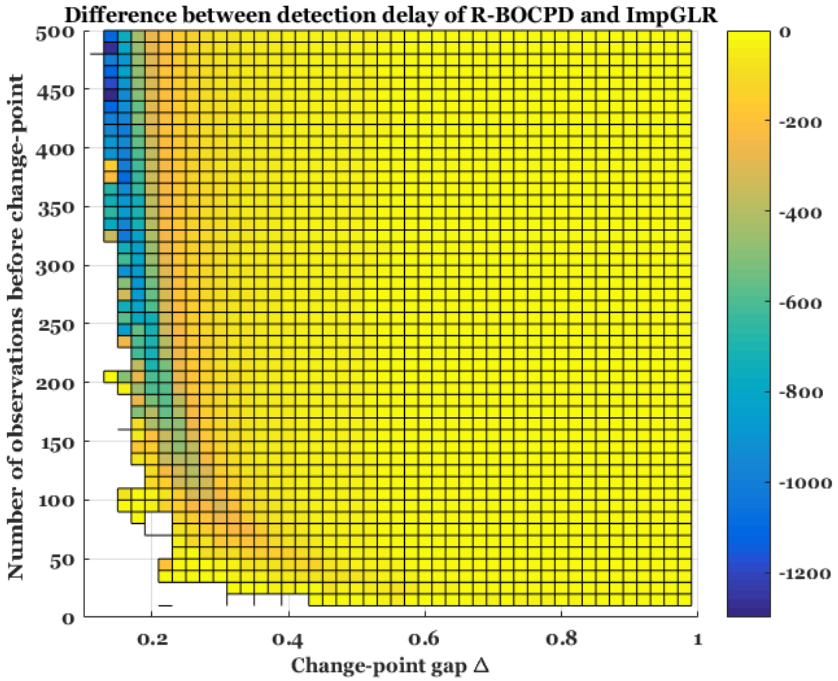


Figure 4.6.3: Difference between detection delays of R-BOCPD and GLR. The white square means that ImprGLR isn't able to perform a detection while R-BOCPD does. In all the experiments, we choose $\eta_{r,s,t} = \frac{1}{n_{r:t}}$ for R-BOCPD. The parameter δ (false alarm rate of ImpGLR) is set to 0.01.

during the drilling of a well. The data are used to interpret the geophysical structure of the rock surrounding the well. The variations in mean reflect the stratification of the earth's crust. In order to perform the experiments on the Well-log data, we typically proceed by computing the re-scaled values $\tilde{y}_1, ..., \tilde{y}_{4050} \in [0, 1]$ then we use the sequence of samples $x_1 \sim \mathcal{B}(\tilde{y}_1) ..., x_{4050} \sim \mathcal{B}(\tilde{y}_{4050})$ as input for BOCPD, R-BOCPD and ImpGLR (see figure 4.6.4). Finally, we plot the estimation of the change-points for each algorithm. Notice that R-BOCPD is more suitable to detect the change-point at $t = 2800$ than BOCPD or ImpGLR.

4.7 Extension to finite support distributions

In this chapter, the restarted Bayesian Online Change-point detector (R-BOCPD) has been specially designed for Bernoulli distributions (see Lemma 4.3 and Definition 4.3). This does not seem to have much limited its applications. Extensions to discrete-support distributions might be possible, resorting for instance to other concentration inequalities results, but this is not the purpose of this work. Note that, algorithms working on Bernoulli distributions can be conveniently extended to work with other set of distributions with bounded support: A classical way to do so when considering distribution $\mathcal{D}$ with bounded support in $[a, b]$, and some observation $y_t \sim \mathcal{D} \in [a, b]$, is to compute the re-scaled observation $\tilde{y}_t \in [0, 1]$, then use the sample $x_t \sim \mathcal{B}(\tilde{y}_t)$ as input (This transformation may not preserve optimality properties, though). The experiments in section 4.6.2 has been performed following this procedure.

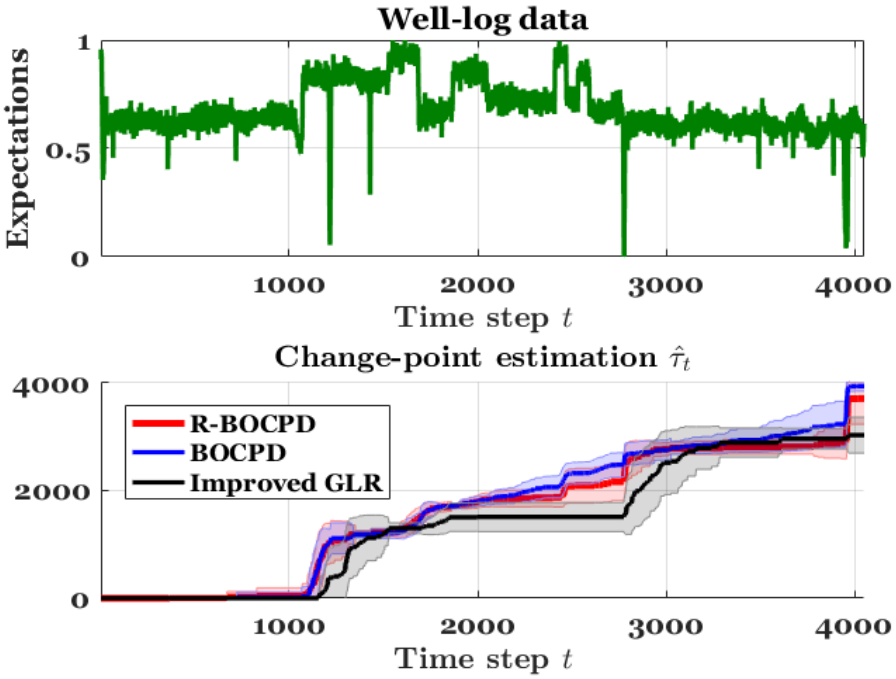


Figure 4.6.4: In all the experiment, we choose $\eta_{r,s,t} = \frac{1}{n_{r:t}}$ for R-BOCPD and $h = 1/T$ for BOCPD. The curves are averaged over 300 runs. (Their error bars are also plotted).

It should be noted that our analysis makes use of some specific properties of Bernoulli distributions, such as concentration inequalities, the key Lemma 4.3 and existence of explicit conjugate priors. Lemmas 4.5, 4.6 and 4.7 on the other hand are valid for any sub-Gaussian distributions. The Extension of our analysis to other popular distributions (e.g. Gaussians, Poisson, etc.) would need the specific equivalent of Lemma 4.3, and depend on the specific concentration inequalities and conjugate priors for this family. We believe this requires careful case by case examination.

4.8 Conclusion and future work

In this chapter, we introduced an improvement of the Bayesian Online Change-point Detector, called Restarted BOCPD. We provided a non-asymptotic analysis of its false alarm rate and detection delay, and shown numerically that our proposal outperforms its previous version. This constitutes arguably the first problem-dependent analysis of a Bayesian strategy in the context of change-point detection, and opens the path towards a complete analysis of BOCPD on the one hand, and the development of other Bayesian alternative on the other hand. We note that obtaining such guarantees is of primary importance in some applications, and in particular in the increasingly popular context of non-stationary multi-armed bandits.

Chapter 5

Decentralized Exploration in Multi-Armed Bandits

Overview of the chapter

In this chapter, we consider the *decentralized exploration problem*: a set of players collaborate to identify the best arm by asynchronously interacting with the same stochastic environment. The objective is to ensure privacy in the best arm identification problem between asynchronous, collaborative, and thrifty players. In the context of a digital service, we advocate that this decentralized approach allows a good balance between conflicting interests: the providers optimize their services, while protecting privacy of users and saving resources. We define the privacy level with respect to the amount of information an adversary could infer by intercepting all the messages concerning a single user. We provide a generic algorithm Decentralized Elimination, which uses any best arm identification algorithm as a subroutine. We prove that this algorithm ensures privacy, with a low communication cost, and that in comparison to the lower bound of the best arm identification problem, its sample complexity suffers from a penalty depending on the inverse of the probability of the most frequent players. Then, thanks to the generality of the approach, we extend the proposed algorithm to the non-stationary bandits. Finally, experiments illustrate and complete the analysis.

Publication. This chapter is mainly based on our article [47].

5.1 Introduction

We consider a collaborative exploration problem, the *decentralized exploration problem*. The main motivation of this new problem setting comes from sequential A/B and multivariate testing applications. For instance, most digital applications perform sequential A/B and multivariate testing in order to optimize the value of their audience. When a device is connecting to the application, the application presents an option to the user of the device. The aim is to maximize the clicks of users on the proposed options. Using the standard centralized exploration approach, the click stream of users is gathered and processed to choose the option which generates the most clicks. In this chapter, we formulate this standard exploration problem in a decentralized way.

When the event *"player n is active"* occurs, player n reads the messages received from other players and then chooses an arm to play. The reward of the played arm is revealed to player

n. Finally, she may send a message to the other players for sharing information about the arms.

The decentralized approach presents significant advantages. First, the clicks of users contain information that may be embarrassing when revealed, or that can be used by a third party in an undesirable way. The decentralization of exploration favors privacy since the click stream is not transmitted. However, it is not sufficient. The messages sent by a user may still contain private information such as her favorite topics, and therefore her political views, sexual orientation... As the players broadcast messages to other players, a malicious adversary can pretend to be a player, and then listening the exchanged messages. To ensure privacy one must guarantee that no useful information can be inferred from the messages sent by a single user. Second, the decentralization of exploration reduces the communication cost. This is a significant requirement for the Internet of Thing applications, since the smart devices often run on batteries. Third and finally, for all digital applications and in particular for the mobile phone applications, the decentralization with a low communication cost increases the responsiveness of applications by minimizing the number of interactions between the application server and the devices.

Finally, the objective of the *decentralized exploration problem* is threefold:

1. *sample efficiency*: finding a near-optimal arm with high-probability using a minimal number of interactions with the environment.
2. *user privacy*: protecting information contained in the interaction history of a single player.
3. *low communication cost*: minimizing the number of exchanged messages.

5.2 Related works

The problem of the best arm identification has been studied in two distinct settings in the literature:

- the fixed budget setting: the duration of the exploration phase is fixed and is known by the forecaster, and the objective is to maximize the probability of returning the best arm [29, 11, 51];
- the fixed confidence setting: the objective is to minimize the number of rounds needed to achieve a fixed confidence to return the best arm [44, 76, 51, 80].

In this chapter, we focus on the fixed confidence setting. Its theoretical analysis is based on the *Probably Approximately Correct* framework [140], and focuses on the sample complexity to identify a near-optimal arm with high probability. This theoretical framework has been used to analyze the best arm identification problem in [44], the dueling bandit problem in [139], the batched bandit problem in [116], the linear bandit problem in [127], the contextual bandit problem in [48], and the non-stationary bandit problem in [7].

The *decentralized multi-player multi-armed bandits* have been studied for opportunistic spectrum access in [98, 17, 108] or for optimizing communications in Internet of Things, even when no sensing information is available [23]. The objective is to avoid collisions between concurrent players that share the same channels, while choosing the best channels and minimizing the communication cost between players.

Recent years have seen an increasing interest for the study of the distributed collaborative scheme, where there is no collision when players choose the same arm at the same time. The distributed collaborative multi-armed bandits have been studied when the agents communicate through a neighborhood graph in [131, 94]. Here, we allow each player to broadcast messages to all players. In [35], a team of agents collaborate to handle the same multi-armed bandit problem. At each step the agent can broadcast her last obtained reward for the chosen arm to the team or pull an arm. The communication cost corresponds to the lost of the potential reward. As the pull of the arm of the agent is broadcasted, this approach does not ensure privacy of users. The tradeoff between the communication cost and the regret has been studied in the case of distributed collaborative non-stochastic experts [78]. In [68], the best arm identification task with fixed budget is distributed using Thompson Sampling in order to accelerate the exploration of the chemical space. In [69], the best arm identification task with fixed confidence is distributed on a parallel processing architecture. The analysis focuses on the trade-off between the number of communication rounds and the number of pulls per player. Here, we consider here that the players activation is under the control of the environment. As a consequence, synchronized communication rounds can no longer be used to control the communication cost. In our chapter, the cost of communications is assessed by the number of exchanged messages.

Moreover, our purpose is also to protect privacy of players. In the current context of massive storage of personal data and massive usage of models inferred from personal data, privacy is an issue. Even if individual data are anonymized, the pattern of data associated with an individual is itself uniquely identifying. The k-anonymity approach [130] provides a guarantee to resist to direct linkage between stored data and the individuals. However, this approach can be vulnerable to composition attacks: an adversary could use side information that combined with the k-anonymized data allows to retrieve a unique identifier [55]. The *differential privacy* [42] provides an alternative approach. The sensitive data are hidden. The guarantee is provided by algorithms that allow to extract information from data. An algorithm is differentially private if the participation of any record in the database does not alter the probability of any outcome by very much. The *differential privacy* has been extended to *local differential privacy* in which the data remains private even from the learner [41]. In [53], the authors propose an approach which handles the stochastic multi-armed bandit problem, while ensuring *local differential privacy*. The ε-differential privacy is ensured to the players by using a stochastic corruption of rewards. As all the rewards are transmitted to a centralized bandit algorithm, this approach has the maximum communication cost. Here, we define the privacy level with respect to the information about the preferred arms of a player, that an adversary could infer by intercepting the messages of this player. The messages could be corrupted feedbacks as in [53], or as we choose a more compact representation of the same information.

Chapter Outline In Section 5.3, we propose a new problem setting for ensuring privacy in the best arm identification problem between asynchronous, collaborative, and thrifty players. In Section 5.4, we propose a generic algorithm, Decentralized Elimination, which handles the *decentralized exploration problem* using any best arm identification algorithm as a subroutine. Theorem 5.12 states that Decentralized Elimination ensures privacy, finds an approximation of the best arm with high probability, and requires a low communication cost. Furthermore, Theorem 5.13 states a generic upper bound of the sample complexity of Decentralized Elimination. More specifically, Corollary 5.1 and 5.2 state the sample complexity bound when respectively Median Elimination and Successive Elimination [44] are used as subroutine.

Then, in Section 5.5, we extend the algorithmic approach to the *decentralized exploration in non-stationary bandit problem*. In Section 5.6, to illustrate and complete the analysis, we empirically compare the performances of Decentralized Elimination with two natural baselines [78]: an algorithm that does not share any information between the players, and hence that ensures privacy with a zero communication cost, and a centralized algorithm that shares all the information between players, and hence that does not ensure privacy and that has the maximum communication cost.

5.3 The decentralized exploration problem

Let $\mathcal{N} = \{1, ..., N\}$ be a set of N players. Let $x \in \mathcal{N}$ be a discrete random variable which realization denotes the index n of the *active* player (the player for which an event occurs). Let P_x be the probability distribution of x which is assumed to be stationary and unknown to the players. Let $\mathcal{K} = \{1, ..., K\}$ be a set of K arms. Let $y_k^n \in [0, 1]$ be the bounded random variable which realization denotes the reward of arm k for player n, and μ_k^n be its mean reward. Let $\mathbf{y}_{x=n} = \{y_k^n\}_{k \in \mathcal{K}}$ be the vector of independent random variables y_k^n. Let P_{y} and $P_{x,\mathrm{y}}$ be respectively the probability distribution of $\mathbf{y}$ and the joint probability distribution of x and $\mathbf{y}$, which are assumed to be unknown to the players.

Assumption 5.1: Stationary reward distributions
The mean reward of arms does not depend on time i.e.

$$\forall t, \forall n \in \mathcal{N} \text{ and } \forall k \in \mathcal{K}, \quad \mu_k^n(t) = \mu_k^n.$$

Assumption 5.2: Multi-armed bandits
The mean reward of arms does not depend on the player such that:

$$\forall n \in \mathcal{N} \text{ and } \forall k \in \mathcal{K}, \quad \mu_k^n = \mu_k.$$

Assumption 5.1 and 5.2 are used to focus on the stationary stochastic multi-armed bandits. This section lays the theoretical foundations of the *decentralized exploration problem* in its elementary form. The next section proposes an extension to the *decentralized exploration in non-stationary bandits*. The extension to the *decentralized exploration in contextual bandits* is discussed in section 5.7.

Definition 5.1: ε-optimal arm
Let $\varepsilon \in (0, 1]$. An arm $k \in \mathcal{K}$ is said to be ε-optimal, if $\mu_k \geqslant \mu_{k^*} - \varepsilon$, where $k^* = \arg\max_{k \in \mathcal{K}} \mu_k$. Thus, we denote by $\mathcal{K}_\varepsilon = \{k \in \mathcal{K} : \mu_k \geqslant \mu_{k^*} - \varepsilon\} \subseteq \mathcal{K}$ the set of all ε-optimal arms.

Definition 5.2: Message
A message $\lambda_k^n \in \{0,1\}$ is a binary random variable, that is sent by player n to other players, and where $\lambda_k^n = 1$ means that player n estimates that k is not an ε-optimal arm.[a]

[a] We choose a Bernoulli random variable for the sake of clarity. Notice that any random variable could be used as message.

Let $\mathcal{M}_n$ be the set of sent messages by player n at stopping time. Let $\mathcal{K}^n(l^n) \subseteq \mathcal{K}$ be the set of remaining arms at epoch $l^n \in \{1, ..., L\}$ for player n, where L is the maximal number of epochs.

Definition 5.3: (ε, η)-private
The decentralized algorithm $\mathcal{A}$ is (ε, η)-private for finding an ε-optimal arm, if for any player n, an adversary, that knows $\mathcal{M}_n$, the set of messages of player n, and the algorithm $\mathcal{A}$, cannot infer what arm is ε-optimal for player n with a probability higher than $1 - \eta$:

$$\forall n \in \mathcal{N}, \forall l^n \in \{1, ..., L\}, \nexists \eta_1, 0 \leqslant \eta_1 < \eta \leqslant 1, \mathbb{P}\Big(\mathcal{K}^n(l^n) \subseteq \mathcal{K}_\varepsilon | \mathcal{M}_n, \mathcal{A}\Big) \geqslant 1 - \eta_1.$$

$1 - \eta$ is the confidence level associated to the decision of the adversary. If η is small, then the adversary can use the set of messages $\mathcal{M}_n$ to infer with high probability which arm is an ε-optimal arm for player n. If η is high, the only information, that can be inferred by the adversary, is that the probability that an arm is an ε-optimal of arm for player n is a little bit higher than 0, which can be much lesser than the random choice $1/K$. η is a parameter which allows to tune the level of privacy: the higher η, the higher the privacy protection.

Remark. Since the adversary knows the algorithm $\mathcal{A}$, she knows also that $\mu_k^n = \mu_k$, where μ_k^n is the mean reward of arm k for player n. Hence the messages of others players provide useful information to infer the best arm of player n. Formally the guarantee is effective for player n, when the adversary has only logged the messages of player n. We have chosen to focus on this simplest case to analyze the *decentralized exploration problem*. In practice, it is not easy for the adversary to intercept and to decipher all messages of all players, and some assumptions are not fully met. The (ε, η)-privacy guaranty is efficient for real applications: based on the messages of player n, the uncertainty on the best arm of player n is high.

The goal of the *decentralized exploration problem* (see Algorithm 5.1) is to design an algorithm, that, when run on each player, samples effectively to find an ε-optimal arm for each player, while ensuring (ε, η)-privacy to players, and minimizing the number of exchanged messages.

The lower bound of the number of samples in $P_{x,y}$ needed to find with high probability an ε-optimal arm, which is $\Omega\left(\frac{K}{\varepsilon^2} \log \frac{1}{\delta}\right)$ [104], holds for the *decentralized exploration problem*, since a message can be sent at each time an arm is sampled by a player. The number of messages, that has to be exchanged in order to find with high probability an ε-optimal arm, could be zero if each player independently handles the best arm identification problem.

Algorithm 5.1 Decentralized Exploration Problem

Inputs: $\mathcal{K}$, $\varepsilon \in [0,1]$, $\eta \in [0,1]$
Output: an arm in each set $\mathcal{K}^n(l^n)$
Initialization: $l^n := 1$, $\mathcal{K}^n(l^n) := \mathcal{K}$

1: **repeat**
2: a player is sampled: $n \sim P_x$
3: player n gets the messages of other players
4: arm $k \in \mathcal{K}^n(l^n)$ is played by player n
5: player n receives reward $y_k^n \sim P_{x=n,y}$
6: **if** player n updates $\mathcal{K}^n(l^n)$ **then** $l^n := l^n + 1$
7: player n sends a message to other players
8: **until** $(\forall n \in \mathcal{N}, |\mathcal{K}^n(l^n)| = 1)$

Assumption 5.3: all players are active
We assume that all the players are active i.e.:

$$\forall n \in \mathcal{N} \quad P_x(x = n) \neq 0$$

Assumption 5.3 is a sanity check assumption for the *decentralized exploration problem*. Indeed, if it exists a player n such that $P_x(x = n) = 0$, then Algorithm 5.1 never stops (the stopping condition line 8 never happens).

5.4 Decentralized Elimination

5.4.1 ArmSelection subroutine

Before describing a generic algorithm for the *decentralized exploration problem*, we need to define an ArmSelection subroutine that handles all best arm identification algorithms. Let $\overline{\mathcal{K}^n}(l^n)$ and $\mathcal{K}^n(l^n)$ be respectively the set of eliminated arms and the set of remaining arms of player n at elimination epoch l^n, such that $\overline{\mathcal{K}^n}(l^n) \cup \mathcal{K}^n(l^n) = \mathcal{K}^n(l^n - 1)$.

Definition 5.4: ArmSelection subroutine
An ArmSelection subroutine takes as parameters an approximation factor ε, a confidence level $1-\eta$, and a set of remaining arm $\mathcal{K}^n(l^n)$. It samples a remaining arm in $\mathcal{K}^n(l^n)$ and returns the set of eliminated arms $\overline{\mathcal{K}^n}(l^n)$. An ArmSelection subroutine satisfies Properties 5.1 and 5.2.

Let t^n be the number of calls of the ArmSelection subroutine. Let $\mathcal{H}_{t^n}$ be the sequence of rewards of chosen arms $\{(k_1, y_{k_1}^n), (k_2, y_{k_2}^n), ..., (k_{t^n}, y_{k_{t^n}}^n)\}$. Let $f : \{1, ..., L\} \to [0,1]$ a function such that $\sum_{l^n=1}^{L} f(l^n) = 1$.

Property 5.1: Remaining ε-optimal arm
The guarantee that an ε-optimal arm is remaining is written as follows:

$$\forall l^n \in [0, L], \mathcal{K}^n(l^n) \subset \mathcal{K}^n(l^n - 1),$$
$$\mathbb{P}\Big(\{\mathcal{K}^n(l^n) \cap \mathcal{K}_\varepsilon = \emptyset\} | \mathcal{H}_{t^n}, \mathcal{K}^n(l^n - 1) \cap \mathcal{K}_\varepsilon \neq \emptyset\Big) \leqslant \eta \times f(l^n).$$

Property 5.2: Finite sample complexity
The guarantee that the sample complexity is finite takes the following form:

$$\exists t^n \geqslant 1, \forall \eta \in (0,1), \forall \varepsilon \in (0,1], \mathbb{P}\Big(\{\mathcal{K}^n(L) \subset \mathcal{K}_\varepsilon\} | \mathcal{H}_{t^n}\Big) \geqslant 1 - \eta.$$

Property 5.1 ensures that with high probability at least an ε-optimal arm remains in the set of arms $\mathcal{K}^n(l^n)$, while Property 5.2 ensures that the ArmSelection subroutine finds in a finite time an ε-optimal arm whatever the confidence level $1 - \eta$ and the approximation factor ε. To the best of our knowledge, all best arm identification algorithms can be used as ArmSelection subroutine with straightforward transformations. We consider three classes of best arm identification algorithms.

- The fixed-design algorithms use *uniform sampling* during a predetermined number of samples. Naive Elimination ($L = 1$ and $f(l^n) = 1$) and Median Elimination ($L = \log_2 K$ and $f(l^n) = 1/2^{l^n}$) [44] are *fixed-design* algorithms which can be used as ArmSelection subroutines.
- The successive elimination algorithms are based on *uniform sampling* and *arm eliminations*. At each time step a remaining arm is uniformly sampled. The empirical mean of the played arm is updated. The arms, which cannot be an ε-optimal arm with high probability, are discarded. If sub-optimal arms are discarded the epoch l is increased by one. Successive Elimination ($L = K$ and $f(l^n) = 1/K$) [44], KL-Racing ($L = K$ and $f(l^n) = 1/K$) [80] are *successive elimination* algorithms which can be used as ArmSelection subroutines.
- The explore-then-commit algorithms are based on *adaptive sampling* and *a stopping rule*. Rather than choosing arms uniformly, the *explore-then-commit* algorithms play one of the two critical arms: the empirical best arm, and the empirical sub-optimal arm associated with the maximum upper confidence bound. The stopping rule simply tests if the difference, between the maximum of upper confidence bound of sub-optimal arms and the lower confidence bound of the empirical best arm, is higher than the approximation factor ε. When the algorithm stops it returns the best arm. LUCB [76], KL-LUCB [80], UGapEc [51] can also be used as ArmSelection subroutines by returning the set of eliminated arms when the stopping event occurs ($L = 1$ and $f(l^n) = 1$).

5.4.2 Algorithm description

The basic idea of Decentralized Elimination is to use the vote of independent players, which communicate the arm they would like to eliminate with a high probability of failure for ensuring privacy. As the players are independent, the probability of failure of the vote is the multiplication of the individual probability of failures. The number of players needed for eliminating an arm is provided by the analysis.

Decentralized Elimination (see Algorithm 5.2) takes as parameters the privacy level η, the failure probability δ, the approximation factor ε, and an ArmSelection subroutine. It outputs an ε-optimal arm for each player with high probability. The algorithm sketch is described below: When player n is active (i.e. when player n is sampled):

- player n gets messages from other players (line 3).
- When enough players have eliminated an arm, it is eliminated from the shared set of arms $\mathcal{K}(l)$ and from the set of arms $\mathcal{K}^n(l^n)$ of player n with a low probability of failure (lines 5-10).
- When there is only one arm in $\mathcal{K}(l)$, it is an ε-optimal arm with high probability $1-\delta$, and the set of arms of player n is $\mathcal{K}(l)$ (line 11).
- An ArmSelection subroutine, run with a low confidence level $1-\eta$ (i.e. high privacy level) on the set $\mathcal{K}^n(l^n)$, samples an arm and returns $\overline{\mathcal{K}}^n(l^n)$ the set of arms that player n has eliminated at step t^n (line 13).
- When player n has eliminated an arm, she communicates to other players the index of the arm (lines 14-20).

Algorithm 5.2 Decentralized Elimination

Inputs: $\varepsilon \in (0,1]$, $\eta \in (0,1)$, $\delta \in [\eta^N, \eta^2]$, $\mathcal{K}$, an ArmSelection subroutine
Output: an arm in each set $\mathcal{K}^n(l^n)$
Initialization: $l := 1$, $\mathcal{K}(l) := \mathcal{K}$, $\forall n\ t^n := 1$, $l^n := 1$, $\mathcal{K}^n(l^n) := \mathcal{K}$, $\forall (k,n)\ \lambda_k^n := 0$

repeat
  player n is sampled: $n \sim P_x$
  player n gets the messages λ_k^j from other players
  if $|\mathcal{K}(l)| > 1$ **then**
    for all $k \in \mathcal{K}(l)$ **do**
      if $\sum_{j=1}^N \lambda_k^j \geqslant \lfloor \frac{\log \delta}{\log \eta} \rfloor$ **then**
        $\mathcal{K}(l) := \mathcal{K}(l) \setminus \{k\}$, $l := l+1$
        $\mathcal{K}^n(l^n) := \mathcal{K}^n(l^n) \setminus \{k\}$
      end if
    end for
    else $\mathcal{K}^n(l^n) := \mathcal{K}(l)$
  end if
  $\overline{\mathcal{K}}^n(l^n) :=$ ArmSelection$(\varepsilon, \eta, \mathcal{K}^n(l^n))$
  if $|\overline{\mathcal{K}}^n(l^n)| > 1$ **then**
    $l^n := l^n + 1$
    for all $k \in \overline{\mathcal{K}}^n(l^n)$ **do**
      $\mathcal{K}^n(l^n) := \mathcal{K}^n(l^n) \setminus \{k\}$
      $\lambda_k^n := 1$, λ_k^n is sent to other players
    end for
  end if
  $t^n := t^n + 1$
until $\forall n\ |\mathcal{K}^n(l^n)| = 1$

5.4.3 Analysis of the algorithm

Theorem 5.12 states the upper bound of the communication cost for obtaining with high probability an ε-optimal arm while ensuring (ε, η)-privacy to the players. The communication cost depends only on the problem parameters: the privacy constraint η, the probability of failure δ, the number of actions, and notably not on the number of samples. Notice that the probability of failure is low since the failure probability is lower than the level of privacy guarantee: $\delta < \eta$.

Theorem 5.12: Guarantees of Decentralized Elimination
Using any ArmSelection subroutine, Decentralized Elimination is an (ε, η)-private algorithm, that finds an ε-optimal arm with a failure probability $\delta \leqslant \eta^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}$ and that exchanges at most $\lfloor \frac{\log \delta}{\log \eta} \rfloor K - 1$ messages.

Proof. The proof is composed of three parts.

Part 1: (ε, η)-privacy. Let $E_{l^n} = \{\mathcal{K}^n(l^n) \cap \mathcal{K}_\varepsilon = \emptyset\}$ be the event denoting that there is no ε-optimal arm in the remaining set of arm $\mathcal{K}^n(l^n)$ at epoch l^n, and $\neg E_{l^n}$ be the event denoting that there is at least an ε-optimal arm in the remaining set of arm $\mathcal{K}^n(l^n)$ at epoch l^n.

As Decentralized Exploration ($\mathcal{A}$) performs an ArmSelection subroutine on each player, Property 5.1 ensures that for any player at epoch l^n:

$$\mathbb{P}\Big(E_{l^n}|\mathcal{H}_{t^n}, \mathcal{A}, \neg E_{l^n}\Big) \leqslant \eta \times f(l^n).$$

For the sake of simplicity, in the following we will omit the dependence on $\mathcal{A}$ of probabilities.

The message λ_k^n is sent by player n as soon as the arm k is eliminated from $\mathcal{K}^n(l^n)$ (see lines $17-18$ in Algorithm 5.2). Hence, we have:

$$\mathbb{P}\Big(E_{l^n}|\mathcal{M}_n, \neg E_{l^n}\Big) = \mathbb{P}\Big(E_{l^n}|\mathcal{H}_{t^n(l^n)}, \neg E_{l^n}\Big) \leqslant \eta \times f(l^n),$$

where $t^n(l^n)$ is the time where epoch l^n has begun.

To infer what arm is an ε-optimal arm for player n on the basis of $\mathcal{M}_n$ and $\mathcal{A}$, we first consider the favorable case for the adversary, where player n has sent $K-1$ elimination messages which corresponds to epoch $l^n = L$. Using Property 1 of the subroutine used by $\mathcal{A}$ and the set of messages $\mathcal{M}_n$ the adversary can infer that:

$$\mathbb{P}\Big(\{\mathcal{K}^n(L) \not\subseteq \mathcal{K}_\varepsilon\}|\neg E_{L-1}\Big) = \sum_{l^n=1}^{L} \mathbb{P}\Big(E_{l^n}|\mathcal{M}_n, \neg E_{l^n}\Big) \leqslant \eta \sum_{l^n=1}^{L} f(l^n) = \eta.$$

The previous equality holds since if at epoch l^n the event $\{\mathcal{K}^n(l^n) \not\subseteq \mathcal{K}_\varepsilon\}$ holds, then it holds also for all following epochs. Then the inequality is obtained by applying Property 1 to each element of the sum. Hence, if $l^n = L$ knowing the set of messages $\mathcal{M}_n$ and Property 1, the adversary cannot infer what arm is an ε-optimal arm for player n with a probability higher that $1-\eta$.

Otherwise if $l^n < L$ then $\mathcal{K}^n(L) \subset \mathcal{K}^n(l^n)$, which implies that:

$$\mathbb{P}\Big(\{\mathcal{K}^n(l^n) \not\subseteq \mathcal{K}_\varepsilon\}|\mathcal{M}_n, \neg E_{l^n}\Big) \geqslant \mathbb{P}\Big(\{\mathcal{K}^n(L) \not\subseteq \mathcal{K}_\varepsilon\}|\mathcal{M}_n, \neg E_{L-1}\Big).$$

Hence, if $l^n < L$ the adversary cannot infer what arm is an ε-optimal arm with a probability higher that $1-\eta$.

Part 2: Low probability of failure. An arm is eliminated when the events $\{k \notin \mathcal{K}^n(l^n)\}$ occur for $\lfloor \frac{\log \delta}{\log \eta} \rfloor$ independent players. Assumption 3 ($\forall n \in \mathcal{N}$, $P_x(x=n) \neq 0$) and Property 2 ensures that it exists a time $t = \sum_{n=1}^{N} t^n$ such that for $K-1$ arms, there are $\lfloor \frac{\log \delta}{\log \eta} \rfloor$ voting players. Moreover, Property 1 implies that $\forall n \in \mathcal{N}$, $\forall l^n$:

$$\mathbb{P}\Big(\{\mathcal{K}^n(l^n) \not\subseteq \mathcal{K}_\varepsilon\}|\mathcal{M}_n, \neg E_{l^n}\Big) \leqslant \eta \times f(l^n).$$

Hence, the $\lfloor \frac{\log \delta}{\log \eta} \rfloor$ independent voting players eliminate the ε-optimal arm with a probability at most:

$$\mathbb{P}\Big(\{\mathcal{K}(l) \not\subseteq \mathcal{K}_\varepsilon\}|\mathcal{M}, \neg E_l\Big) \leqslant (\eta \times f(l))^{\lfloor \frac{\log \delta}{\log \eta} \rfloor},$$

where $\mathcal{K}(l)$ denotes the shared set of remaining arms at elimination epoch l (see line 7 in Algorithm 5.2), and $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2 \cup \ldots \cup \mathcal{M}_N$.

If the algorithm fails, then the following event occurs : at stopping time, $\exists k \in \mathcal{K}(L), k \notin \mathcal{K}_\varepsilon$. Using the union bound, we have:

$$\mathbb{P}\Big(\{\mathcal{K}(L) \not\subseteq \mathcal{K}_\varepsilon\} | \mathcal{M}, \neg E_{L-1}\Big) \leqslant \sum_{l=1}^{L} (\eta \times f(l))^{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \leqslant \eta^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}.$$

Finally notice that:

$$\eta^{\lceil \frac{\log \delta}{\log \eta} \rceil} \leqslant \eta^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}.$$

Part 3: Low communication cost. The index of each arm is sent to other players no more than once per player (see line 17 in Algorithm 5.2). When $\lfloor \frac{\log \delta}{\log \eta} \rfloor$ messages have been sent for an arm, this arm is eliminated for all players (see lines $4-9$ in Algorithm 5.2).

Thus $\lfloor \frac{\log \delta}{\log \eta} \rfloor (K-1)$ messages are sent to eliminate the sub-optimal arms. Then, at most $\lfloor \frac{\log \delta}{\log \eta} \rfloor - 1$ messages have been sent for the remaining arm. Thus, the number of sent messages is at most $\lfloor \frac{\log \delta}{\log \eta} \rfloor K - 1$.

□

To finely analyze the sample complexity of Decentralized Elimination algorithm, one needs to handle the randomness of the voting process. Let $T_{P_{x,y}}$ be the number of samples in $P_{x,y}$ at stopping time. Let T_{P_y} be the number of samples in P_y needed by the ArmSelection subroutine to find an ε-optimal arm with high probability. Let $\mathcal{N}_M$ be the set of the $M = \lfloor \frac{\log \delta}{\log \eta} \rfloor$ most likely players, let $p^\star = \min_{n \in \mathcal{N}_M} P_x(x=n)$, and let $p^\dagger = \min_{n \in \mathcal{N}} P_x(x=n)$.

Theorem 5.13: Sample complexity of Decentralized Elimination
Using any ArmSelection subroutine, with a probability higher than $(1-\delta)\left(1 - I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right)\right)^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}$ Decentralized Elimination stops after:

$$\mathcal{O}\left(\frac{1}{p^\star}\left(T_{P_y} + \sqrt{\frac{1}{2}\log\frac{1}{\delta}}\right)\right) \text{ samples in } P_{x,y},$$

where $I_a(b,c)$ denotes the incomplete beta function evaluated at point a with parameters b, c.

Proof. Let T_n be the number of samples of player n at time $T_{P_{x,y}}$ when the algorithm stops. T_n is a binomial law of parameters $T_{P_{x,y}}, P_x(x=n)$. We have:

$$\mathbb{E}_{P_x}[T_n] = P_x(x=n) T_{P_{x,y}}.$$

Let $\mathbf{B}_{\delta,\eta}$ be the set of players that have the $\lfloor \frac{\log \delta}{\log \eta} \rfloor$ highest T_n. The algorithm does not stop, if the following event occurs: $E_1 = \{\exists n \in \mathbf{B}_{\delta,\eta}, T_n < T_{P_y}\}$.

Applying Hoeffding inequality, we have:

$$\mathbb{P}\Big(T_n - P_x(x=n) T_{P_{x,y}} \leqslant -\varepsilon\Big) \leqslant \exp(-2\varepsilon^2)$$

When $\neg E_1$ occurs, $\forall n \in \mathbf{B}_{\delta,\eta}$ we have with a probability at most δ:

$$T_{P_y} - P_x(x=n)T_{P_{x,y}} \leqslant -\sqrt{\frac{1}{2}\log\frac{1}{\delta}}.$$

Then, when $\neg E_1$ occurs we have with a probability at most δ:

$$T_{P_{x,y}} \geqslant \frac{1}{p_{\delta,\eta}}\left(T_{P_y} + \sqrt{\frac{1}{2}\log\frac{1}{\delta}}\right),$$

where $p_{\delta,\eta} = \min_{n\in \mathrm{B}_{\delta,\eta}} P_x(x=n)$.

Finally if E_1 does not occur, then we have with a probability at least $1-\delta$:

$$T_{P_{x,y}} \leqslant \frac{1}{p_{\delta,\eta}}\left(T_{P_y} + \sqrt{\frac{1}{2}\log\frac{1}{\delta}}\right).$$

Let $\mathcal{N}_M$ bet the set of the $M = \lfloor \frac{\log\delta}{\log\eta} \rfloor$ most likely players. Let $n^* = \arg\min_{n\in\mathcal{N}_M} P_x(x=n)$, and $p^\star = \min_{n\in\mathcal{N}_M} P_x(x=n)$.

Now, we consider the following event: $E_2 = \left\{n^* \notin \mathbf{B}_{\delta,\eta}\right\}$. By the definition of $\mathbf{B}_{\delta,\eta}$, the event E_2 is equivalent to the event $\left\{T_{n^*} < T_{P_y}\right\}$. Then, we have:

$$\mathbb{P}\Big(T_{n^*} < T_{P_y}\Big) = I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right),$$

where $I_a(b,c)$ denotes the incomplete beta function evaluated at the point a with parameters b and c. Finally, with a probability at least $(1 - I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right))^{\lfloor \frac{\log\delta}{\log\eta}\rfloor}$, we have $p_{\delta,\eta} = p^\star$.

□

As the number of players involved in the vote is set as small as possible $\lfloor \frac{\log\delta}{\log\eta} \rfloor$, Theorem 2 provides with high probability [1] the sample complexity of Decentralized Elimination. Notice, that when the number of players is high, and when the distribution of players is far from the uniform distribution, we have $p^\star \gg p^\dagger$.

Corollary 5.1: Sample complexity of Decentralized Median Elimination

With a probability higher than $(1-\delta)\left(1 - I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right)\right)^{\lfloor \frac{\log\delta}{\log\eta}\rfloor}$ Decentralized Median Elimination stops after:

$$\mathcal{O}\left(\frac{1}{p^\star}\left(\frac{K}{\lfloor \frac{\log\delta}{\log\eta}\rfloor \varepsilon^2}\log\frac{1}{\delta} + \sqrt{\frac{1}{2}\log\frac{1}{\delta}}\right)\right) \textit{ samples in } P_{x,y}.$$

[1] for instance, $I_{0.99}(500, 500) = 1.47 \times 10^{-302}$

Proof. We have:

$$\eta^{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \leqslant \delta = \eta^{\frac{\log \delta}{\log \eta}} \leqslant \eta^{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \Rightarrow \frac{1}{\delta} \geqslant \frac{1}{\eta^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}} \Leftrightarrow \log \frac{1}{\eta} \leqslant \frac{1}{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \log \frac{1}{\delta}$$

Median Elimination algorithm [44] finds an ε-optimal arm with a probability at least $1-\eta$, and needs at most:

$$T_{P_y} = \mathcal{O}\left(\frac{K}{\varepsilon^2} \log \frac{1}{\eta}\right) \leqslant \mathcal{O}\left(\frac{K}{\lfloor \frac{\log \delta}{\log \eta} \rfloor \varepsilon^2} \log \frac{1}{\delta}\right) \text{ samples in } P_y.$$

Then, the use of Theorem 5.13 finishes the proof.

□

Corollary 5.2: Sample complexity of Decentralized Successive Elimination

With a probability higher than $(1-\delta)\left(1 - I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right)\right)^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}$ Decentralized Successive Elimination stop after:

$$\mathcal{O}\left(\frac{1}{p^\star}\left(\frac{K}{\varepsilon^2}\left(\log K + \frac{1}{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \log \frac{1}{\delta}\right) + \sqrt{\frac{1}{2} \log \frac{1}{\delta}}\right)\right) \textit{ samples in } P_{x,y}.$$

Proof. Successive Elimination algorithm [44] finds an ε-optimal arm with a probability at least $1-\eta$, and needs at most:

$$T_{P_y} = \mathcal{O}\left(\frac{K}{\varepsilon^2} \log \frac{K}{\eta}\right) \leqslant \mathcal{O}\left(\frac{K}{\varepsilon^2}\left(\log K + \frac{1}{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \log \frac{1}{\delta}\right)\right)$$

samples in $P_{x,y}$. Then the use of Theorem 2 finishes the proof. □

Corollary 5.1 and 5.2 state the number of samples in $P_{x,y}$ needed to find an ε-optimal arm by Decentralized Elimination using respectively Median Elimination and Successive Elimination as ArmSelection subroutines.

To illustrate these results, we consider the case of the uniform distribution of players. With a failure probability at most $\delta = \eta^N$ the number of sample in $P_{x,y}$ needed by Decentralized Median Elimination to find an ε-optimal arm is:

$$\mathcal{O}\left(\frac{K}{\varepsilon^2} \log \frac{1}{\delta} + N\sqrt{\frac{1}{2} \log \frac{1}{\delta}}\right) \text{ samples in } P_{x,y}.$$

In comparison to an optimal best arm identification algorithm, which communicates all the messages and does not provide privacy protection guarantee, which has a sample complexity in $\mathcal{O}\left(\frac{K}{\varepsilon^2} \log \frac{1}{\delta}\right)$, the sample complexity of Decentralized Elimination mostly suffers from a penalty depending on the inverse of the probability of the most frequent players, that in the

case of uniform distribution of players is linear with respect to the number of players. The proofs of Theorem 5.13, Corollary 5.1 and 5.2 are provided in Appendix.

5.5 Decentralized exploration in non-stationary bandits

Recently, the best arm identification problem has been studied in the case of non-stationary bandits, where Assumption 5.1 does not hold [7, 1]. In the first reference, the authors analyze the non-stationary stochastic best-arm identification in the fixed confidence setting by splitting the game into independent sub-games where the best arm does not change. In the second reference, the authors propose a simple and anytime algorithm, which is analyzed for stochastic and adversarial rewards in the case of fixed budget setting. For the consistency of the chapter, which focuses on fixed confidence setting, we choose to extend Decentralized Elimination to Successive Elimination with Randomized Round-Robin (SER3 [7]). Basically, SER3 consists in shuffling the set of arms at each step of Successive Elimination. SER3 works for the sequences where Assumption 5.4 holds.

Assumption 5.4: Positive mean-gap
For any $k \in \mathcal{K} \setminus \{k^*\}$ and any $[\tau] \in \mathbb{T}(\tau)$ with $\tau \geqslant \log \frac{K}{\eta}$, we have:

$$\Delta_k^* ([\tau]) = \frac{1}{\tau} \sum_{i=1}^{\tau} \sum_{j=i}^{i+K_i-1} \frac{\Delta_{k^*,k}(j)}{K_i} > 0,$$

where $\mathbb{T}(\tau)$ is the set containing all possible realizations of τ round-robin steps, $\Delta_{k^*,k}(t)$ is the difference between the mean reward of the best arm and the mean reward of arm k at time t, and K_t is the number of remaining arms at time t.

We provide below the sample complexity bound of Decentralized Successive Elimination with Randomized Round-Robin (DSER3), which is simply Decentralized Elimination using SER3 as the ArmSelection subroutine.

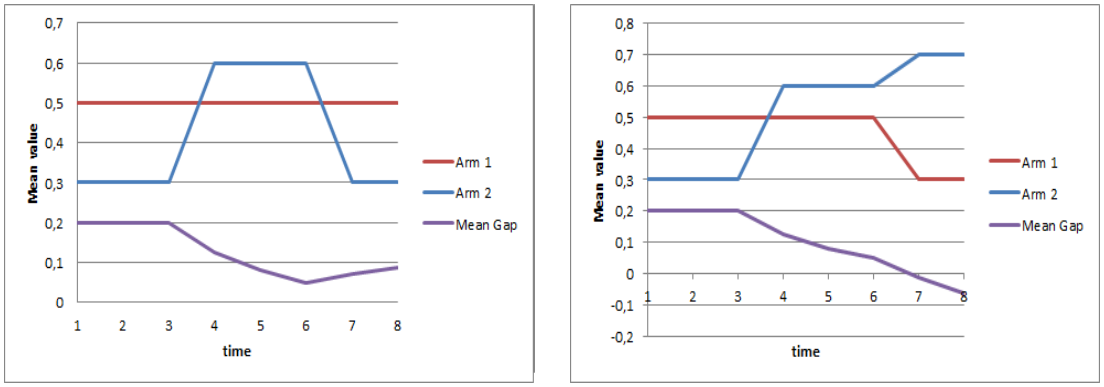
Theorem 5.14: Guarantees of DSER3
For $K \geqslant 2$, $\delta \in (0, 0.5]$, for the sequences of rewards where Assumption 5.4 holds, DSER3 is an (ε, η)-private algorithm, that exchanges at most $\lfloor \frac{\log \delta}{\log \eta} \rfloor K - 1$ messages, that finds an ε-optimal arm with a probability at least $(1-\delta)\left(1 - I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right)\right)^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}$, and that stops after:

$$\mathcal{O}\left(\frac{1}{p^\star}\left(\frac{K}{\varepsilon^2}\left(\log K + \frac{1}{\lfloor \frac{\log \delta}{\log \eta} \rfloor} \log \frac{1}{\delta}\right) + \sqrt{\frac{1}{2} \log \frac{1}{\delta}}\right)\right) \text{ samples in } P_{x,y}.$$

Proof. Theorem 5.14 is a straightforward application of Theorem 5.13, where T_{P_y} is stated in Theorem 5.12 [7]. □

Finally Decentralized Successive Elimination with Randomized Round-Robin and Reset (DSER4) handles any sequence of rewards: when Assumption 5.4 does not hold a switch

occurs (see Figures 5.5.1a, 5.5.1b). DSER4 consists in using SER4 [7] as the Arm Elimination subroutine in Decentralized Elimination. In addition, when a reset occurs in SER4, Decentralized Elimination is reset.



(a) Mean gap versus time. Assumption 5.4 holds: the mean gap stays positive.

(b) Mean gap versus time. Assumption 5.4 does not hold: a switch occurs a time 7.

Figure 5.5.1: Examples of mean gap versus time.

Assumption 5.4 trivially holds when the mean rewards do not change. When the mean rewards change, Assumption 5.4 parts the small changes that do not imply a change of mean gap (see Figure 5.5.1a) from major changes where the mean gap changes (see Figure 5.5.1b). For more details see [7].

Theorem 5.15: Guarantees of DSER4

For $K \geqslant 2$, $\varepsilon \geqslant \frac{\eta}{K}$, $\varphi \in (0,1]$, for any sequences of rewards, DSER4 is an (ε, η)-private algorithm, that exchanges on average at most $\varphi T(\lfloor \frac{\log \delta}{\log \eta} \rfloor K - 1)$ messages, and that plays, with an expected probability at most $\delta + \varphi T I_{1-p^\star}\left(T_{P_{x,y}} - T_{P_y}, 1 + T_{P_y}\right)^{\lfloor \frac{\log \delta}{\log \eta} \rfloor}$, a sub-optimal arm on average no more than:

$$\mathcal{O}\left(\frac{1}{p^\star}\left(\frac{1}{\varepsilon^2}\sqrt{\frac{SK\log K + \frac{1}{\lfloor \frac{\log \delta}{\log \eta} \rfloor}\log\frac{1}{\delta}}{\delta^{\frac{1}{\lfloor \frac{\log \delta}{\log \eta} \rfloor}}}} + \sqrt{\frac{1}{2}\log\frac{1}{\delta}}\right)\right) \text{ times}$$

where S is the number of switches of best arms, φ is the probability of reset in SER4, T is the time horizon, and the expected values are taken with respect to the randomization of resets.

Proof. The upper bound of the expected number of times a suboptimal arm is played by SER4, is stated in Corollary 5.2 [7]. Then this upper bound is used in Theorem 5.13 to state the upper bound of the expected number of times a suboptimal arm is played using DSER4. The expected number of resets is φT. Theorem 2 provides the success probability of each run of Decentralized Elimination, which states the expected failure probability of DSER4. Then using Theorem 1 the expected upper bound of the number of exchanged messages is stated. □

5.6 Experiments

5.6.1 Experimental setting

To illustrate and complete the analysis of Decentralized Elimination, we run three synthetic experiments:

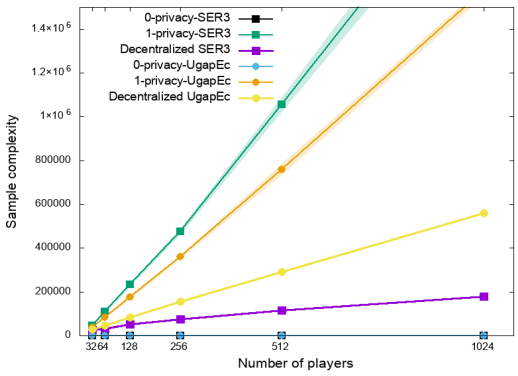
- **Problem 1: Uniform distribution of players.** There are 10 arms. The optimal arm has a mean reward $\mu_1 = 0.7$, the second one $\mu_2 = 0.5$, the third one $\mu_3 = 0.3$, and the others have a mean reward of 0.1. Each player has a probability equal to $1/N$.
- **Problem 2: 50% of players generates 80% of events.** The same 10 arms are reused with an unbalanced distribution of players. The players are split in two groups of sizes $N/2$. When a player is sampled, a uniform random variable $z \in [0, 1]$ is drawn. If $z < 0.8$ the player is uniformly sampled from the first group, otherwise it is uniformly sampled from the second group.
- **Problem 3: non-stationary rewards.** The distribution of players is uniform. The same 10 arms are reused. The mean reward of the optimal arm does not change during time. The mean reward of sub-optimal arms linearly decrease such that: $\mu(t) = \mu(0) - 10^{-5}t$.

As comparison points, we include two natural baselines:

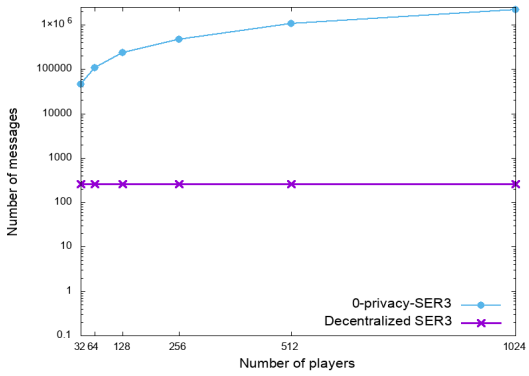
- 1-privacy: an $(\varepsilon, 1)$-private algorithm that does not share any information between the players, and hence that runs at a zero communication cost. The ArmSelection subroutine is run with parameters $(\varepsilon, \delta/N)$ to ensure that all the players find with a probability $1 - \delta$ an ε-optimal arm.
- 0-privacy: an $(\varepsilon, 0)$-private algorithm that shares all the information between players, and hence that runs at a minimal privacy and a maximal communication cost. This algorithm does not meet the original goal but is interesting as a reference to assess the sample efficiency loss stemming from the privacy constraint.

As ArmSelection subroutines, We choose two frequentist algorithms[2] based on Hoeffding inequality: a *explore-then-commit* algorithm UGapEc [51] and a *successive elimination* algorithm SER3 [7], which handles non-stationary rewards. Combining Decentralized Elimination and the two baselines with the two ArmSelection subroutines, we compare 6 algorithms (Decentralized SER3, Decentralized UGapEc, 1-privacy-SER3, 1-privacy-UGapEc, 0-privacy-SER3, 0-privacy-UGapEc) on the three problems. The algorithms are compared with respect to two key performance indicators: the sample complexity and the communication cost. For all the experiments, ε is set to 0.25, and δ is set to 0.05. The privacy level η is set to 0.9. All the curves and the measures are averaged over 20 trials.

[2] due to high values of sampling complexity obtained by Median Elimination which flatten the differences between algorithms, we report its performance in appendix.

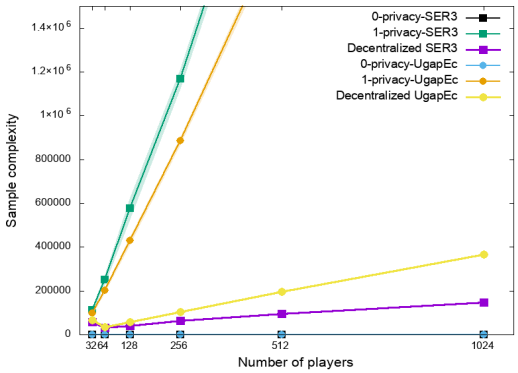


(a) Problem 1 - Sample Complexity.

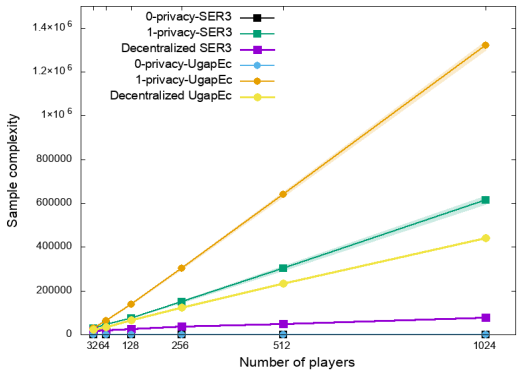


(b) Problem 1 - Number of messages.

Figure 5.6.1: Uniform distribution of players. The sample complexities of 0-privacy baselines are the same: 800.



(a) Problem 1 - Sample Complexity.



(b) Problem 1 - Number of messages.

Figure 5.6.2: 50% of players generates 80% of events (a), and the mean rewards of sub-optimal arms linearly decrease (b).

5.6.2 Discussion about the empirical results

The results reveal that the sample efficiency of 1-privacy baselines is horrendous on both problems: it increases super-linearly as the number of players increases. Worse, when the distribution of players moves away from the uniformity, which is the case in most of digital applications, the performances of 1-privacy baselines decreases (see Figure 5.6.1a, 5.6.2a). Unlike 1-privacy baselines, the performances of Decentralized UGapEc and Decentralized SER3 increases in Problem 2 (see Figure 5.6.2a). More precisely, the sample complexity curves of Decentralized UGapEc and Decentralized SER3 exhibit two regimes: first the sample complexity decreases (between 32 to 64 players), and then the sample complexity linearly increases with the number of players. The values of hyper-parameters: $\delta = 0.05$ and $\eta = 0.9$, imply that the number $M = \lfloor \frac{\log \delta}{\log \eta} \rfloor$ of player votes required to eliminate an arm is 28. In Problem 2 with 32 players, it means that the algorithm has to wait for infrequent players votes to terminate. When the number of players is 64, this issue disappears. This is the reason why the sample complexity for 64 players is lower than for 32 players. The linear dependency of the sample complexity with respect to the number of players of the second regime is due to the fact that in the considered problems, the probability of the most likely player $p^\star$decreases in $1/N$.

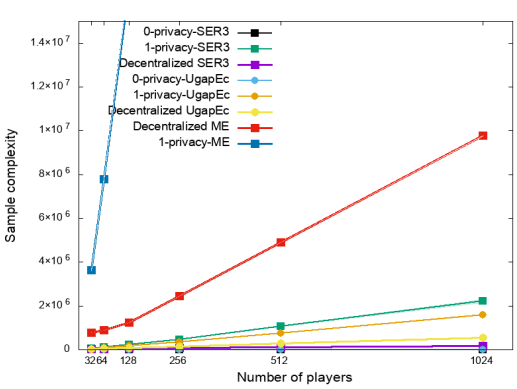
Concerning the ArmSelection subroutines, we observe that 1-privacy-UGapEc clearly outperforms 1-privacy-SER3 on stationary problems (see Figures 5.6.1a and 5.6.2a). Moreover, the performance gain of 1-privacy-UGapEc increases with the number of players. This is due to the adaptive sampling strategy of UGapEc: by sampling alternatively the empirical best arm and the most loosely estimated sub-optimal arm, 1-privacy-UGapEc reduces the variance of the sample complexity, and thus reduces the maximum of sample complexities of players. However, when used as a subroutine in Decentralized Elimination, the *successive elimination* algorithms such as SER3 are more efficient: thanks to the different sub-optimal arms which are progressively eliminated by different groups of voting players, Decentralized SER3 clearly outperforms Decentralized UGapEc (see Figure 5.6.1a and 5.6.2a).

When the mean rewards of sub-optimal arms are decreasing (Figure 5.6.2b), in comparison to SER3 the performances of UGapEc, which is not designed for non-stationary rewards, collapse: 1-privacy-UGapEc and Decentralized UGapEc are respectively outperformed by 1-privacy-SER3 and Decentralized SER3. The optimistic approach used in the sampling rule of UGapEc is too optimistic when the mean reward are decreasing.

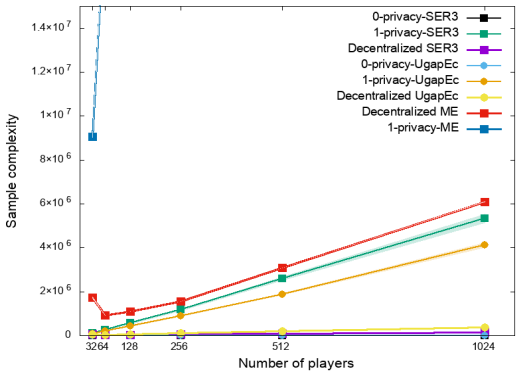
The communication cost is the number of exchanged messages: 1-privacy baselines send zero messages, while 0-privacy baselines send $N-1$ messages per time step until the ε-optimal arm is found. Decentralized SER3 needs three to four orders of magnitude less messages than 0-privacy-SER3 (see Figure 5.6.1b).

5.6.3 Additional Experiments

Median Elimination is designed to be order optimal in the worst case: its sample complexity is in $O(K \log \frac{1}{\delta})$. However, in practice it is clearly outperformed by Successive Elimination or UGapEc on both problems (see Figures 5.6.3a, 5.6.3b).



(a) Problem 1: Uniform distribution of players.



(b) Problem 2: 50% of players generates 80% of events.

Figure 5.6.3: Evaluating the Median elimination algorithm

5.7 Conclusion and perspectives

We have provided a new definition of privacy for the decentralized algorithms. We have proposed a new problem, the *decentralized exploration problem*, where players sampled from a distribution collaborate to identify a near-optimal arm with a fixed confidence, while ensuring privacy to players and minimizing the communication cost. We have designed and

analyzed a generic algorithm for this problem: Decentralized Elimination uses any best arm identification algorithm as an ArmSelection subroutine. Thanks to the generality of the approach, we have extended the analysis of the algorithm to the case where the distributions of rewards are not stationary. Finally, our experiments suggest that *successive elimination* algorithms are better suited for the *decentralized exploration problem* than *explore-then-commit* algorithms.

Future work may focus on user-dependent best arms. When Assumption 2 does not hold, Decentralized Elimination finds with high probability the best arm of the most frequent players. However, in lot of applications the players can observe a context before choosing an arm. The extension of the proposed approach to *contextual bandits* is not straightforward because to collaborate for building a model, the players have to exchange messages about their favorite arms and their contextual variables, that also contain private information.

Part III

Industrial applications

Chapter 6

LoRa Network optimization via multi-armed bandits

Overview of the chapter

The use of Low Power Wide Area Networks (LPWANs) is growing due to their advantages in terms of low cost, energy efficiency and range. Although LPWANs attract the interest of industry and network operators, it faces certain constraints related to energy consumption, network coverage and quality of service. In this chapter we demonstrate the possibility to optimize the performance of the LoRaWAN (Long Range Wide Area Network) technology, one of the most widely used LPWA technology. We suggest that nodes use light-weight learning methods, namely, multi-armed bandit algorithms, to select the communication parameters (spreading factor and emission power). Extensive simulations show that such learning methods allow to manage the trade-off between energy consumption and packet loss much better than an Adaptive Data Rate (ADR) algorithm adapting spreading factors and transmission powers on the basis of Signal to Interference and Noise Ratio (SINR) values.

Publication. This chapter is mainly based on our article [83].

6.1 Introduction and related works

The interest of the industry towards the Low Power Wide Area Networks (LPWANs) is gradually increasing [117]. Several technologies operating on license-free industrial, scientific and medical radio bands (868 MHz for Europe, 915 MHz for North America and 433 MHz band for Asia) are used by industries; among those, the LoRaWAN technology is one of the most widely used.

The LoRaWAN network architecture is based on a star-of-stars topology with gateways forming a transparent bridge. These gateways relay messages between end-devices and a central network server in the backend. Nodes use a single-hop wireless connection to one or more gateways whereas gateways are connected to the network server using standard IP connections. Communications with end-point nodes are generally bi-directional, but it is also possible to support multi-cast operations [8]. Communications between end-devices and gateways are spread out over different frequency channels, using so-called spreading factors (SF) defined as the logarithmic ratio between the symbol rate (R_s) and the chip rate (R_c): $\mathrm{SF} = \log_2(R_c/R_s)$. Accordingly, selecting the data rate (or equivalently, the SF) can be seen as a trade-off between communication range and message duration [3]. This possibility

to manage data rates and power outputs for each end-device allows to maximize network capacity [8].

The LoRa spread spectrum modulation scheme defines six different spreading factors, SF7 (data rate of 50 kbps) to SF12 (data rate of 0.3 kbps). SF7 allows to send messages with a higher data rate and a reduced time on air but at a shorter distance than the others SFs and vice versa. Before sending a packet, each node also selects a transmission power between 2 dBm and 14 dBm in addition to selecting a spreading factor.

As any IoT technology, LoRa faces several constraints. One strict constraint is the optimization of energy consumption as the end-devices have generally limited energy resources. Another constraint is a limited *duty cycle*, preventing nodes from sending data too often in order to leave space for the other nodes.

LoRa can operate successfully at ranges exceeding 15 km in suburban settings, and more than 2 km in dense urban environments [43]. However, it is necessary to choose an appropriate spreading factor, to have a compromise between data rate and network coverage in order to avoid high battery consumption or frequent packet loss. That is the possibility offered by the so-called Adaptive Data Rate (ADR) scheme [8], which currently implemented in LoRa nodes.

Research to optimize time-on-air, receiver sensitivity, packet loss and energy has been conducted by combining game theory and auction-based algorithms [65]. The goal there is to choose the best transmission power without changing the allocated spreading factor; the global results are satisfactory, but in case of a change in power requirements all estimations need to be performed again, implying a huge overhead. In [65], the authors design a multi-radio testing instrument, called LoRabox, for sending and receiving data packets via LoRaWAN, Bluetooth Low Energy (BLE) and Wi-Fi, but this instrument does not have any mechanism to evaluate the cost associated with this overhead. Recently, an approach based on multi-armed bandits has been proposed to optimize the channel choice of end-devices in IoT networks [26]. The authors consider an hypothetical protocol where in each time slot the devices try to send packets to a unique Base Station. In the considered environment, half of the devices are static and the other half learns, hence due to the learning the environment is non-stationary. The reported performance of the stochastic multi-armed bandit algorithms are close to the ones of the optimal policy, which suggests that the environment evolves slowly and sparsely and that learning methods are good candidates to improve performance in those settings.

In this chapter, we aim to minimize the energy consumption and the packet losses of end-devices in a LoRa network. These objectives are conflicting, hence a trade-off between them has to be found by selecting the transmission power and the spreading factor. We propose to use multi-armed bandit algorithms, instead of the standard ADR algorithm, to handle that trade-off. In contrast to the ADR algorithm–managed by the gateway–, the SF and power choices are left to each node, that learns and adapts to its environment with limited memory and computational costs. Our work departs from the recent study of [26], since instead of considering an hypothetical protocol, we test multi-armed bandit approaches on the actual LoRa protocols. The tested algorithms compete with the standard ADR algorithm: in our simulation setting, we consider that all devices select their transmission parameters using the ADR algorithm, except one that learns using a multi-armed bandit algorithm. This implies an a priori non-stationary environment (due to ADR having nodes change their parameters), which is likely to favor the non-stationary bandit algorithms over stationary ones.

6.2 The ADR (Adaptive Data Rate) algorithm

In order to limit the energy consumption while ensuring successful transmissions, the ADR algorithm, currently recommended by the LoRa Alliance, and implemented by the network, assigns the transmission power and the spreading factor to the nodes based on the packets received from them. Note that only the network may increase the data rate (i.e., decrease the SF) through ADR, while only nodes may decrease their data rate (i.e., increase their SF) during transmissions.

The algorithm is based on the Signal to Interference and Noise Ratio (SINR) of the last 20 transmissions. For each node, the last 20 Signal to Interference and Noise Ratio (SINR) values at the gateway are taken into account to calculate a so-called margin SINR, denoted by $\text{SINR}_{\text{margin}}$. Also a constant margin (10 dB) is taken into account. Mathematically, we have

$$\text{SINR}_{\text{margin}} := \text{SINR}_{\text{max}} - \text{SINR}_{\text{Required}} - \text{Margin},$$

where:

- SINR_{max} is the maximum SINR of the last 20 (successfully) received packets from the node,
- Margin is a constant set to 10 dB,
- and $\text{SINR}_{\text{Required}}$ depends on the spreading factor, as reported in Table 6.1.

Table 6.1: Spreading Factors, and corresponding RX windows [99], antenna sensitivities [141], and $\text{SINR}_{\text{Required}}$ [59]

SF	RX windows (ms)	Antenna sensitivity (in dBm)	Required SINR for ADR (in dB)
SF7	5.1	-124	-7.5
SF8	10.2	-127	-10
SF9	20.5	-130	-12.5
SF10	41.0	-133	-15
SF11	81.9	-135	-17.5
SF12	163.8	-137	-20

At each iteration of the ADR algorithm, the transmission power and the spreading factor are modified according to a value denoted by *Nstep* , which is defined as: $\text{Nstep} := \text{round}(\text{SINR}_{\text{margin}}/3)$. *Nstep* corresponds to the number of steps to perform. If *Nstep* is negative (i.e., measured SINRs are low) then the transmission power is incremented by 3 dB until it reaches the maximum transmission power (14 dB), otherwise the

spreading factor (SF) is decreased at each step. If the limit (SF7) is reached and there are still steps remaining, then the transmission power is decreased by 3 dB until the minimum transmission power (2 dB) is reached.

Note that while the ADR algorithm rules seem intuitive (increase power or decrease data rate in case of low SINRs, and the opposite in case of large SINRs), they are heuristics, and not based on any explicit objective optimization. By contrast, this chapter, we intend to optimize specific performance metrics by playing on the SF and power choice. We analyze the performance of ADR, but also of methods specifically designed to optimize metrics in an unknown environment, namely, the multi-armed bandit algorithms described in the next section.

6.3 Decision Making Using Multi-Armed Bandits

The LoRa network currently uses ADR, a partially decentralized heuristic to tune the SF and transmission power of nodes. The values of these two parameters induce a trade-off between energy consumption and packet losses. In this chapter, we claim that this trade-off can be handled using multi-armed bandit algorithms.

The arms correspond to a discretization of the parameter space, namely of the pair (transmission power,SF). We consider a limited number of possible pairs, each one corresponding to an arm (cf. Table 6.2). This means that each time an arm is selected, the transmission power and the spreading factor are also selected.

Table 6.2: The parameters of each arm

Arm	1	2	3	4	5	6	7	8	9
SF	7	7	7	7	8	9	10	11	12
TX power (dB)	2	6	10	14	14	14	14	14	14

The trade-off between energy consumption and packet losses is expressed through a *cost* metric, that is a weighted sum of the energy cost and a loss cost. Mathematically we define the cost perceived at each turn (i.e., each time the node wants to send a new packet) as:

$$\text{Cost} := \text{Energy} \cdot \text{Nb\_transmission} + \text{penalty} \cdot \mathbf{1}\{\text{failure}\} \tag{6.3.1}$$

where:

- Energy is the energy cost for one packet emission. It equals the product of the emission duration (which depends on the SF) and the transmission power;
- Nb_transmission represents the number of transmissions to send the packet: it is an integer between 1 (the first emission is a success) and 8 (since there are a maximum of 7 re-transmissions);

- $\mathbf{1}\{\text{failure}\}$ equals 1 if the 8 transmissions of the same packet fail, i.e., the packet is lost. It equals 0 otherwise.
- penalty corresponds to the conversion of a packet loss into the same cost unit as the energy cost. Its choice is specific to the application and to user preferences: a large penalty value means that a high quality of service is needed–for instance for healthcare applications–even if the energy cost is significant; by contrast a small value for the penalty corresponds to applications where the lifetime of the connected device is favored over having highly reliable transmissions.

Since bandit algorithms function with rewards instead of costs, we first normalize cost values with respect to the largest possible cost (highest power and lowest data rate, and transmission failure after 8 emissions), to obtain normalized values Cost_N in the interval $(0, 1]$, then we define the reward of each decision step as:

$$\text{reward} := 1 - \text{Cost}_\text{N}. \tag{6.3.2}$$

6.4 LoRa transmission model and simulator

We describe here the MATLAB realistic LoRa network simulator we use to perform capacity studies of the LoRaWAN technology [141].

6.4.1 Collision rules

In telecommunications, the Received Signal Strength Indicator (RSSI) is a measurement of the power level of a received radio signal [125]. The Signal-to-Interference-plus-Noise Ratio (SINR) is also an important metric of the wireless link quality [74], since it directly affects the bit error rate in the transmissions.

A *collision* occurs when two LoRa frames are received simultaneously. There are two types of collisions: Inter-SF collisions and Intra-SF collisions, which are modeled according to the two following rules [141]:

- Intra-SF collisions: if a collision occurs between two LoRa frames with the same SF on the same frequency, then only the LoRa frame with the highest power can be decoded, and it is if and only if the power difference exceeds 6 dB (otherwise it is lost).
- Inter-SF collisions: if a collision occurs on the same frequency between two LoRa frames "a" and "b" with different SFs, then packet "a" is demodulated only if: $\text{RSSI}_a - \text{RSSI}_b > \text{SINR}_a$.

6.4.2 Propagation Model

We use the Okumura-Hata model [40, 67] because it is one of the most popular and accurate models, especially used for urban and suburban areas. It is generally applied for frequencies in the range of 150 MHz-1920 MHz, for a distance separation ranging from 1 km to 100 km, and for antenna heights from 30 m to 1000 m [9]. We consider typical indoor penetration losses, with an additional 6 dB loss for deep indoor environments [141, 49, 121].

6.4.3 Shadowing and Fast Fading modeling

Fading is the term used to describe the fluctuations in a received signal as a result of multipath components. We model it using a Rayleigh distribution for the amplitude, hence an exponential distribution on power. Shadowing represents the fact that the received signal power fluctuates due to objects obstructing the propagation path between transmitter and receiver when fast fading is characterized by rapid fluctuations over very short distances. In our LoRa network simulator the shadowing effect is modeled through a log-normal distribution with a 12 dB standard deviation outdoor and 6 dB standard deviation for indoor applications [141, 54].

6.4.4 The relationship between data rate and spreading factor

The data rate DR equals: $\mathrm{DR} = \mathrm{SF} \cdot \frac{\mathrm{BW}}{2^{\mathrm{SF}}} \cdot \mathrm{CR}$, with:

- SF: the spreading factor (an integer between 7 and 12),
- BW: the bandwidth,
- CR: the coding rate.

6.4.5 Transmission and re-transmission

Following each uplink transmission the end-device opens two short receive windows in order to receive a downlink message from the server as acknowledgment of its uplink message. The receive window start times are defined using the end of the transmission as a reference [99]. A node can receive a message only when one of these two windows is open. If no acknowledgment is received after closing the second window, then the message is re-transmitted under certain conditions. Indeed, the number of re-transmissions is limited, generally to a maximum of 7 re-transmissions (what we consider in our simulator), but this number may differ depending on the end-devices. The duration of the reception window depends on the spreading factor (cf. Table 6.1).

During re-transmissions 3, 5, 7 (if any) the node increases its spreading factor (decreases the data rate) before sending the packet again. The rule used to increase the spreading factor is defined as: $\min\left(\mathrm{SF}_{\mathrm{node}} + 1, \mathrm{SF12}\right)$.

Both transmission and re-transmission must respect the duty cycle, defined as the maximum percentage of time during which an end-device can occupy a channel and is a key constraint for networks operating in unlicensed bands. For instance, the duty-cycle is 1% in EU 868 for end-devices [3].

In addition to the cases of collisions seen above, a re-transmission can occur if the gateway receives the frame with an RSSI strictly below the antenna sensitivity. In all other cases, the transmission is assumed to succeed. Mathematically, the RSSI is computed as:

$$\mathrm{RSSI} = \mathrm{tx}_{\mathrm{power}} \cdot \mathrm{Rayleigh}_{\mathrm{power}} / \mathrm{PL} \tag{6.4.1}$$

with $\mathrm{Rayleigh}_{\mathrm{power}}$ a random variable following an exponential law (with mean 1). PL represents the path loss predicted by the Okumura-Hata model and $\mathrm{tx}_{\mathrm{power}}$ is the transmission power. The sensitivity of the antenna depends on the spreading factor (cf. Table 6.1).

6.5 Experiments

6.5.1 Experimental setup

We consider that the network works in the LoRa European band 863-870 MHz, and we use the 868 MHz frequency channel. The frame size is 11 bytes (4 bytes of payload for the consumption index + 7 bytes Zigbee Cluster Library application protocol overhead, corresponding to a smart metering application. Times on air for each SF are calculated using Semtech LoRaWAN specifications [99] and are summarized below:

SF	7	8	9	10	11	12
Time on air (s)	0.04	0.07	0.14	0.25	0.49	0.99

Our simulations are for a network consisting of one gateway and 100 end-devices. In order to consider the worst case, devices are supposed to be located deep indoor.

The experiment runs over 1000 time slots (30 minutes per time slot). Each node sends one packet per time slot to the gateway. While respecting the duty cycle constraint, it can send it up to 8 times (i.e., 7 retransmissions) until receiving an acknowledgment from the gateway. We select a penalty of 1 for handling the trade-off between the energy consumption and the packet loss, and 9 arms corresponding to 9 couples [spreading factor, transmission power] as summarized in Table 6.2.

We consider devices with a -5 dBi antenna gain, which corresponds to the reality of LoRa devices on the market [141].

The first experiment considers a single node to be optimized at different distances from the gateway: 592 m, 1000 m and 1975 m, the 99 remaining nodes following the ADR algorithm. As the transmission between a node and the gateway may interfere with the transmissions of other nodes, the nodes are not independent. The changes of transmission parameters (due to ADR) of the other nodes can change the distribution of rewards for the node of interest, hence an a priori non-stationary environment. The second experiment considers that at time step 500 the node moves from 592 meters to 1975 meters from the gateway. This moving node introduces a switch of the best arm (cf. Fig. 6.5.1).

We compare the simulated standard ADR algorithm with 7 different multi-armed bandit algorithms: UCB [14] and Thompson Sampling (TS) [135] designed for stationary environments, Sliding Window UCB (SWUCB) [106], Switching Thompson Sampling (STS) [105], and Switching Thompson Sampling with Bayesian Aggregation (STSBA) [4] designed for switching environments, and EXP3 [16], REXP3 [22] for adversarial environments.

6.5.2 Simulation Results

Tables 6.3, 6.4, and 6.5 show the energy consumption, the number of lost packets, and the cumulative cost (averaged over 20 simulations) for a node at distance of 592 m, 1000 m,and 1975 m from the gateway, respectively, for different SF, $\mathrm{tx}_{\mathrm{power}}$ selection algorithms (ADR and various bandit algorithms).

The total cost value is defined as: $\sum_{i=1}^{t} \mathrm{Cost}_N(i)$, where t represents the current time slot, and $\mathrm{Cost}_N(i)$ the normalized cost at time slot i (cf. Section 6.3).

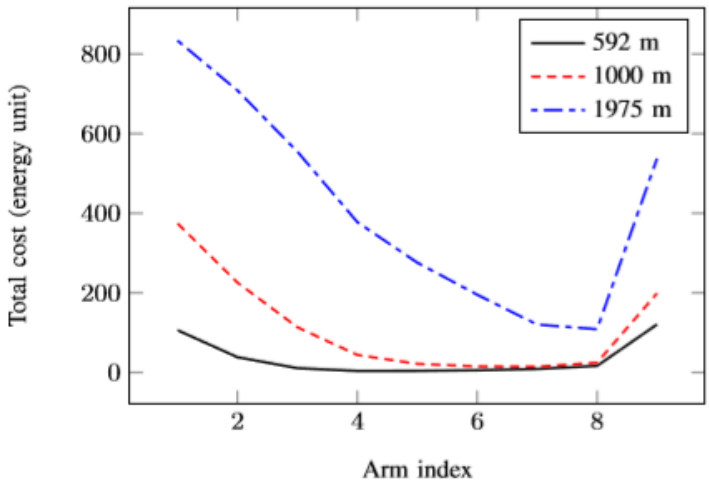


Figure 6.5.1: Average costs for the 9 arms, for three different distances from the gateway: depending on the distance of the node to the gateway, the best arm to play is not always the same.

First, note that for all distances, the MAB algorithms have a total cost below that of the ADR algorithm (cf. Tables 6.3, 6.4, and 6.5). Second, the ADR algorithm is dominated, both in terms of energy consumption and packet loss, by the multi-armed bandit algorithms whatever the distance of the node: hence for any penalty value (i.e., any application an user preferences) applying a MAB algorithm instead of ADR guarantees a cost reduction.

The multi-armed bandits based on a switching environment clearly outperform the adversarial bandits and slightly outperform stationary bandits (cf. Tables 6.3, 6.4, 6.5), which is an indication of the slow evolution of the stochastic environment.

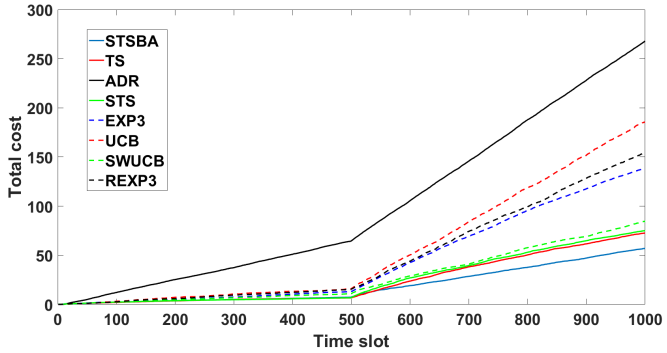


Figure 6.5.2: Total cost versus time averaged over 20 trials, when at time step 500 the node moves from 592 meters to 1975 meters from the gateway.

When the node moves at step 500, a clear switch is introduced (cf. Fig. 6.5.2). ADR, which not handles moving node is clearly dominated by multi-armed bandit algorithms. The best-performing multi-armed bandit algorithm is Switching Thompson Sampling with Bayesian Aggregation. Surprisingly, Thompson Sampling algorithm performs as well as Sliding Window UCB and Switching Thompson Sampling, which are designed for switching environments. Adversarial algorithms explores too much to be competitive in this stochastic environment, and UCB algorithm is the worst bandit algorithm in this non-stationary environment.

Table 6.3: Performance for a node located 592 m from the gateway

Algorithm	Energy (J)	Packet loss	Total cost
ADR	2.07	130	132.07
UCB	**1.24**	15	16.24
TS	7.91	**5**	12.91
SWUCB	1.28	14	15.28
STS	5.53	**5**	**10.53**
STSBA	8.66	**5**	13.66
EXP3	7.81	18	25.81
REXP3	12.10	23	35.1

Table 6.4: Performance for a node located 1000 m from the gateway

Algorithm	Energy (J)	Packet loss	Total cost
ADR	5.20	362	367.2
UCB	**1.96**	49	50.96
TS	7.91	5	12.91
SWUCB	1.99	40	41.99
STS	8.67	15	23.67
STSBA	7.01	**4**	**11.01**
EXP3	11.52	71	82.52
REXP3	8.48	91	99.48

Table 6.5: Performance for a node located1975 m from the gateway

Algorithm	Energy (J)	Packet loss	Total cost
ADR	33.90	553	586.33
UCB	3.29	135	138.29
TS	15.23	61	76.23
SWUCB	**3.21**	105	108.21
STS	25.18	88	113.18
STSBA	14.04	**44**	**58.04**
EXP3	30.59	224	254.59
REXP3	18.25	301	319.25

6.6 Conclusions and Perspectives

In this chapter, we suggest to optimize the performance of uplink LoRaWAN communications by replacing the standard ADR algorithm with multi-armed bandit algorithms to select both the spreading factor and the transmission power. The experiments are carried out with a simulator that meets the standards of the LoRaWAN technology and are performed on nodes located at different distances from the gateway and located in a deep indoor environment.

Simulation results show that the ADR algorithm has a tendency to perform quite well in terms of energy consumption, but incurs large packet losses. All our experiments suggest that the multi-armed bandit algorithms outperform the ADR algorithm, and can be tuned to reach a compromise between energy consumption and packet loss.

As directions for future research, we plan to investigate the case of several gateways, that correlate their received signals to improve packet reception, as well as the interactions among multiple node selfishly optimizing their communications are worth considering.

Chapter 7

TSCH Network optimization via multi-armed bandits

Overview of the chapter

The Industrial Internet of Things (IIoT) faces multiple challenges to achieve high reliability, low-latency and low power consumption. The IEEE 802.15.4 Time-Slotted Channel Hopping (TSCH) protocol aims to address these issues by using frequency hopping to improve the transmission quality when coping with low-quality channels. However, an optimized transmission system should also try to favor the use of high-quality channels, which are unknown a priori. Hence reinforcement learning algorithms could be useful.

In this chapter, we perform an evaluation of 9 Multi-Armed Bandit (MAB) algorithms–some specific learning algorithms adapted to that case–in a IEEE 802.15.4-TSCH context, in order to select the ones that choose high-performance channels, using data collected through the FIT IoT-LAB platform. Then, we propose a combined mechanism that uses the selected algorithms integrated with TSCH. The performance evaluation suggests that our proposal can significantly improve the packet delivery ratio compared to the default TSCH operation, thereby increasing the reliability and the energy efficiency of the transmissions.

Publication. This chapter is mainly based on our article [39].

7.1 Introduction

The Internet of Things (IoT) has gained a considerable attention recently. With the overgrowing of IoT, there is a heavy focus on evolving and establishing new protocols with high performance and energy savings. For instance, Industrial IoT (IIoT) networks require a very high reliability close to 99.9% and ultra-low delay performance. To this end, standards such as IEEE 802.15.4 [71], WirelessHART [129] or ISA100.11a [72] employ Time Division Multiple Access (TDMA) in conjunctions with Frequency Division Multiple Access (FDMA) scheme to guarantee high level of reliability, by introducing the slotted time to mitigate the collisions, and enabling channel hopping to reduce external interferences.

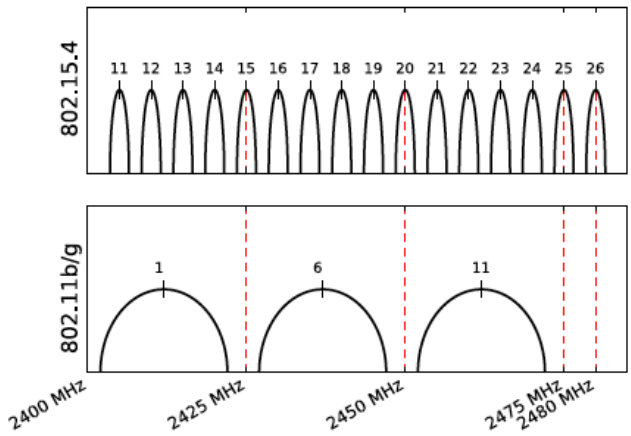


Figure 7.1.1: Interfering radio channels: IEEE 802.15.4 and IEEE 802.11: 1, 6 and 11 are the three non-overlapping and broadly used radio channels for IEEE 802.11 technology [148].

Two physical phenomena affect the reliability of wireless networks. The first is external interference, that occurs in the case of the operation of two or more technologies in the same radio space, causing thus, packet losses, i.e., interference between IEEE 802.11 and IEEE 802.15.4 radio channels as shown in Figure 7.1.1. The other is multi-path fading, which is caused by the reception of different copies of the same signal coming from different paths. Channel hopping is a well-known approach to mitigate such issues; if a transmission failure occurs, a re-transmission takes place on a different frequency [144].

However, there is no guarantee that the following radio channel selection will be better than the previous one and, thus, a new failure may occur. To overcome such issues, an intelligent algorithm is required to support the sensor devices to select the best link quality radio channel to achieve higher success probability.

In this chapter, we introduce Multi-Armed Bandit (MAB) algorithms to select the channel to transmit on, in the context of IEEE 802.15.4 Time Slotted Channel Hopping (TSCH) networks. MAB algorithms are used in problems where multiple choices each with unknown reward are available for a user who must select only one in such a way the cumulative reward is maximum. They handle the exploitation-exploration dilemma to select the best choice. In this work, we apply the bandit algorithm that shows the best performance to identify the best radio channel among a set of channels available to a given node in TSCH networks.

The contributions of this work are as follows:

- We first thoroughly evaluate a large set of bandit algorithms by employing realistic traces from the FIT IoT-LAB testbed [89], using a simulator built on Matlab.
- We then select the best bandit algorithm, the one that estimates correctly the high-quality channels, and propose a new TSCH-based radio channel selection pattern.
- Finally, we evaluate the performance of our proposed mechanism in comparison with the IEEE 802.15.4-TSCH behavior according to the standard.

The chapter is organized as follows. Section 7.2 provides a background on IEEE 802.15.4-TSCH networks, and reviews the most pertinent related works from the literature. An

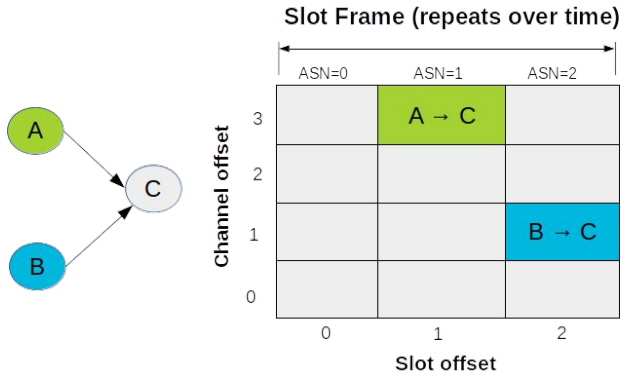


Figure 7.2.1: Scheduling process in IEEE 802.15.4 TSCH networks.

overview on multi-armed bandit problems and algorithms are presented in Section 7.3. Section 7.4 describes our Matlab-based simulator to evaluate the performance of the 9 bandit algorithms and demonstrate the results. We present our new channel selection mechanism based on multi armed bandit algorithms and compare its performance with the default behavior of TSCH networks in Section 7.5. Finally we conclude in Section 7.6.

7.2 Background & Related Work

7.2.1 Background on TSCH mechanism

Industrial IoT applications require reliable communication, low-power operation and robustness against potential external interference. Among the existing standards, IEEE 802.15.4 is the most appropriate candidate to guarantee QoS requirements for such industrial networks. It comes with 16 radio channels operating in 2.4GHz, where each channel has a bandwidth of 2MHz and channel separation of 5MHz (see Figure 7.1.1).

Among the defined Medium Access Control (MAC) protocols in IEEE 802.15.4, TSCH aims for high reliable and low power multi-hop networks. Under TSCH, the communication among the nodes are coordinated by a scheduler, while the nodes are synchronized with their neighbors. These schedules can be adjusted to the industrial network's requirements and its topology and guarantee collision-free communications.

In TSCH, the nodes continuously re-synchronize on a periodic slotframe, based on Enhanced Beacons (EB) packets. A slotframe is a collection of timeslots that are repeated continuously; according to the standard it consists of 101 timeslots but it is configurable. The length of a timeslot is long enough for the source node to send a packet, and for the destination node to send an acknowledgment. If the acknowledgment is not received within a predefined timeout, the re-transmission of the packet will be delayed to one of the following timeslots. At each timeslot, each node knows if it has to stay awake in order to transmit or receive a packet, or to sleep to save energy. The timeslots are identified by an Absolute Sequence Number (ASN) counter that increments as time elapses; ASN basically counts the number of timeslots since the network was started.

Furthermore, each node maintains a list with the available radio channels called Hopping Sequence List (HSL) [133]. It is the list of the 16 available radio channels. During the scheduling process (see Figure 7.2.1), each node computes the actual radio channel to transmit on according to the following relation:

$$\text{Channel} = \text{HSL}[(\text{ASN} + \text{ChOffSet})[|\ \text{mod HSL}]|] \tag{7.2.1}$$

where HSL[i] is the channel with index i in the list HSL, |HSL| is the number of channels in the HSL, and ChOffSet (channel offset) is an integer between 0 and |HSL|-1 assigned to the communication in the node's schedule. Authors in [62] interpret the channel offset as a virtual channel to be translated into an actual frequency that is going to be employed. In case of failure, a re-transmission will be carried out using a channel also selected according to (7.2.1).

In order to avoid internal interference, nodes can be assigned different channel offsets (translated to real frequencies) from the pair neighbor nodes communicating in the same time-slot to ensure that they are communicating on different channels. An illustration of this channel selection method is provided in Figure 7.2.2.

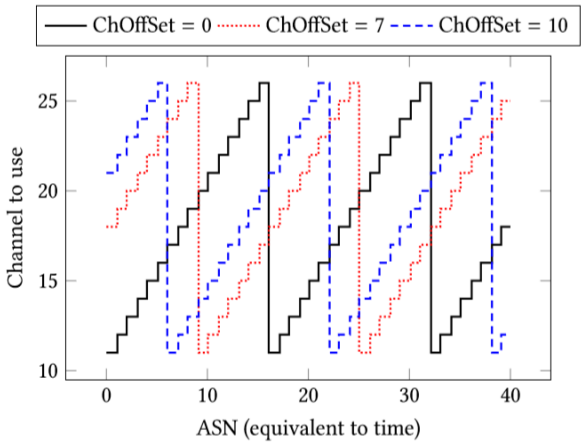


Figure 7.2.2: Illustration of (7.2.1) when HSL is made of channels 11 to 26. In classical TSCH, a node is given one ChOffSet, while in our proposition, and as suggested in [62], a node may be allocated several values.

However, TSCH is still suffering from external interference due to the coexistence of technologies operate in the ISM band such as WiFi, Bluetooth, Zigbee [89, 114, 115]. Therefore, the concept of avoiding (possibly by blacklisting) poor radio channels was introduced to improve TSCH performance. In the following subsection, we present the works that tackled this issue.

7.2.2 Related Work on channels selection in TSCH

The channel hopping mechanism of TSCH allows potentially to address the impact of external interference and multi-path fading, by avoiding being stuck with one bad channel. However, in the process of selection of a new radio channel, we might end up with another affected frequency.

To address this, some works have focus on identifying the radio channel with poor performance based on Window Mean with Exponentially Weighted Moving Average (WMEWMA) estimator [19] and blacklist them from the list of available radio channels [90], They propose heuristics to decide which channels to blacklist considering the dynamic, unpredictable and highly localized nature of interference. They take also into account a mechanism to

whitelist the channels when they become reliable again. Achieving thus, to improve the network reliability by 20%.

In [62], Gomes et al. considered learning techniques approach as a way to tackle the channel quality estimation problem. They propose to model the channel quality estimation as a MAB problem, they employ a distributed blacklist for improving the performance of multi-hop wireless networks, as result they find the best channel in 75% of the transmissions.

In [96], channels selection in TSCH networks was seen as a multi-armed bandit problem. They proposed a channel selection scheme which adapts to the environment and finds the optimal channel. The selection of each channel was associated with the so-called Gittins index [60]. This index is incremented with each successful transmission, set to zero in case of disabled transmission due to assesment of busy channel (Clear Channel Assesmment CCA) or consecutive transmission failures. Their algorithm adapts to the environment to choose exploring or exploiting the channels. They showed that by employing their algorithm, the throughput can achieve approximately 1.5 times larger than hopping with the default list.

Multi-armed Bandit problem have been applied to model others decision approaches not just in TSCH networks but also in LoRa networks [83]. In this case, authors aim to optimize the performance of LoRAWANs using MAB algorithms to select the communication parameters such as spreading factor and emission power so that a trade-off can be accomplished between energy consumption and packet loss. They evaluated number of multi-armed bandit algorithms, finding that Switching Thompson Sampling with Bayesian Aggregation (STSBA) [4] outperforms the others given the best trade-off.

Avrachenkov et al. in [18] consider a wireless network where transmitters can select a frequency band from a shared pool to communicate, the transmitters are affected by slow fading channel phenomena. In this case, the main goal is to maximize the Signal to Noise Ratio (SNR), thus, they use a multi-armed bandit algorithm, Thompson Sampling [81], to deal with exploration-exploitation trade-off to select the frequency that provides maximization of SINR.

7.3 Learning Techniques

Multi-armed bandit problems are the most known examples of sequential decision problems with an exploration-exploitation trade-off. This is the balance between staying with the option that gave highest rewards in the past and exploring new options that might give higher rewards in the future.

Normally, a set of arms is available to the "gambler". At each turn, that gambler has to choose one arm and receives a reward corresponding to the played arm. For the next turn, using the previously received rewards, the gambler has to take the decision of exploring, that is playing an arm whose mean reward is loosely estimated in order to build a better estimate, or exploiting, that is playing a seemingly best arm based on current estimates in order to maximize its cumulative reward. The difference between the chosen strategy and an optimal strategy, which always chooses the best arm, is called *regret* and measures the accuracy of the gambler's policy.

The channel selection in TSCH networks can be formulated as a multi-armed bandit problem. The 16 channels are the 16 arms to play. Playing a channel means selecting it to send a packet. The reward is either 0 or 1, where 0 corresponds to a transmission failure and 1 to a

success. The aim is to increase the cumulative gain of the network, which is the summation of the rewards.

Multi-armed bandit algorithms are algorithms used to solve such problems. They try to find the best arm to play at each iteration according to the rewards received so far, in order to maximize the cumulative gain. The algorithms can be classified into three categories according to the environment.

7.3.1 Stochastic static environment

One of the strongest assumptions of many statistical models, including most variants of the multi-armed bandit problem, is that the underlying distributions and parameters are stationary.

In this setting, each arm is associated to an unknown and constant distribution for which rewards are considered to be generated independently. Thompson Sampling [81] and Upper Confidence Bound (UCB) [14] are two algorithms that reach optimal upper-bound on the cumulative regret under that assumption of stationarity.

Thompson Sampling assumes a prior distribution for the reward distribution of each arm, and at each time step, plays an arm according to its posterior probability of being the best arm. The reward distributions are assumed to have two possible values 0 or 1, known as Bernoulli bandit formulation. This algorithm is studied also for switching environments, which are explained next.

7.3.2 Stochastic switching environment

In a switching environment, when a switch occurs, all arms change their expected reward. This approach behaves as piece-wise stationary, since the mean rewards stay stationary between those changes.

For this setting, Sliding Windows UCB [106] has shown near-optimal regret bound. Switching Thompson Sampling combined with a Bayesian online change point detector has been used for handling switching environments [105]. This algorithm is based on a growing number of experts: at each time step, a new expert is introduced. To take the decision, the expert with the highest likelihood is sampled according to its weight. Then, the Thompson Sampling related to the chosen expert is launched to choose an arm. Finally, based on the reward observed, the weight of each expert is updated.

Finally, let us evoke Switching Thompson Sampling with Bayesian Aggregation (STSBA) [4], which uses Bayesian aggregation of experts instead sampling the best expert, to avoid the sampling noise.

7.3.3 Adversarial environments

In an adversarial environment, the sequence of rewards is assumed to be chosen in advance by an oblivious adversary. The EXP3 algorithm uses a follow-the-perturbed-leader approach for computing the probability of each action [16]. It achieves an optimal regret with respect to the best policy that pulls the same arm over the totality of the game. This weakness is overcame by EXP3.S [16], that forgets the past adding at each time step a proportion of the mean gain and achieves controlled regret with respect to policies that allow arm switches during the run. To compete against an optimal policy that changes over time, EXP3.R [6]

uses a statistical test to reset the EXP3 algorithm. REXP3 [22] simply resets the EXP3 algorithm after a time period.

7.4 Performance Evaluation

In this section, we describe the different scenarios on which we evaluate the multi-armed bandit algorithms. We also describe how to build scenario simulations using real data.

7.4.1 Principle and scenarios considered

The evaluation of the bandit algorithms is based on two different scenarios referring to two different environments. Our evaluation criterion is the Packet Data Rate (PDR), which is the proportion of packets successfully received (number of received acknowledgments divided by number of transmitted packets).

Our first scenario corresponds to a *stationary environment*: it assumes that the channels' qualities are constant over time, while our second scenario considers a stochastic switching environment. In both those scenarios, at any instant each channel has a given quality, that we summarize by the probability of a successful transmission if the channel is chosen; we will also call PDR that probability.

To summarize, based on the unknown PDR values of the channels (which can vary over time in a switching environment), the algorithms have to learn which channels to use to maximize the experienced PDR for a device implementing them.

7.4.2 Building a simulation from real data

In order to build a realistic experiment (in terms of PDR values and channel dynamicity), we used the data collected by the authors of [89] during their experimental studies in the FIT IoT-LAB platform in Grenoble site which is a part of the FIT, an open large-scale and multiuser testing infrastructure for IoT-related systems and applications. The experiments were conducted with a testbed of 380 nodes subjected to external interference originated from wireless devices using other technologies, such as WiFi. The data used is the result of 276 experiments, of 90 min each, done on two communicating nodes separated by distances varying between 0.6 and 16.8 m. The transmissions are done every 3 sec in timeslots of 30 ms each. For each packet sent, the data give its ASN, the channel that was used, and whether the transmission was successful or not.

Figure 7.4.1 summarizes the observed PDR per channel from those experiments, where each point corresponds to more than 4600 packets sent.

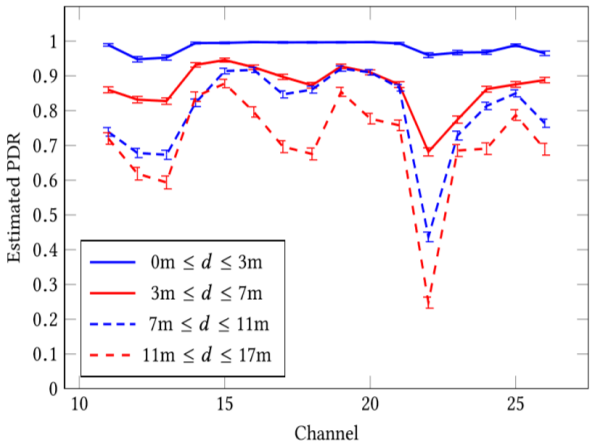


Figure 7.4.1: Average PDR (and 99% confidence intervals) per channel for different ranges for the distance d between nodes. Note how bad channels are generally the ones overlapping with 802.11 channels (see Figure 7.1.1).

These experimental data are used to build up our experiments as explained in the following subsection.

7.4.3 Estimating time-varying channel PDRs

Channel quality varies over time, but the limited number of samples imposes limits on the dynamicity (the speed at which PDRs change) we can consider: indeed, we use the measures to estimate the PDR of each channel over given periods, during which PDRs are treated as constants. Hence a trade-off between estimating the PDR accurately enough (through enough samples, hence over a long period), and encompassing the time-varying aspect of the PDR. To address that trade-off, we use confidence intervals: for instance, a 90%-confidence interval for the PDR when an estimation $\widehat{\text{PDR}}$ is built over N (Bernoulli) samples is $[\widehat{\text{PDR}} - 1.645\frac{\sigma}{\sqrt{N}}, \widehat{\text{PDR}} + 1.645\frac{\sigma}{\sqrt{N}}]$, with $\sigma = \sqrt{\text{PDR}(1-\text{PDR})}$ the standard deviation of the underlying Bernoulli variable. Hence the distance between $\widehat{\text{PDR}}$ and PDR is below 0.05 with probability at least 0.9 if $1.645\frac{\sigma}{\sqrt{N}} \leqslant 0.05$, or $N \geqslant \sigma^2 \left(\frac{1.645}{0.05}\right)^2$. But for any value of PDR, $\sigma \leqslant \frac{1}{2}$, so it is sufficient to have

$$N \geqslant \left(\frac{1.645}{0.1}\right)^2 \simeq 270 \text{ samples.}$$

Of course, a higher accuracy would impose having more samples, for example a 99% guarantee that the PDR is correctly estimated within 0.01 necessitates, using the same methodology, $N \geqslant (\frac{2.576}{0.02})^2 = 16590$ samples.

7.4.4 Simulation Setup

Stationary environment

For the stationary environment, we defined a vector of 16 values presenting the average PDR on each channel at a certain distance (we chose the largest-distance range of Figure 7.4.1). For those given fixed expected PDR (unknown to the algorithms), we ran each of the algorithms 40 times separately at the transmitter node.

The algorithms select one channel among the 16 radio channels every 3 seconds (slot frame duration), then simulate a packet emission, and get the reward which is 1 with probability equal to the PDR of the chosen channel (representing a successful transmission) or 0 with the complementary probability (representing a transmission failure).

For each iteration, the node sends 300 packets, hence a simulated duration of 15 min.

Switching environment

On the other hand, for the switching environment, we separate the data from 6 experiments for the same distance range, and define a matrix consisting of the PDRs on each channel for each of the 6 experiments, i.e., assuming that the link qualities change 5 times during the simulation. The same simulations as in the stationary environment were done with 300 packets (15 min) in total, which leaves only 50 packets (2.5 min) to learn the channel qualities before they change.

Additionally, in order to compare these algorithms with the default behavior of TSCH, we need to simulate the latter also. For this, in the same settings of the two environments (stationary and switching), the node is assigned a certain channel offset for the whole simulation. Then for each packet, the node selects the radio channel according to (7.2.1), with |HSL|= 16 (16 available channels), sends the packet and receives the reward (success or failure).

7.4.5 Simulation Results

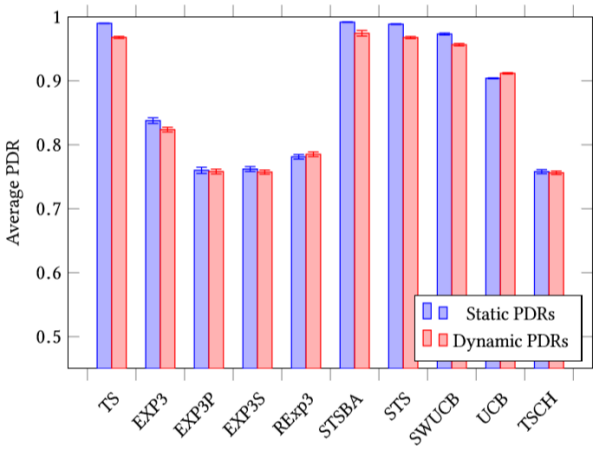


Figure 7.4.2: Average PDR per algorithm, based on experimentally-decided channel PDRs, and 95% confidence intervals.

Our results for channel PDRs estimated from real-experiment data are summarized in Figure 7.4.2, where we plot the average success rate experienced by the transmitting node, for each channel selection method (note that all of them are multi-armed bandit algorithms except the benchmark TSCH method). The experiments were repeated independently 40 times in order to compute confidence intervals. The best-performing methods for our setting are Thompson Sampling, Switching Thomspson Sampling, Switching Thompson Sampling with Bayesian Aggregation and Sliding Window UCB, which have almost the same average experienced PDR and outperform the other methods.

Hence those results are insufficient to isolate one single algorithm to use; we think this is due to several channels having good qualities (about 5 channels have PDRs close to 1), and only one channel being really bad. The limited heterogeneity among channel PDRs does not leave enough margin for the algorithms to differentiate. Therefore, in order to evaluate the algorithms, we will sometimes use artificially set PDR values (rather than based on real experimental data) to have significant differences between the qualities of all the channels. This may correspond to what nodes at a distance above 17m would experience, for which we do not have data.

The algorithms need some time to learn the channels and discover the good and bad ones. But since channel qualities may vary, the algorithms should perform well over a limited time. To estimate that, we ran the simulations with PDRs varying more or less frequently: every 50, 150, or 200 packet emissions. As before, we consider 6 different values (5 PDR switches over a simulation). The results (average PDR per algorithm) are shown in Figure 7.4.3.

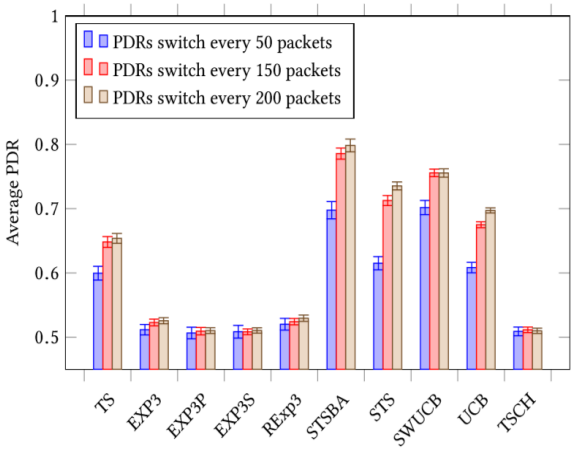
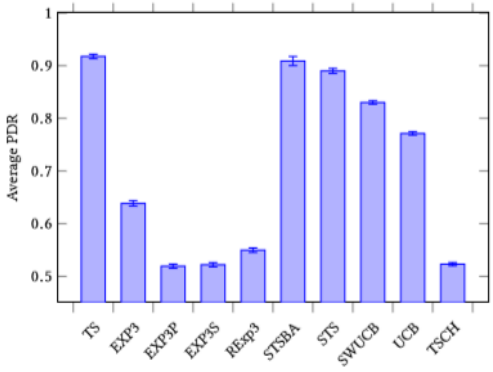


Figure 7.4.3: Average PDR per algorithm (switching environment) over 5 PDR value switches, for different switching frequencies (artificially generated PDRs), with 95% confidence intervals.

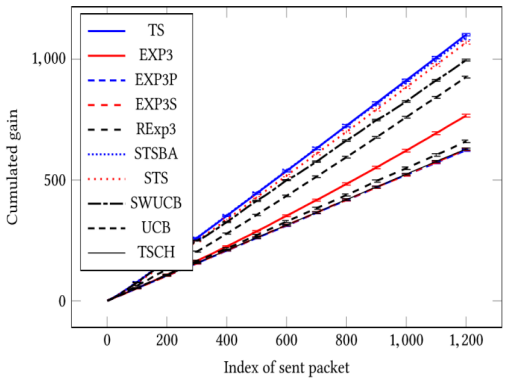
As expected with learning methods, as the number of packet emissions before a switch increases, the average PDR increases too, meaning that the algorithms improve their estimation of what channels to use. Also, as expected, that parameter has no impact with the TSCH method, since it does not use the information from transmission successes/failures.

Equivalently, as channel qualities change faster, the algorithms need more time to learn the new qualities and find the best channels. With frequent switches (every 50 packet emissions), SWUCB and STSBA are the two best-performing algorithms. With more time to learn the new PDR values, STSBA shows the best performance. This suggests that STSBA takes more time to learn the channels than SWUCB. In the simulations we will carry out next, we will assume that PDRs do not change too fast: for the switching case we will assume one switch occurs every 200 packet emissions (i.e., every 10 minutes if one packet is sent every 3 seconds).

The average PDR achieved by each algorithm as well as the cumulative gain over time (number of successfully sent packets) in the stationary environment are shown in Figures 7.4.4a-7.4.4b respectively, while Figures 7.4.5a-7.4.5b show results for the switching environment.



(a) Average PDR per algorithm with 95% confidence intervals.

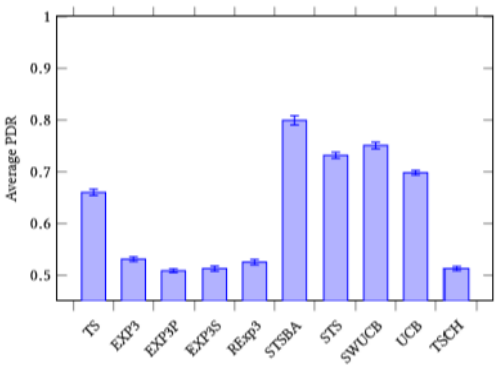


(b) Cumulative gain (number of successfully sent packets) for each learning algorithm (average of 40 trajectories, with 95% confidence intervals).

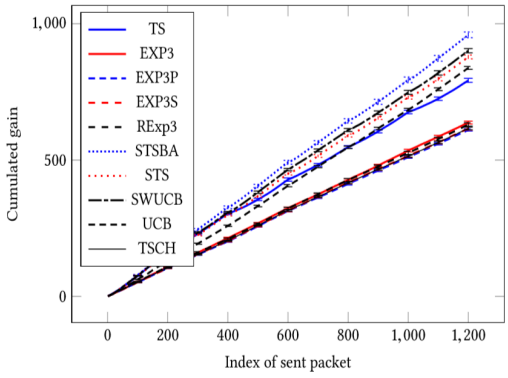
Figure 7.4.4: Performance of channel selection algorithms in a stationary environment (artificial link qualities).

Let us first consider Figures 7.4.4a-7.4.4b. We notice that the Thompson Sampling (TS) algorithm has the highest average PDR (equivalently highest number of successes), slightly outperforming Switching Thompson Sampling with Bayesian Aggregation. This could be expected, since TS is proved to be near optimal. In contrast, the normal behavior of TSCH has the lowest average PDR. To quantify the gain that those simple learning algorithms could bring, we note that TS would increase the average reward by about 75% (from 0.52 to 0.92) in case of 16 available channels, with respect to simply following (7.2.1) as TSCH does. Or equivalently, that the number of lost packets could be divided by 6.

Now let us analyze the results for the switching environment, shown in Figures 7.4.5a-7.4.5b.



(a) Average PDR per algorithm with 95% confidence intervals.



(b) Cumulative gain (number of successfully sent packets) for each learning algorithm (average of 40 trajectories, with 95% confidence intervals).

Figure 7.4.5: Performance of channel selection algorithms in a stochastic switching environment (artificial link qualities).

We can see in Figure 7.4.5b the PDR switching points: the chosen arms perform less well, hence a reduced slope, and the algorithms try to learn again to find the new best-performing channels (exploration) before using them extensively (exploitation). In this environment, STSBA has the highest average PDR. In this setting, the capacity of STSBA of updating

the estimation of the different channels each time an arm is pulled allows it to adapt to switching environments [142]. The algorithm keeps selecting the optimal channel until its quality degrades, indicating a switch may have occurred. Compared to TSCH normal behavior, STSBA would increase the average reward by about 55% (from 0.51 to 0.8), or equivalently divide by 2.5 the packet losses.

Summarizing, the TS and STSBA algorithms have the best performance in the stationary environment, and STSBA is the best in the switching environment. In both environments the MAB algorithms lead to a significant improvement with regard to the TSCH normal behavior.

However, using the MAB algorithms in a network with multiple nodes to select a radio channel among 16 ones to transmit on (rather than the fundamental strategy) might lead to internal interference and collisions since multiple nodes may select the same channel at the same time. For this, we have to adapt the MAB algorithms with the normal behavior of TSCH. That is why a new approach is proposed and discussed in the next section.

7.5 Multi-armed Bandit Algorithms for TSCH-compliant channel selection

7.5.1 Proposed method: MAB algorithm over a set of channel offsets

In this work, we propose to integrate Multi-armed Bandit Algorithms into the TSCH channel selection protocol, while avoiding too much interference. To do so, we will limit the set of channels that the node can use.

The authors of [62] work with the idea of a blacklisting solution where a *set of channel offsets* is initially assigned to each node. The channel offsets are used to generate a list of radio channels using (7.2.1), where a channel is selected at each iteration (each time slot) using one offset from the available set chosen randomly. This process generates a whitelist of channels that are used to transmit. Note that the channel is included in the whitelist just if it is available in the HSL i.e., it's not blacklisted, otherwise if the node does not find any available channel there, it suspends the transmission.

The procedure helps avoiding overlapping between neighbors sending at the same time. Nonetheless, there is a high probability of running the process and getting no available frequency, since the assigned offsets can produce radio channels that have already been blacklisted, representing one of the main drawback of their proposal.

In order to address that issue, Kotsiou et al [148] used single channel offsets with a global blacklisting mechanism and distributing the bad channels to the entire network. Each time a frequency is assigned, it is taken out of the HSL. In this way, the protocol will always generate a whitelisted channel for the node.

We use an approach similar to the one suggested in [62]: in our proposal, each node is assigned a set of channel offsets. At each time slot (ASN) we find the available channels according to (7.2.1): that gives a number of channels equal to the number of offsets. Subsequently, we suggest to use bandit algorithms to select one channel among the allowed ones based on the rewards achieved by each of those channels. This procedure is presented in Figure 7.5.1, where the node is assigned 4 channel offsets $\{1, 7, 5, 13\}$. At ASN=30, the available radio channels are $\{3, 5, 11, 15\}$. In order to transmit, the MAB algorithm selects one of these channels.

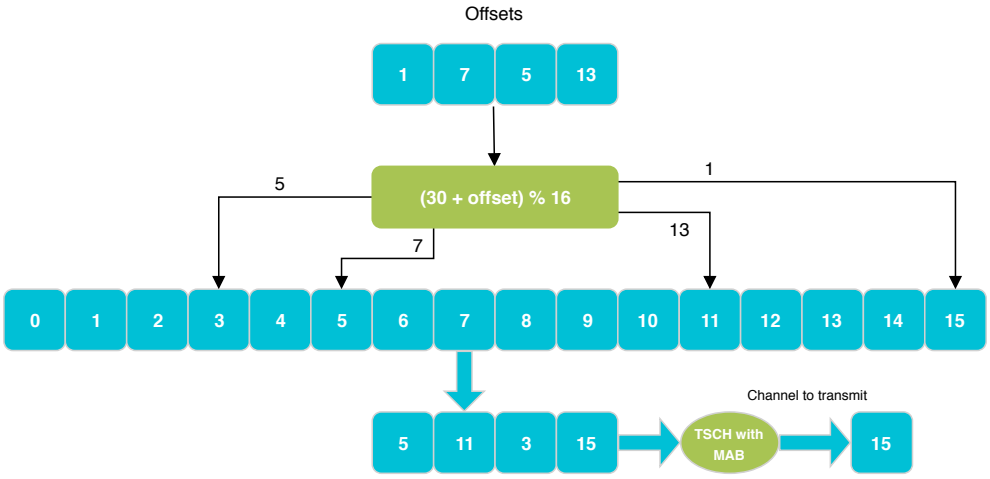


Figure 7.5.1: TSCH combined with Multi-Armed Bandit algorithms.

7.5.2 Performance evaluation

Simulation Setup

To evaluate our proposed mechanism, we use the same setup described in Section 7.4 for both stationary and switching environments. However, since in this setup we assign only a set of channel offsets for each node, at a given instant only a subset of channels can be selected, in contrast to the previous experiments where all 16 channels could be used (equivalent to 16 channels offsets). Here, we restrict the study to the MAB algorithms that proved to have the best performance, i.e, TS and STSBA. We also compare these algorithms with the normal behavior of TSCH.

For this, we assign different channel offsets to each node. For instance, if node A is assigned 4 channel offsets $\{1, 7, 5, 13\}$ as shown in Figure 7.5.1, another neighbor node should be assigned a different list of channel offsets to avoid internal interference. The node will keep these channel offsets through the whole simulation; at each ASN the TSCH protocol generates the corresponding radio channels iterating through the list of offsets, still according to (7.2.1). Then MAB algorithms choose among those available channels (corresponding to the channel offsets) the one to use, while with the normal TSCH behavior the radio channel is chosen randomly.

Simulation results

To see the impact of the number of channel offsets assigned to each node, we run our experiments with different sets of channel offsets.

Figure 7.5.2 shows the average PDR achieved by TS, STSBA and the (reference) TSCH methods in a switching environment, when the channel PDR values vary more or less frequently, and with different numbers of channel offsets assigned to each node.

In these experiments we used the artificial data rather than those conducted from real experiments in order to have significant differences among channels' qualities. As expected, for the learning method TS, the performance improves as the number of channel offsets increases. Also, STSBA brings some performance improvement, in accordance with the results of Figures 7.4.3 and 7.4.5a, but at the cost of a higher computational complexity.

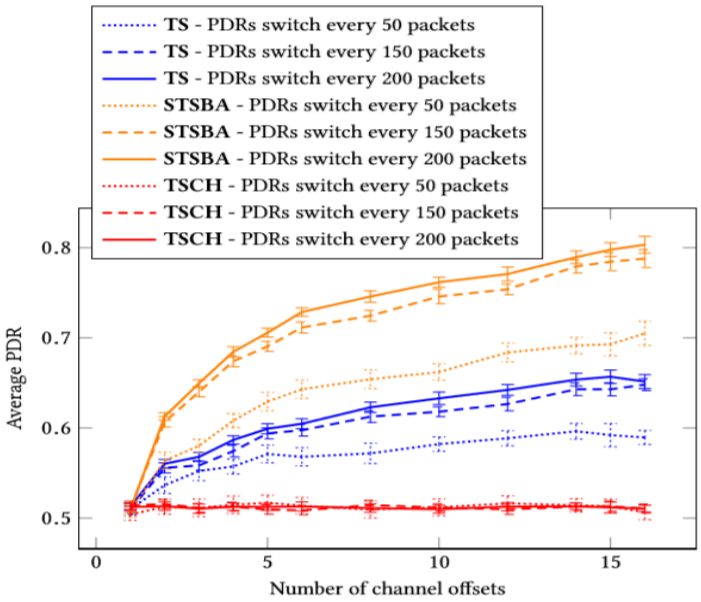


Figure 7.5.2: Average PDRs achieved by TS and STSBA (two bandit algorithms) and the classical TSCH method, with different numbers of channel offsets, in a switching environment (artificial PDR values). 95% confidence intervals are also provided.

For TSCH, as expected the performance remain unchanged when increasing the number of channel offsets, since no clever choice is made on the available channel set. The figure additionally provides some insight regarding an "optimal" number of offsets to allocate, to trade-off individual performance and possible overlapping among the channels that different nodes may use. From the figure, a number of 4 offsets allocated to the node yields a significant performance improvement, while the marginal gain from further increases in the number of offsets may be too small (this is up to the network manager to decide, mainly depending on the node density). In what follows, we consider 4 offsets to be a reasonable choice.

Finally, in order to study the impact of using MABs (specifically TS and STSBA) in real TSCH networks, our experiments were repeated using real-experimental data for both stationary and switching environments. The results are shown in Figure 7.5.3, when 4 channel offsets are assigned to each node. The graph shows that even with realistic channel PDR values (with heterogeneity below what we set when fixing artificial values), applying learning algorithms in stationary or switching environments significantly improves the performance of TSCH networks, by more than 20%. Or equivalently:

- in a stationary setting, using TS with 4 channel offsets instead of TSCH allows to divide by about 14 the number of packet losses;
- in a switching setting, TS and STSBA seem to offer comparable performance, even if STSBA is theoretically more adapted. From our simulations, both lead to a reduction of packet losses by a factor above 3 with respect to the classical TSCH protocol.

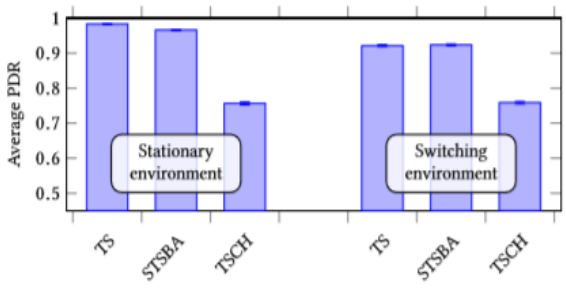


Figure 7.5.3: Average PDR per algorithm for 4 channel offsets with experiment-based link qualities, and 95% (hardly visible) confidence intervals.

7.6 Conclusions

In this chapter, we analyze how multi-armed bandit algorithms can solve the problem of selecting the best channel to transmit in TSCH-based networks. We evaluate 9 multi-armed bandit (MAB) algorithms with both and artificially-generated and experiment-based link quality values, through a realistic simulator covering diverse scenarios in terms of variability of channel qualities over time. That analysis highlights that Thompson Sampling and STSBA, two bandit algorithms, are the best-performing methods in stationary and switching scenarios respectively, yielding significant improvements in terms of average packet success delivery rate.

To also cope with controlling the overlap of the channels that different nodes may use, we propose a new channel selection mechanism for TSCH that integrates a MAB algorithm. The algorithm uses a multi-offset technique, using MAB to select the channel to use among a subset of available channels, determined by the offsets allocated to the node. The comparison between the multi-offset TSCH normal operation and multi-offset TSCH with MAB approach showed a significant improvement with more than 20% in PDR, or a reduction of packet losses by a factor larger than 3 in switching environments, and larger than 10 in stationary ones.

In future works, we consider providing another degree of freedom for nodes, where instead of choosing between a few offsets, if all the corresponding channels are bad a node may decide to delay its transmission to wait for a *good* channel to appear in its channel list.

Part IV

Conclusion

Chapter 8

General conclusion and perspectives

This final chapter summarizes our contributions and presents future work directions.

8.1 Conclusion on our contributions

We started this manuscript by detailing the historical, scientific and technical contexts of our research during this PhD. We explained the problems that motivate and justify our works.

Our work aims at responding to several issues.

The first was the study of the non-stationary bandit problem in the case of a piece-wise stationary environment. In this context we have analyzed in depth the algorithm proposed by Joseph Mellor and Jonathan Shapiro in [105] which consists in combining the Thompson Sampling algorithm with the Bayesian change point detector in order to handle a piece-wise environment. The original algorithm is based on the principle of sequentially selecting the best Thompson Sampling model adapted to the current state of the environment. For that, the authors propose to infer an integer random variable named runlength describing the number of time steps since the last change. Then, the choice of the environment model is made by sampling the current runlength distribution. In our first contribution called "Memory bandit", we revisited this philosophy by rewriting the algorithm as the aggregation of a growing number of experts. Experimentally, expert aggregation gives more satisfactory results than sampling the runlength distribution.

Since the combination of Thompson sampling and the Bayesian online change point detector seems to give very interesting results, we wanted to understand in detail how this change-point detector works, which is renowned for its unmatched experimental performance. Hence our second contribution where we showed the mathematical limits which prevented the community from proving the optimality of the Bayesian online change-point detector algorithm. We thus succeeded in extracting a version almost similar to the original algorithm but easier to mathematically analyze in terms of false alarm rate and detection delay. We show that the false alarm is perfectly controlled and that the detection delay asymptotically reaches the lower bound of the problem of detecting change-points in an online fashion. This represents the first proof of the optimality of a change point detector based on the same principles as the original Bayesian online change point detector.

Regarding our third contribution, decentralized exploration for multi-armed bandits, we were interested in the problem of the asynchronous collaboration of a set of agents for the best arm identification task. This collaboration must respect privacy constraints as well as constraints related to communication between agents. Indeed, for the collaboration between agents to be effective, it would be necessary to guarantee privacy to the agents as well as the

minimization of communication costs. We have thus developed a generic algorithm taking any bandit algorithm for the best arm identification task as a subroutine. We have also shown that our generic algorithm respects the constraints of privacy as well as communication costs remain low.

Finally, we were also interested in showing the importance of bandit algorithms for industrial use cases. On the one hand, we suggested to optimize the performance of uplink LoRaWAN communications by replacing the standard ADR algorithm with multi-armed bandit algorithms to select both the spreading factor and the transmission power. Simulation results show that the ADR algorithm has a tendency to perform quite well in terms of energy consumption, but incurs large packet losses. All our experiments suggest that the multi-armed bandit algorithms outperform the ADR algorithm, and can be tuned to reach a compromise between energy consumption and packet loss. On the other hand, we analyzed how multi-armed bandit algorithms can solve the problem of selecting the best channel to transmit in TSCH-based networks. We evaluate 9 multi-armed bandit (MAB) algorithms through a realistic simulator covering diverse scenarios in terms of variability of channel qualities over time. That analysis highlights that multi-armed bandits yield significant improvements in terms of average packet success delivery rate.

8.2 Future works

The purpose of this section is to present some of the new research directions extending the work presented in this document.

8.2.1 Building the whole memory bandits and non-stationary Markov decision processes

As indicated at the conclusion of Chapter 3, extending the switching Thompson sampling for the rest of the stationary stochastic bandits is feasible. Thus, as future work, we first propose to reuse the principle of the growing number of experts aggregation with other bandit algorithms namely KL UCB, Bayes UCB ... Since we have succeeded in showing the optimality of a version of the Bayesian online change-point detector, we can now think about how to exploit it to mathematically analyze memory bandits in terms of pseudo-cumulative regret upper bound.

We can also extend the use of the Bayesian online change point detector for more complex problems than the bandit namely the Markov decision process (MDP). We can thus extend Q Learning [143] or UCRL [73] type algorithms for non-stationary MDP problems.

8.2.2 Extension of the restarted Bayesian online change-point detection for general distributions

In chapter 4, the formulation and analysis of the Bayesian change point detector were done in a Bernoulli framework (the observations follow a piece-wise stationary Bernoulli law). It would be very interesting to study the extensibility of the formulation for the general case of exponential distributions. We can also be interested in the formulation of the algorithm for multidimensional distributions which are very useful for non-stationary MDP problems.

In this case, we shall be surprised by the analysis difficulties that will appear and which will push the community to seek and build several stochastic process controls.

Part V

Appendices

Chapter 9

Appendix of chapter 4: proofs of Lemmas and Theorems

In this chapter, we provide the complete mathematical analysis of the main results stated in chapter 4 as well as all the theoretical tools needed for the previous purpose.

Notation 4 (Useful short-hand notations)**.** *In the following, for some element $x \in [0,1]$, we denote by $\bar{x}$ its complementary such that: $\bar{x} = 1 - x$. Then, we denote by $\Sigma_{s:t}$, and $\bar{\Sigma}_{s:t}$ the two following cumulative sums:*

$$\Sigma_{s:t} = \sum_{i=s}^{t} x_i \text{ and } \bar{\Sigma}_{s:t} = \sum_{i=s}^{t} \bar{x}_i.$$

9.1 Proof of Lemma 4.3

First, notice that the cumulative loss $\widehat{L}_{s:t}$ can be written as follows:

$$\widehat{L}_{s,t} = -\log \prod_{s'=s}^{t} \mathrm{Lp}\left(x_s | \mathbf{x}_{s':s-1}\right)$$

Then, in order to demonstrate the result of Lemma 4.3, we only need to show by induction that:

$$\forall \mathbf{x}_{1:n} \in \{0,1\}^n \quad \prod_{s=1}^{n} \mathrm{Lp}\left(x_s | \mathbf{x}_{1:s-1}\right) = \frac{1}{(n+1)\binom{n}{\sum_{i=1}^{n} x_i}}.$$

Step 1: For $n = 1$, we have to deal with two cases, $x_1 = 1$ and $x_1 = 0$. Using the definition of the predictor $\mathrm{Lp}\left(\cdot|\cdot\right)$, we obtain:

$$\begin{cases} \mathrm{Lp}\left(1|\emptyset\right) = 1/2 = \frac{1}{(1+1)\binom{1}{1}}, \\ \mathrm{Lp}\left(0|\emptyset\right) = 1/2 = \frac{1}{(1+1)\binom{1}{0}}. \end{cases}$$

Step 2: Assume that for some $\mathbf{x}_{1:n} \in \{0,1\}^n$, we have:

$$\prod_{s=1}^{n} \mathrm{Lp}\left(x_s | \mathbf{x}_{1:s-1}\right) = \frac{1}{(n+1)\binom{n}{\sum_{i=1}^{n} x_i}}. \tag{9.1.1}$$

Then, let us verify that:

$$\forall \mathbf{x}_{n+1} \in \{0,1\} \quad \prod_{s=1}^{n+1} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) = \frac{1}{(n+2)\binom{n+1}{\sum_{i=1}^{n+1} x_i}}.$$

To this end, we need to deal with two cases, depending on the values taken by x_{n+1}.

Case 1: $x_{n+1} = 1$ Observe that:

$$\prod_{s=1}^{n+1} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) = \prod_{s=1}^{n} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) \mathrm{Lp}\left(1|\mathbf{x}_{1:n}\right).$$

Using the definition of the predictor and the assumption (9.1.1), we obtain:

$$\begin{aligned}
\prod_{s=1}^{n+1} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) &= \frac{1}{(n+1)\binom{n}{\sum_{i=1}^{n} x_i}} \times \frac{\sum_{i=1}^{n} x_i + 1}{n+2} \\
&\overset{(a)}{=} \frac{\left(\sum_{i=1}^{n} x_i + 1\right) \times \left(\sum_{i=1}^{n} x_i\right)! \times \left(\sum_{i=1}^{n} \bar{x}_i\right)!}{(n+2)\,(n+1)\,n!} \\
&= \frac{\left(\sum_{i=1}^{n} x_i + 1\right)! \times \left(\sum_{i=1}^{n} \bar{x}_i + 0\right)!}{(n+2)\,(n+1)!} \\
&= \frac{\left(\sum_{i=1}^{n+1} x_i\right)! \times \left(\sum_{i=1}^{n+1} \bar{x}_i\right)!}{(n+2)\,(n+1)!} \\
&= \frac{1}{(n+2)\binom{n+1}{\sum_{i=1}^{n+1} x_i}}.
\end{aligned}$$

where (a) holds using the definition of the Binomial operator.

Case 2: $x_{n+1} = 0$ Observe that:

$$\prod_{s=1}^{n+1} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) = \prod_{s=1}^{n} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) \mathrm{Lp}\left(0|\mathbf{x}_{1:n}\right).$$

Using the definition of the predictor and the assumption (9.1.1), we obtain:

$$\begin{aligned}
\prod_{s=1}^{n+1} \mathrm{Lp}\left(x_s|\mathbf{x}_{1:s-1}\right) &= \frac{1}{(n+1)\binom{n}{\sum_{i=1}^{n} x_i}} \times \frac{\sum_{i=1}^{n} \bar{x}_i + 1}{n+2} \\
&\overset{(b)}{=} \frac{\left(\sum_{i=1}^{n} \bar{x}_i + 1\right) \times \left(\sum_{i=1}^{n} x_i\right)! \times \left(\sum_{i=1}^{n} \bar{x}_i\right)!}{(n+2)\,(n+1)\,n!} \\
&= \frac{\left(\sum_{i=1}^{n} x_i + 0\right)! \times \left(\sum_{i=1}^{n} \bar{x}_i + 1\right)!}{(n+2)\,(n+1)!} \\
&= \frac{\left(\sum_{i=1}^{n+1} x_i\right)! \times \left(\sum_{i=1}^{n+1} \bar{x}_i\right)!}{(n+2)\,(n+1)!} \\
&= \frac{1}{(n+2)\binom{n+1}{\sum_{i=1}^{n+1} x_i}}.
\end{aligned}$$

where (b) holds using the definition of the Binomial operator.

9.2 Proof of Lemma 4.4

The proof follows three main steps:

Step 1: Controlling the binomial $\binom{n}{k}$ Using the the Stirling formula:

$$\forall\, n \geqslant 1 \quad \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \leqslant n! \leqslant \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \exp\left(\frac{1}{12}\right),$$

the control of the binomial $\binom{n}{k}$ takes the following form:

$$\forall\, n \geqslant 1, \forall k \in [0,n] \quad \frac{n^n}{k^k\,(n-k)^{n-k}}\frac{\exp(b_1)}{\sqrt{n}} \leqslant \binom{n}{k} \leqslant \frac{n^n}{k^k\,(n-k)^{n-k}} \tag{9.2.1}$$

with $b_1 = -\frac{1}{6} - \frac{1}{2}\log(2\pi)$.

Step 2: First bounds for the cumulative loss $\widehat{L}_{s:t}$ Following Lemma 4.3, we can rewrite the cumulative loss $\widehat{L}_{s:t}$ as follows:

$$\widehat{L}_{s:t} = \log(n_{s:t}+1) + \log\binom{n_{s:t}}{\Sigma_{s:t}}.$$

Then by letting $\Phi(x) = x\log x$ and by following Equation (9.2.1), we obtain the following two bounds:

$$\begin{cases} \widehat{L}_{s:t} & \leqslant \log(n_{s:t}+1) + \Phi(n_{s:t}) - \Phi(\Sigma_{s:t}) - \Phi(\bar{\Sigma}_{s:t}), \\ \widehat{L}_{s:t} & \geqslant \log(n_{s:t}+1) + \Phi(n_{s:t}) - \Phi(\Sigma_{s:t}) - \Phi(\bar{\Sigma}_{s:t}) - \frac{9}{8} - \frac{1}{2}\log n_{s:t}. \end{cases} \tag{9.2.2}$$

Step 3: Controlling the cumulative loss First, notice that:

$$\Sigma_{s:t}\log\Sigma_{s:t} + \bar{\Sigma}_{s:t}\log\bar{\Sigma}_{s:t} = \Sigma_{s:t}\log\theta + \bar{\Sigma}_{s:t}\log\bar{\theta} + n_{s:t}\log n_{s:t} + n_{s:t}\mathbf{kl}\left(\frac{\Sigma_{s:t}}{n_{s:t}}, \theta\right). \tag{9.2.3}$$

Then, using Equations (9.2.2) with Equation (9.2.3), we obtain:

- **for the upper bound of the loss $\widehat{L}_{s:t}$**

$$\begin{aligned} \widehat{L}_{s:t} &\leqslant \log(n_{s:t}+1) - \Sigma_{s:t}\log\frac{\Sigma_{s:t}}{n_{s:t}} - \bar{\Sigma}_{s:t}\log\frac{\bar{\Sigma}_{s:t}}{n_{s:t}} \\ &\leqslant \log(n_{s:t}+1) - \Sigma_{s:t}\log\Sigma_{s:t} - \bar{\Sigma}_{s:t}\log\bar{\Sigma}_{s:t} + n_{s:t}\log n_{s:t} \\ &\overset{(a)}{\leqslant} \log(n_{s:t}+1) - \Sigma_{s:t}\log\theta - \bar{\Sigma}_{s:t}\log\bar{\theta} - n_{s:t}\mathbf{kl}\left(\frac{\Sigma_{s:t}}{n_{s:t}}, \theta\right) \\ &\overset{(b)}{\leqslant} \log(n_{s:t}+1) - \Sigma_{s:t}\log\theta - \bar{\Sigma}_{s:t}\log\bar{\theta}, \end{aligned}$$

  where (a) holds by using Equation (9.2.3) and (b) holds using the positiveness of the Kullback Leibler divergence ($\mathbf{kl}(\bullet,\bullet) \geqslant 0$),

- **for the lower bound of the loss** $\widehat{L}_{s:t}$

$$\begin{aligned}\widehat{L}_{s:t} &\geqslant \log(n_{s:t}+1) - \frac{1}{2}\log n_{s:t} - \Sigma_{s:t}\log\theta - \bar{\Sigma}_{s:t}\log\bar{\theta} - n_{s:t}\mathbf{kl}\left(\frac{\Sigma_{s:t}}{n_{s:t}},\theta\right) + b_1. \\ &\geqslant \log(n_{s:t}+1) - \frac{1}{2}\log n_{s:t} - \Sigma_{s:t}\log\theta - \bar{\Sigma}_{s:t}\log\bar{\theta} - n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t},\theta\right) - \frac{9}{8}.\end{aligned}$$

Lemma 9.9: Uniform confidence intervals
Let $Y_1, \dots Y_t$ be a sequence of t i.i.d. real-valued random variables with mean μ, such that $Y_t - \mu$ is σ-sub-Gaussian. Let $\widehat{\mu}_t = \frac{1}{t}\sum_{s=1}^{t} Y_s$ be the empirical mean estimate. Then, for all $\delta \in (0,1)$, it holds

$$\mathbb{P}\left(\exists t \in \mathbb{N}, \quad |\widehat{\mu}_t - \mu| \geqslant \sigma\sqrt{\left(1+\frac{1}{t}\right)\frac{2\ln(\sqrt{t+1}/\delta)}{t}}\right) \leqslant \delta$$

(The "Laplace" method refers to using the Laplace method of integration for optimization)

9.3 Proof of Lemma 9.9

We introduce for a fixed $\delta \in [0,1]$ the random variable

$$\tau = \min\left\{t \in \mathbb{N} : \mu_t - \mu \geqslant \sigma\sqrt{\left(1+\frac{1}{t}\right)\frac{2\ln(\sqrt{1+t}/\delta)}{t}}\right\}$$

This quantity is a random stopping time for the filtration $\mathcal{F} = (\mathcal{F}_t)_t$, where $\mathcal{F}_t = \sigma(Y_1, \dots, Y_t)$, since $\{\tau \leqslant m\}$ is $\mathcal{F}_m$ -measurable for all m. We want to show that $\mathbb{P}(\tau < \infty) \leqslant \delta$. To this end, for any λ, and t, we introduce the following quantity

$$M_t^\lambda = \exp\left(\sum_{s=1}^{t}\left(\lambda(Y_s - \mu) - \frac{\lambda^2\sigma^2}{2}\right)\right)$$

By assumption, the centered random variables are σ -sub-Gaussian and it is immediate to show that $\{M_t^\lambda\}_{t\in\mathbb{N}}$ is a non-negative super-martingale that satisfies $\ln\mathbb{E}\left[M_t^\lambda\right] \leqslant 0$ for all t. It then follows that $M_\infty^\lambda = \lim_{t\to\infty} M_t^\lambda$ is almost surely well-defined and so, M_τ^λ as well. Further, using the face that M_t^λ and $\{\tau > t\}$ are $\mathcal{F}_t$ measurable, it comes

$$\begin{aligned}\mathbb{E}\left[M_\tau^\lambda\right] &= \mathbb{E}\left[M_1^\lambda\right] + \mathbb{E}\left[\sum_{t=1}^{\tau-1} M_{t+1}^\lambda - M_t^\lambda\right] \\ &= 1 + \sum_{t=1}^{\infty}\mathbb{E}\left[\left(M_{t+1}^\lambda - M_t^\lambda\right)\mathbb{I}\{\tau > t\}\right] \\ &= 1 + \sum_{t=1}^{\infty}\mathbb{E}\left[\left(\mathbb{E}\left[M_{t+1}^\lambda \mid \mathcal{F}_t\right] - M_t^\lambda\right)\mathbb{I}\{\tau > t\}\right] \\ &\leqslant 1\end{aligned}$$

The next step is to introduce the auxiliary variable $\Lambda = \mathcal{N}\left(0, \sigma^{-2}\right)$, independent of all other variables, and study the quantity $M_t = \mathbb{E}\left[M_t^{\wedge} \mid \mathcal{F}_\infty\right]$. Note that the standard deviation of Λ is σ^{-1} due to the fact we consider σ -sub-Gaussian random variables. We immediately get $\mathbb{E}\left[M_\tau\right] = \mathbb{E}\left[\mathbb{E}\left[M_\tau^{\wedge} \mid \Lambda\right]\right] \leqslant 1$. For convenience, let $S_t = t\left(\mu_t - \mu\right)$. By construction of M_t, we have

$$\begin{aligned} M_t &= \frac{1}{\sqrt{2\pi\sigma^{-2}}} \int_{\mathbb{R}} \exp\left(\lambda S_t - \frac{\lambda^2\sigma^2 t}{2} - \frac{\lambda^2\sigma^2}{2}\right) d\lambda \\ &= \frac{1}{\sqrt{2\pi\sigma^{-2}}} \int_{\mathbb{R}} \exp\left(-\left[\lambda\sigma\sqrt{\frac{t+1}{2}} - \frac{S_t}{\sigma\sqrt{2(t+1)}}\right]^2 + \frac{S_t^2}{2\sigma^2(t+1)}\right) d\lambda \\ &= \exp\left(\frac{S_t^2}{2\sigma^2(t+1)}\right) \frac{1}{\sqrt{2\pi\sigma^{-2}}} \int_{\mathbb{R}} \exp\left(-\lambda^2\sigma^2\frac{t+1}{2}\right) d\lambda \\ &= \exp\left(\frac{S_t^2}{2\sigma^2(t+1)}\right) \frac{\sqrt{2\pi\sigma^{-2}/(t+1)}}{\sqrt{2\pi\sigma^{-2}}} \end{aligned}$$

Thus, we deduce that

$$|S_t| = \sigma\sqrt{2(t+1)\ln\left(\sqrt{t+1}M_t\right)}$$

We conclude by applying a simple Markov inequality (see Theorem 9.16):

$$\mathbb{P}\left(\tau\left|\mu_\tau - \mu\right| \geqslant \sigma\sqrt{2(\tau+1)\ln(\sqrt{\tau+1}/\delta)}\right) = \mathbb{P}\left(M_\tau \geqslant 1/\delta\right) \leqslant \mathbb{E}\left[M_\tau\right]\delta$$

Theorem 9.16: Markov inequality
Let X be a non-negative random variable and let $a > 0$, then:

$$\mathbb{P}\Big(X \geqslant a\Big) \leqslant \frac{\mathbb{E}[X]}{a}$$

9.4 Proof of Lemma 4.7

Step 1 Without a loss of generality, we consider that $r = 1$ and we consider that the sequence $(x_t)_t$ has σ-sub Gaussian noise meaning that:

$$\forall t, \forall\lambda \in \mathbb{R}, \quad \log\mathbb{E}\left[\exp\left(\lambda\left(x_t - \mathbb{E}\left[x_t\right]\right)\right)\right] \leqslant \frac{\lambda^2\sigma^2}{2} \tag{9.4.1}$$

Note that the Bernoulli case is a σ sub-Gaussianity case where $\sigma = \frac{1}{2}$. Indeed:

$$\forall\lambda \in \mathbb{R}, \quad \log\mathbb{E}_{X\sim B(p)}\exp(\lambda(X-p)) \leqslant \frac{\lambda^2}{8}$$

Let $\bar{z}_{s+1:t} = \widehat{\mu}_{s+1:t} - \widehat{\mu}_{s+1:t}$ be the centered empirical mean using observations from $s+1$ to t. We first introduce for each $\lambda \in \mathbb{R}$ and each $s \leqslant t$ the following quantity:

$$B^{\lambda}_{s,t} = \exp\left(\lambda(t-s)\bar{z}_{s+1:t} - \frac{\lambda^2\sigma^2(t-s)}{2}\right)$$

Note that $\left(B^{\lambda}_{s,t}\right)_{t\in[s,\infty]\cap\mathbb{N}}$ is a non-negative supermartingale. Let us introduce $B_{s,t} = \mathbb{E}\left[B^{\Lambda}_{s,t}\right]$, where $\Lambda \sim \mathcal{N}\left(0, \frac{1}{\sigma^2(t-s)c}\right)$, for some $c > 0$. We note that by simple algebra,

$$|\bar{z}_{s+1:t}| = \sqrt{\frac{2\sigma^2(1+c)}{t-s}\ln\left(B_{s,t}\sqrt{1+1/c}\right)}$$

In particular, choosing $c = 1/(t-s)$, it comes for all deterministic $g(t) > 0$, that

$$\begin{aligned}\mathbb{P}\left(\exists t, \exists s < t, |\bar{z}_{s+1:t}| \geqslant \sqrt{\frac{2\sigma^2\left(\frac{t-s+1}{t-s}\right)}{t-s}\ln\left(\frac{g(t)\sqrt{1+t-s}}{\delta}\right)}\right) &= \mathbb{P}\Big(\exists t, \exists s < t, B_{s,t} \geqslant \frac{g(t)}{\delta}\Big)\\ &\leqslant \mathbb{P}\left(\exists t, \max_{s<t} B_{s,t} \geqslant \frac{g(t)}{\delta}\right)\\ &\leqslant \delta\mathbb{E}\left[\max_t \frac{\max_{s<t} B_{s,t}}{g(t)}\right]\end{aligned}$$

Step 2 This leads to study the quantity $\frac{\max_{s<t} B_{s,t}}{g(t)}$. To this end, it is convenient to introduce $\bar{B}_t = \frac{\sum_{s<t} B_{s,t}}{g(t)}$ for $t > 1$. Indeed, for every random stopping time $\tau > 1$,

$$\mathbb{E}\left[\frac{\max_{s<\tau} B_{s,\tau}}{g(\tau)}\right] \leqslant \mathbb{E}\left[\bar{B}_\tau\right] = \mathbb{E}\left[\bar{B}_2 + \sum_{t=2}^{\infty}\left(\bar{B}_{t+1} - \bar{B}_t\right)\mathbb{I}\{\tau > t\}\right]$$

Further, we note that, conveniently

$$\bar{B}_{t+1} - \bar{B}_t = \frac{B_{t,t+1}}{g(t+1)} + \sum_{s<t}\left(\frac{B_{s,t+1}}{g(t+1)} - \frac{B_{s,t}}{g(t)}\right)$$

Next, by construction, we note that

$$\mathbb{E}\left[B_{s,t+1} \mid \mathcal{F}_t\right] = \frac{\sigma}{\sqrt{2\pi}}\int_{\mathbb{R}} \mathbb{E}\left[B^{\lambda}_{s,t+1} \mid \mathcal{F}_t\right] e^{-\frac{\lambda^2\alpha^2}{2}} d\lambda \leqslant \frac{\sigma}{\sqrt{2\pi}}\int_{\mathbb{R}} B^{\lambda}_{s,t} e^{-\frac{\lambda^2 q^2}{2}} d\lambda = B_{s,t}$$

Thus, since $\mathbb{I}\{\tau > t\} \in \mathcal{F}_t$, we deduce that

$$\begin{aligned}\mathbb{E}\left[\frac{\max_{s<\tau} B_{s,\tau}}{g(\tau)}\right] &\leqslant \mathbb{E}\left[\bar{B}_2\right] + \sum_{t=2}^{\infty}\frac{\mathbb{E}\left[B_{t,t+1}\right]}{g(t+1)} + \sum_{t=1}^{\infty}\sum_{s<t}\mathbb{E}\left[\left(\frac{1}{g(t+1)} - \frac{1}{g(t)}\right)B_{s,t}\mathbb{I}\{\tau > t\}\right]\\ &= \mathbb{E}\left[\bar{B}_2\right] + \sum_{t=2}^{\infty}\frac{\mathbb{E}\left[B_{t,t+1}\right]}{g(t+1)} + \sum_{t=1}^{\infty}\sum_{s<t}\left(\frac{1}{g(t+1)} - \frac{1}{g(t)}\right)\underbrace{\mathbb{E}\left[B_{s,t}\mathbb{I}\{\tau > t\}\right]}_{\geqslant 0}\end{aligned}$$

Hence, choosing g as an increasing function of t ensures that the last sum is upper bounded by 0. since on the other hand $\mathbb{E}[B_{t,t+1}] \leqslant 1$ and $\mathbb{E}[\bar{B}_2] \leqslant 1/g(2)$, we deduce that

$$\mathbb{E}\left[\frac{\max_{s<\tau} B_{s,\tau}}{g(\tau)}\right] \leqslant \frac{1}{g(2)} + \sum_{t=2}^{\infty} \frac{1}{g(t+1)} = \sum_{t=2}^{\infty} \frac{1}{g(t)}$$

Choosing $g(t) = Ct\ln^2(t)$ for $t > 1$ yields

$$\mathbb{E}\left[\frac{\max_{s<\tau} B_{s,\tau}}{g(\tau)}\right] \leqslant \frac{1}{C\ln(2)}$$

Plugging-in this in the control of the deviation and choosing $C = 1/\ln(2)$ thus gives

$$P\left(\exists t, \exists s < t \quad |\bar{z}_{s+1:t}| \geqslant \sqrt{\frac{2\sigma^2\left(1+\frac{1}{t-s}\right)}{t-s}\ln\left(\frac{t\ln^2(t)\sqrt{t+1-s}}{\ln(2)\delta}\right)}\right) \leqslant \delta$$

since on the other hand, by the classical Laplace method (see Lemma 9.9),

$$\mathbb{P}\left(\exists s, \quad |\bar{z}_{1:s}| \geqslant \sqrt{\frac{2\sigma^2\left(1+\frac{1}{s}\right)}{s}\ln\left(\frac{\sqrt{s+1}}{\delta}\right)}\right) \leqslant \delta$$

we conclude by using the triangular inequality $|\bar{z}_{1:s} - \bar{z}_{s+1:t}| \leqslant |\bar{z}_{1:s}| + |\bar{z}_{s+1:t}|$ together with a union bound argument.

9.5 Proof of Theorem 4.10

Assume that: $\forall t \in [r, \tau_c)$ $x_{r:t} \sim \mathcal{B}(\theta)^{\otimes n_{r:\tau_c}}$. The proof follows four main steps:

Step 1: Rewriting Lemma 4.5 and Lemma 4.6

- Let: $\widehat{\mu}_t$ denotes the empirical mean over the sequence $x_1, ..., x_t \sim \mathcal{B}(\theta)^{\otimes n_{1:t}}$, then:

$$\forall \delta \in (0,1), \forall \alpha > 1 \quad \mathbb{P}_\theta\left(\underbrace{\forall t \in \mathbb{N}^\star : \mathbf{kl}(\widehat{\mu}_t, \theta) < \frac{\alpha}{t}\log\frac{\log(\alpha t)\log(t)}{\log^2(\alpha)\delta}}_{E^{(1)}_{\theta,\delta,\alpha}}\right) \geqslant 1-\delta \tag{9.5.1}$$

- Let: $\widehat{\mu}_{s:t}$ denotes the empirical mean over the sequence $x_s, ..., x_t \sim \mathcal{B}(\theta)^{\otimes n_{s:t}}$, and:

$$E^{(2)}_{\theta,\delta,\alpha} = \left\{\forall t \in \mathbb{N}^\star, \forall s \in (r,t] : \mathbf{kl}(\widehat{\mu}_{s:t}, \theta) < \frac{\alpha}{n_{s:t}}\log\frac{n_{r:t}\log^2(n_{r:t})\log(\alpha n_{s:t})\log(n_{s:t})}{\log(2)\log^2(\alpha)\delta}\right\}$$

then, we get the following control:

$$\forall \delta \in (0,1), \forall \alpha > 1 \quad \mathbb{P}_\theta\left(E^{(2)}_{\theta,\delta,\alpha}\right) \geqslant 1-\delta \tag{9.5.2}$$

Then, let us build a suitable value of $\eta_{r,s,t}$ in order to ensure the control of the false alarm on the period $[r, \tau_c)$.
To this end, let us control the event: $\{\exists t > r, \texttt{Restart}_{r:t} = 1\}$ which is equivalent to the event $\{\exists t > r,\ s \in (r,t] : \vartheta_{r,s,t} \geqslant \vartheta_{r,r,t}\}$.

Step 2: Equivalent events. First, notice that:

$$\begin{aligned}
&\{\exists t > r,\ s \in (r,t] : \vartheta_{r,s,t} \geqslant \vartheta_{r,r,t}\} \Leftrightarrow \{\exists t > r,\ s \in (r,t] : \log \vartheta_{r,s,t} \geqslant \log \vartheta_{r,r,t}\}.\\
&\overset{(a)}{\Leftrightarrow} \{\exists t > r,\ s \in (r,t] : -\log \eta_{r,s,t} \leqslant \widehat{L}_{r:t} - \widehat{L}_{s:t} - \widehat{L}_{r:s-1}\},
\end{aligned}$$

where (a) comes directly from Equation (4.5.4).

Step 3: Using the cumulative loss controls. Then, note that $\forall \delta \in (0,1), \forall \alpha > 1$ we get:

$$\begin{aligned}
&\mathbb{P}_\theta\Big(\exists t > r,\ s \in (r,t] : \vartheta_{r,s,t} \geqslant \vartheta_{r,r,t}\Big) = \mathbb{P}_\theta\Big(\exists t > r,\ s \in (r,t] : \log \vartheta_{r,s,t} \geqslant \log \vartheta_{r,r,t}\Big)\\
&= \mathbb{P}_\theta\Big(\exists t > r,\ s \in (r,t] : -\log \eta_{r,s,t} \leqslant \widehat{L}_{r:t} - \widehat{L}_{r:s-1} - \widehat{L}_{s:t}\Big)\\
&\overset{(b)}{\leqslant} \mathbb{P}_\theta\Bigg(\exists t > r,\ s \in (r,t] : -\log \eta_{r,s,t} \leqslant \log \frac{\sqrt{n_{r:s-1} \times n_{s:t}} \times (n_{r:t}+1)}{(n_{r:s-1}+1) \times (n_{s:t}+1)}\\
&\qquad\qquad + n_{r:s-1}\mathbf{kl}\left(\widehat{\mu}_{r:s-1}, \theta\right) + n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t}, \theta\right) + \frac{9}{4}\Bigg)\\
&\overset{(c)}{\leqslant} \mathbb{P}_\theta\Bigg(\exists t > r,\ s \in (r,t] : -\log \eta_{r,s,t} \leqslant \log \frac{n_{r:t}+1}{\sqrt{n_{r:s-1} \times n_{s:t}}}\\
&\qquad\qquad + n_{r:s-1}\mathbf{kl}\left(\widehat{\mu}_{r:s-1}, \theta\right) + n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t}, \theta\right) + \frac{9}{4}\Bigg)\\
&\leqslant \frac{\delta}{2} + \mathbb{P}_\theta\Bigg(\exists t > r,\ s \in (r,t] : -\log \eta_{r,s,t} \leqslant \log \frac{n_{r:t}+1}{\sqrt{n_{r:s-1} \times n_{s:t}}}\\
&\qquad\qquad + n_{r:s-1}\mathbf{kl}\left(\widehat{\mu}_{r:s-1}, \theta\right) + n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t}, \theta\right) + \frac{9}{4} \bigcap E^{(1)}_{\theta,\delta/2,\alpha}\Bigg)\\
&\overset{(d)}{\leqslant} \frac{\delta}{2} + \mathbb{P}_\theta\Bigg(\exists t > r, s \in (r,t] : \log \frac{1}{\eta_{r,s,t}} \leqslant \log \frac{n_{r:t}+1}{\sqrt{n_{r:s-1} \times n_{s:t}}}\\
&\qquad\qquad + \alpha \log \frac{2\log(\alpha n_{r:s-1})\log(n_{r:s-1})}{\log^2(\alpha)\delta} + n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t}, \theta\right) + \frac{9}{4}\Bigg)\\
&\leqslant \delta + \mathbb{P}_\theta\Bigg(\exists t > r, s \in (r,t] : \log \frac{1}{\eta_{r,s,t}} \leqslant \log \frac{n_{r:t}+1}{\sqrt{n_{r:s-1} n_{s:t}}}
\end{aligned}$$

$$+\alpha\log\frac{2\log(\alpha n_{r:s-1})\log n_{r:s-1}}{\log^2(\alpha)\delta}+n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t},\theta\right)+\frac{9}{4}\bigcap E^{(2)}_{\theta,\delta/2,\alpha}\Bigg)$$

$$\overset{(e)}{\leqslant}\delta+\mathbb{P}_\theta\Bigg(\exists t>r,\ s\in(r,t]:-\log\eta_{r,s,t}\leqslant\log\frac{n_{r:t}+1}{\sqrt{n_{r:s-1}\times n_{s:t}}}+\frac{9}{4}$$

$$+\alpha\log\frac{2\log(\alpha n_{r:s-1})\log(n_{r:s-1})}{\log^2(\alpha)\delta}+\alpha\log\frac{2n_{r:t}\log^2(n_{r:t})\log(\alpha n_{s:t})\log(n_{s:t})}{\log(2)\log^2(\alpha)\delta}\Bigg)$$

(b) holds by using Lemma 4.4, (c) holds thanks to $(n_{r:s-1}+1)\times(n_{s:t}+1)>n_{r:s-1}\times n_{s:t}$, (d) holds true thanks to Equation 9.5.1 and (e) holds true thanks to Equation 9.5.2.

Step 4: Building the sufficient condition on $\eta_{r,s,t}$ Thus, by using $\exp(-\frac{9}{4})>\frac{1}{10}$, we get the following condition on $\eta_{r,s,t}$:

$$\begin{aligned}\eta_{r,s,t}&<\frac{\sqrt{n_{r:s-1}\times n_{s:t}}}{10\left(n_{r:t}+1\right)}\times\left(\frac{\log^2(\alpha)\delta}{2\log(\alpha n_{r:s-1})\log(n_{r:s-1})}\times\frac{\log(2)\log^2(\alpha)\delta}{2n_{r:t}\log^2(n_{r:t})\log(\alpha n_{s:t})\log(n_{s:t})}\right)^\alpha\\&=\frac{\sqrt{n_{r:s-1}\times n_{s:t}}}{10\left(n_{r:t}+1\right)}\times\left(\frac{\log(4\alpha)\log(2)\delta^2}{4n_{r:t}\log(\alpha n_{r:t})\log^2(n_{r:t})\log(n_{r:t})}\right)^\alpha\\&=\frac{\sqrt{n_{r:s-1}\times n_{s:t}}}{10\left(n_{r:t}+1\right)}\times\left(\frac{\log(4\alpha+2)\delta^2}{4n_{r:t}\log((\alpha+3)\,n_{r:t})}\right)^\alpha,\end{aligned}$$

which allows us to get the following control:

$$\boxed{\mathbb{P}_\theta\left(\exists t>r,s\in(r,t]:\vartheta_{r,s,t}\geqslant\vartheta_{r,r,t}\right)\leqslant\delta.}$$

9.6 Proof of Theorem 4.11

The proof follows three main steps.

Step 1: some preliminaries Before building the detection delay, we need to introduce three intermediate results.

The first result is to link the quantity $\Phi\left(\Sigma_{s:t}\right)$ to $\Phi\left(\widehat{\mu}_{s:t}\right)$ such that:

$$\forall\left(s,t\right):\quad\Phi\left(\Sigma_{s:t}\right)+\Phi\left(\bar{\Sigma}_{s:t}\right)-\Phi\left(n_{s:t}\right)=n_{s:t}\left(\Phi\left(\widehat{\mu}_{s:t}\right)+\Phi\left(1-\widehat{\mu}_{s:t}\right)\right).$$

Then, observe that :

$$\begin{aligned}&n_{r:s-1}\left(\Phi\left(\widehat{\mu}_{r:s-1}\right)+\Phi\left(1-\widehat{\mu}_{r:s-1}\right)\right)+n_{s:t}\left(\Phi\left(\widehat{\mu}_{s:t}\right)+\Phi\left(1-\widehat{\mu}_{s:t}\right)\right)\\&-n_{r:t}\left(\Phi\left(\widehat{\mu}_{r:t}\right)+\Phi\left(1-\widehat{\mu}_{r:t}\right)\right)=n_{r:s-1}\mathbf{kl}\left(\widehat{\mu}_{r:s-1},\widehat{\mu}_{r:t}\right)+n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t},\widehat{\mu}_{r:t}\right).\end{aligned}\tag{9.6.1}$$

Finally, observe that:

$$n_{r:s-1}\left(\widehat{\mu}_{r:s-1}-\widehat{\mu}_{r:t}\right)^2+n_{s:t}\left(\widehat{\mu}_{s:t}-\widehat{\mu}_{r:t}\right)^2=\frac{n_{r:s-1}n_{s:t}}{n_{r:t}}\left(\widehat{\mu}_{r:s-1}-\widehat{\mu}_{s:t}\right)^2.\tag{9.6.2}$$

Then, we will also need a useful notation as $f_{r,s,t}$ (which comes directly from Lemma 4.4):

$$f_{r,s,t} = \log\left(n_{r:s-1}+1\right) + \log\left(n_{s:t}+1\right) - \frac{1}{2}\log n_{r:t} + \frac{9}{8}.$$

Finally, following Lemma 4.7, the control of the quantity $|\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}|$ takes the following form: (with a probability at least $1-\delta$)

$$\forall\, s \in [r:t) \quad |\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}| \geqslant \Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}, \tag{9.6.3}$$

where $\Delta_{r,s,t}$ represents the relative gap and it takes the following form:

$$\Delta_{r,s,t} = |\,\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}| = \begin{cases} \frac{n_{\tau_c:t}}{n_{s:t}}\,|\theta_1-\theta_2| = \frac{n_{\tau_c:t}}{n_{s:t}}\Delta & \text{if } s < \tau_c \leqslant t, \\ \frac{n_{r:\tau_c-1}}{n_{r:s-1}}\,|\theta_1-\theta_2| = \frac{n_{r:\tau_c-1}}{n_{r:s-1}}\Delta & \text{if } \tau_c \leqslant s \leqslant t. \end{cases} \tag{9.6.4}$$

Step 2: Building the sufficient conditions for detecting the change-point τ_c First, assume that: $x_{r:\tau_c-1} \sim \mathcal{B}(\theta_1)$, $x_{\tau_c:t} \sim \mathcal{B}(\theta_2)$. Then, to build the detection delay, we need to prove that at some instant after τ_c the restart criterion $\texttt{Restart}_{r:t}$ is activated. In other words, we need to build the following guarantee:

$$\mathbb{P}\Big(\exists t > \tau_c : \texttt{Restart}_{r:t} = 1\Big) > 1-\delta.$$

Notice that:

$$\begin{aligned}
&\{\forall\, t > \tau_c : \texttt{Restart}_{r:t} = 0\} \Leftrightarrow \{\forall\, t > \tau_c, \forall s \in (r,t] : \log \vartheta_{r,s,t} \leqslant \log \vartheta_{r,r,t}\}. \\
&\Leftrightarrow \left\{\forall\, t > \tau_c, \forall s \in (r,t] : \log \eta_{r,s,t} \leqslant \widehat{L}_{r:s-1} + \widehat{L}_{s:t} - \widehat{L}_{r:t}\right\}. \\
&\overset{(a)}{\Rightarrow} \{\forall\, t > \tau_c, \forall s \in (r,t] : \log \eta_{r,s,t} \leqslant f_{r,s,t} + \Phi\left(\Sigma_{r:s-1}\right) + \Phi\left(\bar{\Sigma}_{r:s-1}\right) - \Phi\left(n_{r:s-1}\right). \\
&\qquad\qquad + \Phi\left(\Sigma_{s:t}\right) + \Phi\left(\bar{\Sigma}_{s:t}\right) - \Phi\left(n_{s:t}\right) - \Phi\left(\Sigma_{r:t}\right) - \Phi\left(\bar{\Sigma}_{r:t}\right) + \Phi\left(n_{r:t}\right)\}. \\
&\overset{(b)}{\Rightarrow} \{\forall\, t > \tau_c, \forall s \in (r,t] : \log \eta_{r,s,t} \leqslant f_{r,s,t} - n_{r:s-1}\mathbf{kl}\left(\widehat{\mu}_{r:s-1}, \widehat{\mu}_{r:t}\right) - n_{s:t}\mathbf{kl}\left(\widehat{\mu}_{s:t}, \widehat{\mu}_{r:t}\right)\}. \\
&\overset{(c)}{\Rightarrow} \left\{\forall\, t > \tau_c, \forall s \in (r,t] : \log \eta_{r,s,t} \leqslant f_{r,s,t} - 2n_{r:s-1}\left(\widehat{\mu}_{r:s-1} - \widehat{\mu}_{r:t}\right)^2 - 2n_{s:t}\left(\widehat{\mu}_{s:t} - \widehat{\mu}_{r:t}\right)^2\right\}. \\
&\overset{(d)}{\Rightarrow} \left\{\forall\, t > \tau_c, \forall s \in (r,t] : \log \eta_{r,s,t} \leqslant f_{r,s,t} - 2 \times \frac{n_{r:s-1}n_{s:t}}{n_{r:t}}\left(\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}\right)^2\right\}. \\
&\Rightarrow \left\{\forall\, t > \tau_c, \forall s \in (r,t] : 2 \times \frac{n_{r:s-1}n_{s:t}}{n_{r:t}}\left(\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}\right)^2 \leqslant f_{r,s,t} - \log \eta_{r,s,t}\right\}. \\
&\overset{(e)}{\Rightarrow} \left\{\forall\, t > \tau_c, \forall s \in (r,t] : \sqrt{\frac{n_{r:s-1}n_{s:t}}{n_{r:t}}}\,|\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}| \leqslant \frac{\sqrt{f_{r,s,t} - \log \eta_{r,s,t}}}{\sqrt{2}}\right\}.
\end{aligned}$$

where (a), holds true thanks to Equation (9.2.2), (b) holds true thanks to Equation (9.6.1), (c) holds true thanks to the Pinsker Inequality taking the following form: $\forall\,(\theta_1,\theta_2) \in [0,1]^2\, \mathbf{kl}(\theta_1,\theta_2) \geqslant 2(\theta_1-\theta_2)^2$. (d) holds true thanks to Equation (9.6.2) and (e) holds true under the condition that $\eta_{r,s,t} \leqslant \exp(f_{r,s,t})$.

Therefore, we obtain:

$$\begin{aligned}
\mathbb{P}\Big(\forall t > \tau_c : \texttt{Restart}_{r:t} = 0\Big) &\leqslant \mathbb{P}\Bigg(\forall t > \tau_c, \forall s \in (r,t] : \sqrt{\frac{n_{r:s-1}n_{s:t}}{n_{r:t}}}|\widehat{\mu}_{r:s-1} - \widehat{\mu}_{s:t}| \\
&\qquad\qquad \leqslant \frac{\sqrt{f_{r,s,t} - \log \eta_{r,s,t}}}{\sqrt{2}}\Bigg) \\
&\overset{(f)}{\leqslant} \delta + \mathbb{P}\Bigg(\forall\ t > \tau_c, \forall s \in (r,t] : \sqrt{\frac{n_{r:s-1}n_{s:t}}{n_{r:t}}}\left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right) \leqslant \frac{\sqrt{f_{r,s,t} - \log \eta_{r,s,t}}}{\sqrt{2}}\Bigg) \\
&= \delta + \mathbb{P}\Bigg(\forall\ t > \tau_c, \forall s \in (r,t] : \ \frac{n_{r:s-1}n_{s:t}}{n_{r:t}}\left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right)^2 \leqslant \frac{f_{r,s,t} - \log \eta_{r,s,t}}{2}\Bigg) \\
&= \delta + \mathbb{P}\Bigg(\forall\ t > \tau_c, \forall s \in (r,t] : \ \underbrace{1 - \frac{f_{r,s,t} - \log \eta_{r,s,t}}{2n_{r,s-1} \times \left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right)^2}}_{A} \leqslant \frac{n_{r:s-1}}{n_{r:t}}\Bigg), \qquad (9.6.5)
\end{aligned}$$

where (f) holds true thanks to Equation (9.6.3) (We recall that the relative gap $\Delta_{r,s,t}$ is defined in Equation (9.6.4)). Before continuing the analysis, one need to verify that term A is valid (i.e. $A \in [0,1]$, otherwise the associated event cannot be controlled). So, notice that:

$$\begin{cases} A > 0 & \Leftrightarrow \eta_{r,s,t} > \exp\left(-2n_{r,s-1}\left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right)^2\right)\exp\left(f_{r,s,t}\right), \\ A < 1 & \Leftrightarrow \eta_{r,s,t} < \exp\left(f_{r,s,t}\right) = \frac{(n_{r:s-1}+1)(n_{s:t}+1)}{\sqrt{n_{r:t}}}\exp\left(\frac{9}{8}\right). \end{cases} \qquad (9.6.6)$$

The second condition in Equation (9.6.6) is always satisfied since, we have:

$$\forall\,(r,s,t) : \frac{(n_{r:s-1}+1)\,(n_{s:t}+1)}{\sqrt{n_{r:t}}}\exp\left(\frac{9}{8}\right) > 1 \text{ and by definition, we have: } \eta_{r,s,t} < 1.$$

Therefore, from Equation (9.6.5) we get the following implication:

$$\begin{aligned}
&\left\{\exists t > \tau_c, s \in (r,t] : 1 + \frac{\log \eta_{r,s,t} - f_{r,s,t}}{2n_{r,s-1}\left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right)^2} > \frac{n_{r:s-1}}{n_{r:t}}\right\} \\
&\Rightarrow \mathbb{P}\Big(\exists t > \tau_c : \texttt{Restart}_{r:t} = 1\Big) > 1 - \delta.
\end{aligned}$$

In other words, the change-point τ_c is detected at time t (with probability at least $1-\delta$) if for some $s \in (r,t]$, we have:

$$1 + \frac{\log \eta_{r,s,t} - f_{r,s,t}}{2n_{r,s-1} \times \left(\Delta_{r,s,t} - \mathcal{C}_{r,s,t,\delta}\right)^2} > \frac{n_{r:s-1}}{n_{r:t}}. \qquad (9.6.7)$$

Step 3: Non-asymptotic expression of the detection delay $\mathfrak{D}_{\Delta,r,\tau_c}$ To build the detection delay, we need to ensure the existence of $s \in (r,t]$ such that Equation (9.6.7) is

satisfied. In particular, Equation (9.6.7) can be satisfied for $s = \tau_c$. By this way, a condition to detect the change-point τ_c is written as follows:

$$1 + \frac{\log \eta_{r,\tau_c,t} - f_{r,\tau_c,t}}{2n_{r,\tau_c-1} \times (\Delta - \mathcal{C}_{r,\tau_c,t,\delta})^2} > \frac{n_{r:\tau_c-1}}{n_{r:t}}. \tag{9.6.8}$$

To build the delay, we should introduce the following variable: $d = t - \tau_c + 1 = n_{\tau_c:t} \in \mathbb{N}^\star$.

Thus from Equation (9.6.8), we obtain:

$$\begin{aligned} & 1 + \frac{\log \eta_{r,\tau_c,d+\tau_c-1} - f_{r,\tau_c,d+\tau_c-1}}{2n_{r,\tau_c-1}(\Delta - \mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta})^2} > \frac{n_{r:\tau_c-1}}{n_{r:\tau_c-1} + d} \\ \Leftrightarrow\ & d > \frac{\left(1 - \frac{\mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta}}{\Delta}\right)^{-2}}{2\Delta^2} \times \frac{-\log \eta_{r,\tau_c,d+\tau_c-1} + f_{r,\tau_c,d+\tau_c-1}}{1 + \frac{\log \eta_{r,\tau_c,d+\tau_c-1} - f_{r,\tau_c,d+\tau_c-1}}{2n_{r,\tau_c-1}(\Delta - \mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta})^2}}. \end{aligned}$$

Finally, the change-point τ_c is detected (with a probability at least $1 - \delta$) with a delay not exceeding $\mathfrak{D}_{\Delta,r,\tau_c}$, such that:

$$\mathfrak{D}_{\Delta,r,\tau_c} = \min \left\{ d \in \mathbb{N}^\star : d > \frac{\left(1 - \frac{\mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta}}{\Delta}\right)^{-2}}{2\Delta^2} \times \frac{-\log \eta_{r,\tau_c,d+\tau_c-1} + f_{r,\tau_c,d+\tau_c-1}}{1 + \frac{\log \eta_{r,\tau_c,d+\tau_c-1} - f_{r,\tau_c,d+\tau_c-1}}{2n_{r,\tau_c-1}(\Delta - \mathcal{C}_{r,\tau_c,d+\tau_c-1,\delta})^2}} \right\}.$$

Bibliography

[1] Yasin Abbasi-Yadkori et al. "Best of both worlds: Stochastic adversarial best-arm identification". In: *Proceedings of the 31st Conference On Learning Theory*. Ed. by Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, June 2018, pp. 918–949. url: http://proceedings.mlr.press/v75/abbasi-yadkori18a.html.

[2] Ryan Prescott Adams and David JC MacKay. "Bayesian online changepoint detection". In: *arXiv preprint arXiv:0710.3742* (2007).

[3] Ferran Adelantado et al. "Understanding the limits of LoRaWAN". In: *IEEE Communications magazine* 55.9 (2017), pp. 34–40.

[4] Réda Alami, Odalric Maillard, and Raphael Féraud. "Memory Bandits: a Bayesian approach for the Switching Bandit Problem". In: *Neural Information Processing Systems: Bayesian Optimization Workshop*. 2016.

[5] Réda Alami, Odalric Maillard, and Raphael Féraud. "Restarted Bayesian Online Changepoint Detector achieves Optimal Detection Delay". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 211–221.

[6] Robin Allesiardo and Raphael Féraud. "EXP3 with drift detection for the switching bandit problem". In: *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*. IEEE. 2015, pp. 1–7.

[7] Robin Allesiardo, Raphael Féraud, and Odalric-Ambrym. Maillard. "The non-stationary stochastic multi-armed bandit problem". In: *International Journal of Data Science and Analytics* 3.4 (2017).

[8] LoRa Alliance. *What is LoRaWAN?* https://www.lora-alliance.org/what-is-lora. Last accessed August, 13th 2017.

[9] TL Alumona and N Nnamani Kelvin. "Path Loss Prediction of Wireless Mobile Communication for Urban Areas of Imo State, South-East Region of Nigeria at 910 MHz". In: *JAIR* (2015).

[10] Samaneh Aminikhanghahi and Diane J. Cook. "A survey of methods for time series change point detection". In: *Knowledge and Information Systems* 51.2 (May 2017), pp. 339–367. issn: 0219-3116. doi: 10.1007/s10115-016-0987-z. url: https://doi.org/10.1007/s10115-016-0987-z.

[11] Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. "Best arm identification in multi-armed bandits." In: *COLT*. 2010, pp. 41–53.

[12] Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. "Exploration–exploitation tradeoff using variance estimates in multi-armed bandits". In: *Theoretical Computer Science* 410.19 (2009), pp. 1876–1902.

[13] Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. "Variance estimates and exploration function in multi-armed bandit". In: *CERTIS Research Report 07–31*. Citeseer, 2007.

[14] Peter Auer, Nicolo Cesa-Bianchi, and Fischer Paul. "Finite-time analysis of the multiarmed bandit problem". In: *Machine Learning* 47.2 (2002).

[15] Peter Auer et al. "Gambling in a rigged casino: The adversarial multi-armed bandit problem". In: *Proceedings of IEEE 36th Annual Foundations of Computer Science*. IEEE. 1995, pp. 322–331.

[16] Peter Auer et al. "The nonstochastic multiarmed bandit problem". In: *SIAM journal on computing* 32.1 (2002), pp. 48–77.

[17] Orly Avner and Shie Mannor. "Concurrent bandits and cognitive radio networks". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2014, pp. 66–81.

[18] Konstantin Avrachenkov, Laura Cottatellucci, and Lorenzo Maggi. "Slow fading channel selection: A restless multi-armed bandit formulation". In: *Proc. of IEEE ISWCS*. 2012, pp. 1083–1087.

[19] Nouha Baccour et al. "Radio Link Quality Estimation in Wireless Sensor Networks: A Survey". In: *ACM Trans. Sen. Netw.* 8.4 (Sept. 2012), 34:1–34:33. issn: 1550-4859. doi: 10.1145/2240116.2240123.

[20] Michèle Basseville, Igor V Nikiforov, et al. *Detection of abrupt changes: theory and application*. Vol. 104. Prentice Hall Englewood Cliffs, 1993.

[21] Dirk Bergemann and Juuso Valimaki. "Bandit problems". In: (2006).

[22] Omar Besbes, Yonatan Gur, and Assaf Zeevi. "Stochastic Multi-armed-bandit Problem with Non-stationary Rewards". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems*. NIPS'14. Montreal, Canada: MIT Press, 2014, pp. 199–207. url: http://dl.acm.org/citation.cfm?id=2968826.2968849.

[23] Lilian Besson and Emilie Kaufmann. "Multi-player bandits revisited". In: *Algorithmic Learning Theory*. PMLR. 2018, pp. 56–92.

[24] Lilian Besson and Emilie Kaufmann. "The generalized likelihood ratio test meets klucb: an improved algorithm for piece-wise non-stationary bandits". In: *arXiv preprint arXiv:1902.01575* (2019).

[25] Rahul Biswas et al. "Towards object mapping in non-stationary environments with mobile robots". In: *IEEE/RSJ international conference on intelligent robots and systems*. Vol. 1. IEEE. 2002, pp. 1014–1019.

[26] Rémi Bonnefoi et al. "Multi-Armed Bandit Learning in IoT Networks: Learning helps even in non-stationary settings". In: *CROWNCOM*. Vol. 12. 2017.

[27] Emily Brodsky and Boris S Darkhovsky. *Nonparametric Methods in Change-Point Problems*. Vol. 243. Springer, 1993.

[28] Sébastien Bubeck and Nicolo Cesa-Bianchi. "Regret analysis of stochastic and nonstochastic multi-armed bandit problems". In: *arXiv preprint arXiv:1204.5721* (2012).

[29] Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. "Pure exploration in multi-armed bandits problems". In: *International conference on Algorithmic learning theory*. Springer. 2009, pp. 23–37.

[30] Yang Cao et al. "Nearly optimal adaptive procedure with change detection for piecewise-stationary bandit". In: *The 22nd International Conference on Artificial Intelligence and Statistics*. 2019, pp. 418–427.

[31] Olivier Cappé et al. "Kullback–Leibler upper confidence bounds for optimal sequential allocation". In: *The Annals of Statistics* 41.3 (2013), pp. 1516–1541.

[32] François Caron, Arnaud Doucet, and Raphael Gottardo. "On-line changepoint detection and parameter estimation with application to genomic data". In: *Statistics and Computing* 22.2 (2012), pp. 579–595.

[33] Andressa Cerqueira and Florencia Leonardi. "Strong consistency of Krichevsky-Trofimov estimator for the number of communities in the Stochastic Block Model". In: *arXiv preprint arXiv:1804.03509* (2018).

[34] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[35] Mithun Chakraborty et al. "Coordinated Versus Decentralized Exploration In Multi-Agent Multi-Armed Bandits." In: *IJCAI*. 2017, pp. 164–170.

[36] Olivier Chapelle and Lihong Li. "An Empirical Evaluation of Thompson Sampling." In: *NIPS*. 2011, pp. 2249–2257.

[37] Miklós Csörgö and Lajos Horváth. *Limit theorems in change-point analysis*. Vol. 18. John Wiley & Sons Inc, 1997.

[38] Fredrik A Dahl. "A reinforcement learning algorithm applied to simplified two-player Texas Hold'em poker". In: *European Conference on Machine Learning*. Springer. 2001, pp. 85–96.

[39] Hiba Dakdouk et al. "Reinforcement learning techniques for optimized channel hopping in IEEE 802.15. 4-TSCH networks". In: *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. 2018, pp. 99–107.

[40] *Digital mobile radio towards future generation systems*. Tech. rep. European Communities, EUR 18957, 1999.

[41] John C Duchi, Michael I Jordan, and Martin J Wainwright. "Privacy aware learning". In: *Journal of the ACM (JACM)* 61.6 (2014), p. 38.

[42] Cynthia Dwork et al. "Calibrating noise to sensitivity in private data analysis". In: *Journal of Privacy and Confidentiality* 7.3 (2016), pp. 17–51.

[43] Digi-Key Electronics. *LoRaWAN Part 1: How to Get 15 km Wireless and 10-Year Battery Life for IoT*. https://www.digikey.com/en/articles/techzone/2016/nov/lorawan-part-1-15-km-wireless-10-year-battery-life-iot. Last accessed August, 13th 2017.

[44] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. "Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems". In: *Journal of machine learning research* 7.Jun (2006), pp. 1079–1105.

[45] Paul Fearnhead and Peter Clifford. "On-line inference for hidden Markov models via particle filters". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65.4 (2003), pp. 887–899.

[46] Paul Fearnhead and Zhen Liu. "On-line inference for multiple changepoint problems". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69.4 (2007), pp. 589–605.

[47] Raphaël Féraud, Réda Alami, and Romain Laroche. "Decentralized exploration in multi-armed bandits". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 1901–1909.

[48] Raphael Feraud et al. "Random Forest for the Contextual Bandit Problem-extended version". In: *arXiv preprint arXiv:1504.06952* (2015).

[49] Lúcio Ferreira et al. "Characterisation of Signal Penetration into Buildings for GSM and UMTS". In: *Proc. of IEEE ISWCS*. 2006.

[50] Daniel Fink. "A compendium of conjugate priors". In: *See http://www. people. cornell. edu/pages/df36/CONJINTRnew% 20TEX. pdf* 46 (1997).

[51] Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. "Best arm identification: A unified approach to fixed budget and fixed confidence". In: *Advances in Neural Information Processing Systems*. 2012, pp. 3212–3220.

[52] Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. "Learning multiuser channel allocations in cognitive radio networks: A combinatorial multi-armed bandit formulation". In: *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum (DySPAN)*. IEEE. 2010, pp. 1–9.

[53] Pratik Gajane, Tanguy Urvoy, and Emilie Kaufmann. "Corrupt bandits for preserving local privacy". In: *arXiv preprint arXiv:1708.05033* (2017).

[54] MS Ganga et al. "Analysis of Fast Fading in Wireless Communication Channels". In: *Department of Electrical and Communication Engineering, JNTU, Hyderabad, India* (2010).

[55] Srivatsava Ranjit Ganta, Shiva Prasad Kasiviswanathan, and Adam Smith. "Composition attacks and auxiliary information in data privacy". In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2008, pp. 265–273.

[56] Ravi Ganti et al. "Thompson sampling for dynamic pricing". In: *arXiv preprint arXiv:1802.03050* (2018).

[57] Aurélien Garivier and Olivier Cappé. "The KL-UCB algorithm for bounded stochastic bandits and beyond". In: *Proceedings of the 24th annual Conference On Learning Theory*. 2011, pp. 359–376.

[58] Aurélien Garivier and Eric Moulines. "On upper-confidence bound policies for switching bandit problems". In: *International Conference on Algorithmic Learning Theory*. Springer. 2011, pp. 174–188.

[59] Sakshama Ghoslya. *How does LoRaWAN Adaptive Data Rate work?* http://www.sghoslya.com/p/how-does-lorawan-nodes-changes-their.html. Last accessed August, 13th 2017.

[60] Jean Charles Gittins and D. Marc Jones. "A Dynamic Allocation Index for the Discounted Multiarmed Bandit Problem". In: *Biometrika* 66.3 (1979).

[61] Dani Goldberg and Maja J Matarić. "Maximizing reward in a non-stationary mobile robot environment". In: *Autonomous Agents and Multi-Agent Systems* 6.3 (2003), pp. 287–316.

[62] Pedro Henrique Gomes, Thomas Watteyne, and Bhaskar Krishnamachari. "MABO-TSCH: Multihop and blacklist-based optimized time synchronized channel hopping". In: *Transactions on Emerging Telecommunications Technologies* (2017).

[63] Marco Grzegorczyk and Dirk Husmeier. "Non-stationary continuous dynamic Bayesian networks". In: *Advances in Neural Information Processing Systems*. 2009, pp. 682–690.

[64] Sudipto Guha, Kamesh Munagala, and Peng Shi. "Approximation algorithms for restless bandit problems". In: *Journal of the ACM (JACM)* 58.1 (2010), pp. 1–50.

[65] Mo Haghighi et al. "Game theoretic and auction-based algorithms towards opportunistic communications in LPWA LoRa networks". In: *Proc. of IEEE WF-IoT*. Dec. 2016, pp. 735–740. doi: 10.1109/WF-IoT.2016.7845517.

[66] Cédric Hartland et al. "Multi-armed bandit, dynamic environments and meta-bandits". In: (2006).

[67] M. Hata. "Empirical formula for propagation loss in land mobile radio services". In: *IEEE Transactions on Vehicular Technology* (1980).

[68] José Miguel Hernández-Lobato et al. "Parallel and distributed Thompson sampling for large-scale accelerated exploration of chemical space". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 1470–1479.

[69] Eshcar Hillel et al. "Distributed exploration in multi-armed bandits". In: *Advances in Neural Information Processing Systems*. 2013, pp. 854–862.

[70] David V Hinkley. "Inference about the change-point from cumulative sum tests". In: *Biometrika* 58.3 (1971), pp. 509–523.

[71] IEEE. "IEEE Standard for Low-Rate Wireless Networks". In: *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)* (2016), pp. 1–709. doi: 10.1109/IEEESTD.2016.7460875.

[72] ISA-100.11a-2011: "Wireless Systems for Industrial Automation:Process Control and Related Applications". In: *International Society of Automation (ISA) Std.* 1 (May 2011).

[73] Thomas Jaksch, Ronald Ortner, and Peter Auer. "Near-optimal Regret Bounds for Reinforcement Learning." In: *Journal of Machine Learning Research* 11.4 (2010).

[74] Daniel R Jeske and Ashwin Sampath. "Signal-to-interference-plus-noise ratio estimation for wireless communication systems: Methods and analysis". In: *Naval Research Logistics (NRL)* (2004).

[75] Chen Jie and AK Gupta. "Parametric statistical change point analysis". In: *Birkh User* (2000).

[76] Shivaram Kalyanakrishnan et al. "PAC Subset Selection in Stochastic Multi-armed Bandits." In: *ICML*. Vol. 12. 2012, pp. 655–662.

[77] Shafi Kamalbasha and Manuel J. A. Eugster. *Bayesian A/B Testing for Business Decisions*. 2020. arXiv: 2003.02769 [stat.AP].

[78] Varun Kanade, Zhenming Liu, and Bozidar Radunovic. "Distributed non-stochastic experts". In: *Advances in Neural Information Processing Systems*. 2012, pp. 260–268.

[79] Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. "On Bayesian upper confidence bounds for bandit problems". In: *Artificial Intelligence and Statistics*. 2012, pp. 592–600.

[80] Emilie Kaufmann and Shivaram Kalyanakrishnan. "Information complexity in bandit subset selection". In: *Conference on Learning Theory*. 2013, pp. 228–251.

[81] Emilie Kaufmann, Nathaniel Korda, and Rémi Munos. "Thompson sampling: An asymptotically optimal finite-time analysis". In: *International Conference on Algorithmic Learning Theory*. Springer. 2012, pp. 199–213.

[82] Godfrey Keller, Sven Rady, and Martin Cripps. "Strategic experimentation with exponential bandits". In: *Econometrica* 73.1 (2005), pp. 39–68.

[83] R. Kerkouche et al. "Node-based optimization of LoRa transmissions with Multi-Armed Bandit algorithms". In: *2018 25th International Conference on Telecommunications (ICT)*. June 2018, pp. 521–526. doi: 10.1109/ICT.2018.8464949.

[84] Jeremias Knoblauch and Theodoros Damoulas. "Spatio-temporal Bayesian on-line changepoint detection with model selection". In: *arXiv preprint arXiv:1805.05383* (2018).

[85] Jeremias Knoblauch, Jack E Jewson, and Theodoros Damoulas. "Doubly Robust Bayesian Inference for Non-Stationary Streaming Data with β-Divergences". In: *Advances in Neural Information Processing Systems*. 2018, pp. 64–75.

[86] Levente Kocsis and Csaba Szepesvári. "Bandit based monte-carlo planning". In: *European conference on machine learning*. Springer. 2006, pp. 282–293.

[87] Levente Kocsis and Csaba Szepesvári. "Discounted ucb". In: *2nd PASCAL Challenges Workshop*. Vol. 2. 2006.

[88] George Konidaris et al. "Constructing skill trees for reinforcement learning agents from demonstration trajectories". In: *Advances in neural information processing systems*. 2010, pp. 1162–1170.

[89] Vasileios Kotsiou et al. "Is local blacklisting relevant in slow channel hopping low-power wireless networks?" In: *Proc. of IEEE ICC*. 2017, pp. 1–6.

[90] Vasileios Kotsiou et al. "Label: Link-based adaptive blacklisting technique for 6tisch wireless industrial networks". In: *Proc. of 20th ACM MSWiM*. 2017, pp. 25–33.

[91] Solomon Kullback and Richard A Leibler. "On information and sufficiency". In: *The annals of mathematical statistics* 22.1 (1951), pp. 79–86.

[92] Tze Leung Lai and Herbert Robbins. "Asymptotically efficient adaptive allocation rules". In: *Advances in applied mathematics* 6.1 (1985), pp. 4–22.

[93] Tze Leung Lai and Haipeng Xing. "Sequential change-point detection when the pre-and post-change parameters are unknown". In: *Sequential analysis* 29.2 (2010), pp. 162–175.

[94] Peter Landgren, Vaibhav Srivastava, and Naomi Ehrich Leonard. "Distributed cooperative decision-making in multiarmed bandits: Frequentist and Bayesian algorithms". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE. 2016, pp. 167–172.

[95] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[96] Peishuo Li et al. "An Adaptive Channel Selection scheme for Reliable TSCH-based Communication". In: *Proc. of IEEE ISWCS*. 2015, pp. 511–515.

[97] Fang Liu, Joohyun Lee, and Ness Shroff. "A change-detection based framework for piecewise-stationary multi-armed bandit problem". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[98] Keqin Liu and Qing Zhao. "Distributed learning in multi-armed bandit with multiple players". In: *IEEE Transactions on Signal Processing* 58.11 (2010), pp. 5667–5681.

[99] *LoRaWAN Specification*. version= V1.0.2. LoRa Alliance. July 2016.

[100] Gary Lorden et al. "Procedures for reacting to a change in distribution". In: *The Annals of Mathematical Statistics* 42.6 (1971), pp. 1897–1908.

[101] Setareh Maghsudi and Mihaela van der Schaar. "A Non-Stationary Bandit-Learning Approach to Energy-Efficient Femto-Caching With Rateless-Coded Transmission". In: *IEEE Transactions on Wireless Communications* 19.7 (2020), pp. 5040–5056.

[102] Odalric-Ambrym Maillard. "Sequential change-point detection: Laplace concentration of scan statistics and non-asymptotic delay bounds". In: *Algorithmic Learning Theory*. 2019, pp. 610–632.

[103] Odalric-Ambrym Maillard, Rémi Munos, and Gilles Stoltz. "A finite-time analysis of multi-armed bandits problems with kullback-leibler divergences". In: *Proceedings of the 24th annual Conference On Learning Theory*. JMLR Workshop and Conference Proceedings. 2011, pp. 497–514.

[104] Shie Mannor and John N Tsitsiklis. "The sample complexity of exploration in the multi-armed bandit problem". In: *Journal of Machine Learning Research* 5.Jun (2004), pp. 623–648.

[105] Joseph Mellor and Jonathan Shapiro. "Thompson Sampling in Switching Environments with Bayesian Online Change Point Detection". In: *Proc. of AISTATS*. 2013.

[106] Eric Moulines and Aurélien Garivier. "On Upper-Confidence Bound Policies for Switching Bandit Problems". In: *Proc. of International Conference on Algorithmic Learning Theory*. 2011.

[107] V Nandhini and MS Geetha Devasena. "Predictive Analytics for Climate Change Detection and Disease Diagnosis". In: *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*. IEEE. 2019, pp. 1152–1155.

[108] Naumaan Nayyar, Dileep Kalathil, and Rahul Jain. "On regret-optimal learning in decentralized multiplayer multiarmed bandits". In: *IEEE Transactions on Control of Network Systems* 5.1 (2016), pp. 597–606.

[109] Gergely Neu. "Explore No More: Improved High-probability Regret Bounds for Non-stochastic Bandits". In: *Proceedings of the 28th International Conference on Neural Information Processing Systems*. NIPS'15. Montreal, Canada: MIT Press, 2015, pp. 3168–3176. url: http://dl.acm.org/citation.cfm?id=2969442.2969593.

[110] Scott Niekum et al. *CHAMP: Changepoint detection using approximate model parameters*. Tech. rep. Carnegie-Mellon Univ Pittsburgh PA Robotics INST, 2014.

[111] James R Norris and James Robert Norris. *Markov chains*. 2. Cambridge university press, 1998.

[112] Ewan S Page. "Continuous inspection schemes". In: *Biometrika* 41.1/2 (1954), pp. 100–115.

[113] Soumya Priyadarsini Panda and Ajit Kumar Nayak. "Automatic speech segmentation in syllable centric speech recognition system". In: *International Journal of Speech Technology* 19.1 (2016), pp. 9–18.

[114] Georgios Z. Papadopoulos et al. "Importance of Repeatable Setups for Reproducible Experimental Results in IoT". In: *Proc. of ACM PE-WASUN*. 2016, pp. 51–59.

[115] Georgios Z. Papadopoulos et al. "Thorough IoT testbed Characterization: from Proof-of-concept to Repeatable Experimentations". In: *Computer Networks* 119 (2017), pp. 86–101.

[116] Vianney Perchet et al. "Batched bandit problems". In: *The Annals of Statistics* 44.2 (2016), pp. 660–681.

[117] Juha Petajajarvi et al. "On the coverage of LPWANs: range evaluation and channel attenuation model for LoRa technology". In: *ITST*. IEEE. 2015.

[118] Aleksey S Polunchenko, Alexander G Tartakovsky, and Nitis Mukhopadhyay. "Nearly optimal change-point detection with an application to cybersecurity". In: *Sequential Analysis* 31.3 (2012), pp. 409–435.

[119] Vishnu Raj and Sheetal Kalyani. "Taming non-stationary bandits: A Bayesian approach". In: *arXiv preprint arXiv:1707.09727* (2017).

[120] Herbert Robbins. "Some aspects of the sequential design of experiments". In: *Bulletin of the American Mathematical Society* 58.5 (1952), pp. 527–535.

[121] Ignacio Rodriguez et al. "Path loss validation for urban micro cell scenarios at 3.5 GHz compared to 1.9 GHz". In: *GLOBECOM*. IEEE. 2013.

[122] Michael Rothschild. "A two-armed bandit theory of market pricing". In: *Journal of Economic Theory* 9.2 (1974), pp. 185–202.

[123] Eric Ruggieri and Marcus Antonellis. "An exact approach to Bayesian sequential change point detection". In: *Computational Statistics & Data Analysis* 97 (2016), pp. 71–86.

[124] Yunus Saatçi, Ryan D Turner, and Carl Edward Rasmussen. "Gaussian Process Change Point Models." In: *ICML*. 2010, pp. 927–934.

[125] Martin Sauter. "Mobility Management in the Cell-DCH State". In: *From GSM to LTE: An Introduction to Mobile Networks and Mobile Broadband (eBook). John Wiley & Sons* (2010), p. 160.

[126] Thilo A Schmitt et al. "Non-stationarity in financial time series: Generic features and tail behavior". In: *EPL (Europhysics Letters)* 103.5 (2013), p. 58003.

[127] Marta Soare, Alessandro Lazaric, and Rémi Munos. "Best-arm identification in linear bandits". In: *Advances in Neural Information Processing Systems*. 2014, pp. 828–836.

[128] Morten Sorensen. "Learning by investing: Evidence from venture capital". In: *AFA 2008 New Orleans Meetings Paper*. 2008.

[129] WirelessHART Specification. "75: TDMA Data-Link Layer". In: *HART Communication Foundation Std., Rev* 1 (2008).

[130] Latanya Sweeney. "k-anonymity: A model for protecting privacy". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570.

[131] Balázs Szörényi et al. "Gossip-based distributed stochastic bandit algorithms". In: *Journal of Machine Learning Research Workshop and Conference Proceedings*. Vol. 2. International Machine Learning Societ. 2013, pp. 1056–1064.

[132] AG Tartakovsky. *Sequential methods in the theory of information systems*. 1991.

[133] Rasool Tavakoli et al. "Dependable interference-aware time-slotted channel hopping for wireless sensor networks". In: *ACM Transactions on Sensor Networks (TOSN)* 14.1 (2018), p. 3.

[134] Cem Tekin and Mingyan Liu. "Online learning of rested and restless bandits". In: *IEEE Transactions on Information Theory* 58.8 (2012), pp. 5588–5611.

[135] William R. Thompson. "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples." In: *Biometrika* 25 (1933), pp. 285–294.

[136] Francesco Trovo et al. "Sliding-window thompson sampling for non-stationary settings". In: *Journal of Artificial Intelligence Research* 68 (2020), pp. 311–364.

[137] Ryan D Turner, Steven Bottone, and Clay J Stanek. "Online variational approximations to non-exponential family change point models: with application to radar tracking". In: *Advances in Neural Information Processing Systems*. 2013, pp. 306–314.

[138] Ryan Turner, Yunus Saatci, and Carl Edward Rasmussen. "Adaptive Sequential Bayesian Change Point Detection". In: *Advances in Neural Information Processing Systems (NIPS): Temporal Segmentation Workshop*. 2009.

[139] Tanguy Urvoy et al. "Generic exploration and k-armed voting bandits". In: *International Conference on Machine Learning*. 2013, pp. 91–99.

[140] Leslie G Valiant. "A theory of the learnable". In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. ACM. 1984, pp. 436–445.

[141] Nadège Varsier and Jean Schwoerer. "Capacity limits of LoRaWAN technology for smart metering applications". In: *Proc. of IEEE ICC*. 2016. doi: 10.1109/ICC.2017.7996383.

[142] Paolo Viappiani. "Thompson sampling for Bayesian bandits with resets". In: *Proc. of International Conference on Algorithmic Decision Theory*. Springer. 2013, pp. 399–410.

[143] Christopher JCH Watkins and Peter Dayan. "Q-learning". In: *Machine learning* 8.3-4 (1992), pp. 279–292.

[144] Thomas Watteyne, Ankur Mehta, and Kris Pister. "Reliability Through Frequency Diversity: Why Channel Hopping Makes Sense". In: *PE-WASUN*. ACM. 2009.

[145] Robert C Wilson, Matthew R Nassar, and Joshua I Gold. "Bayesian online learning of the hazard rate in change-point problems". In: *Neural computation* 22.9 (2010), pp. 2452–2476.

[146] Yanhong Wu. *Inference for change point and post change means after a CUSUM test*. Vol. 180. Springer, Lecture Notes in Statistics, 2007.

[147] Xiang Xuan and Kevin Murphy. "Modeling changing dependency structure in multivariate time series". In: *Proceedings of the 24th international conference on Machine learning*. 2007, pp. 1055–1062.

[148] Dimitrios Zorbas, Georgios Z. Papadopoulos, and Christos Douligeris. "Local or Global Radio Channel Blacklisting for IEEE 802.15.4-TSCH Networks?" In: *Proc. of IEEE ICC*. 2017, pp. 1–6.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Bandits à mémoire pour la prise de décision en environnement dynamique. Application à l'optimisation des réseaux de télécommunications.

Mots clés: Bandit non-stationnaire, détection de point de changement, optimisation des réseaux de télécommunication, exploration pour les bandits.

Résumé: Dans cette thèse de doctorat, nous étudions le problème du bandit manchot non stationnaire où le comportement de non-stationnarité de l'environnement est caractérisé par plusieurs changements brusques appelés "points de changement". Nous proposons les bandits à mémoire : une combinaison entre un algorithme pour le bandit manchot stochastique et le détecteur Bayésien de point de changement. L'analyse de ce dernier a toujours été un problème ouvert dans la communauté de la théorie statistique et de l'apprentissage séquentiel. Pour cette raison, nous dérivons une variante du détecteur Bayésien de point de changement qui est plus facile à analyser mathématiquement en termes de taux de fausses alarmes et de délai de détection (qui sont les critères les plus courants pour la détection de point de changement). Ensuite, nous introduisons le problème d'exploration décentralisée dans le cadre du bandit manchot où un ensemble de joueurs collaborent pour identifier le meilleur bras en interagissant de manière asynchrone avec le même environnement stochastique. Nous proposons une première solution générique appelée élimination décentralisée qui utilise n'importe quel algorithme d'identification du meilleur bras comme sous-programme avec la garantie que l'algorithme assure la confidentialité, avec un faible coût de communication. Enfin, nous effectuons une évaluation des stratégies de bandit manchot dans deux contextes différents de réseaux de télécommunications. Tout d'abord, dans le contexte LoRaWAN (Long Range Wide Area Network), nous proposons d'utiliser des algorithmes de bandit manchot à la place de l'algorithme par défaut qui porte le nom d'ADR (Adaptive Data Rate) afin de minimiser la consommation d'énergie et les pertes de paquets des terminaux. Ensuite, dans le contexte IEEE 802.15.4-TSCH, nous effectuons une évaluation de 9 algorithmes de bandits manchot afin de sélectionner ceux qui choisissent les canaux les plus performants, en utilisant les données collectées via la plateforme FIT IoT-LAB. L'évaluation des performances suggère que notre proposition peut améliorer considérablement le taux de livraison des paquets par rapport à la procédure TSCH par défaut, augmentant ainsi la fiabilité et l'efficacité énergétique des transmissions.

Title: Memory bandits for decision making in dynamic environments. Application to network optimization.

Keywords: Non-stationary multi-armed bandits, online change-point detection, network optimization, exploration in multi-armed bandits.

Abstract: In this PhD thesis, we study the non-stationary multi-armed bandit problem where the non-stationarity behavior of the environment is characterized by several abrupt changes called "change-points". We propose Memory Bandits: a combination between an algorithm for the stochastic multi-armed bandit and the Bayesian Online Change-Point Detector (BOCPD). The analysis of the latter has always been an open problem in the statistical and sequential learning theory community. For this reason, we derive a variant of the Bayesian Online Change-point detector which is easier to mathematically analyze in term of false alarm rate and detection delay (which are the most common criteria for online change-point detection). Then, we introduce the decentralized exploration problem in the multi-armed bandit paradigm where a set of players collaborate to identify the best arm by asynchronously interacting with the same stochastic environment. We propose a first generic solution called decentralized elimination: which uses any best arm identification algorithm as a subroutine with the guarantee that the algorithm ensures privacy, with a low communication cost. Finally, we perform an evaluation of the multi-armed bandit strategies in two different context of telecommunication networks. First, in LoRaWAN (Long Range Wide Area Network) context, we propose to use multi-armed bandit algorithms instead of the default algorithm ADR (Adaptive Data Rate) in order to minimize the energy consumption and the packet losses of end-devices. Then, in a IEEE 802.15.4-TSCH context, we perform an evaluation of 9 multi-armed bandit algorithms in order to select the ones that choose high-performance channels, using data collected through the FIT IoT-LAB platform. The performance evaluation suggests that our proposal can significantly improve the packet delivery ratio compared to the default TSCH operation, thereby increasing the reliability and the energy efficiency of the transmissions.

Maison du doctorat de l'Université Paris-Saclay
2ème étage aile ouest, Ecole normale supérieure Paris-Saclay
4 avenue des Sciences,

91190 Gif sur Yvette, France