Multi-armed bandit based policies for cognitive radio's decision making issues

Wassim Jouini
SUPELEC/IETR
wassim.jouini@supelec.fr

Damien Ernst
University of Liège
dernst@ulg.ac.be

Christophe Moy
SUPELEC/IETR
christophe.moy@supelec.fr

Jacques Palicot
SUPELEC/IETR
jacques.palicot@supelec.fr

***Abstract*—We suggest in this paper that many problems related to Cognitive Radio's (CR) decision making inside CR equipments can be formalized as Multi-Armed Bandit problems and that solving such problems by using Upper Confidence Bound (UCB) algorithms can lead to high-performance CR devices. An application of these algorithms to an academic Cognitive Radio problem is reported.**

***Index Terms*—Cognitive Radio, Decision making, Multi-Armed Bandit, Upper Confidence Bound Algorithm.**

I. Introduction

Today's radio devices need a specific dedicated electronic chain for each standard. With the growth of the number of these standards (GSM, EDGE, Wi-Fi, etc) in one equipment, the design and development of these radio devices has become a real challenge. Recent hardware advances have offered the possibility to design software solutions to problems which were requiring in the past hardware signal processing devices. These advances are part of a field called Software Defined Radio (SDR). With this added software layer, it is technically possible now to control a large number of parameters in order to operate the radio devices with great flexibility and efficiency (e.g., change the bandwidth of the devices, switch from one communication protocol to another, minimize the energy consumption of a device, and so on). Soon after the emergence of the SDR field, researchers have studied ways to control at best these parameters, especially when the radio devices are used in dynamic and partially unknown environments. This has lead to the emergence of a new research field, named Cognitive Radio (CR), a term introduced by J. Mitola [1] in 1999. Cognitive Radio presents itself as a set of concepts and technologies that enable radio equipments to have the autonomy and the cognitive abilities to become aware of their environment as well as of their own operational abilities. The purpose of this new concept is to meet the user's expectations and to maximize operators' resources usage (e.g. spectral resource allocation) without compromising the efficiency of the network. Thus, it presupposes the capacity to collect information from its surrounding environment (perception), to digest it (learning and decision making problems) and to act in the best possible way by considering several constraints. Therefore, it is a new paradigm of wireless communication whose purpose is to combine Software Defined Radio technologies and cognitive abilities. At the level of an equipment it gives very promising perspectives but also raises design challenges.

One challenge faced when designing a cognitive engine, seen as a cognitive agent within the cognitive radio framework, is the following. The agent must gather information on its environment (i.e., explore its environment) to be able to improve its decision strategy while at the same time, it has to exploit at best its current knowledge of the environment to make good decisions. Hence, the agent faces a tradeoff between exploring its environment and exploiting the current information it has. This tradeoff has been studied in the literature for different types of environments such as: environments having a linear dynamics [2] [3], Markov Decision Processes (MDP) [4] or Multi-Armed Bandit (MAB) problems [5].

The MAB mapping problems are particular instances of MDP problems for which the space state is reduced to a single element. As we will see later in Section II, many problems faced when designing the CR engine can be well modeled as MAB problems. In 2002, Peter Auer proposed in his seminal paper [6] a simple yet efficient approach for solving MAB problems. This approach based on the computation of some Upper Confidence Bound indexes (UCB) has since then received a lot of attention in the machine learning community. UCB based algorithms have been shown to behave well on some complex problems (e.g, [7] [8]). In this paper, we suggest that algorithms based on this UCB approach could also be helpful to address many challenges faced in CR context.

The rest of this paper is organized as follows. Section II starts by giving a brief overview of the various challenges faced when designing a cognitive radio engine and, afterwards, suggests that many of these problems can be formalized as (variants of) Multi-Armed Bandit problems. Section III formally defines the MAB problem considered in this paper. Section IV describes a class of algorithms - named Upper Confidence Bound algorithms - for solving MAB problems and motivates the use of such algorithms in the Cognitive Radio framework. Section V reports some simulations results obtained on the CR Dynamic Configuration Adaptation problem. And, finally, Section VI concludes.

II. Cognitive radio decision making framework

A. Cognitive radio challenges

A Cognitive Radio (CR) device is a communication system aware of its environment as well as of its operational abilities and capable of using them intelligently. Thus it is a device that has the ability to collect information through its sensors and

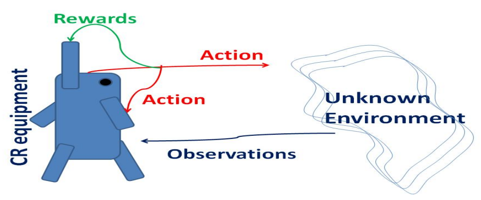

Fig. 1. Cognitive Radio context.

that can use that information to adapt itself to its surrounding environment. It presupposes cognitive abilities that enable CR equipments to deal with all the collected information in order to take appropriate decisions [1]. For that purpose, we refer to as Cognitive Agent (CA) the decision making engine of the CR equipment that can be seen as the brain of the CR device. Designing a CA for CR equipments is challenging for the following reasons (see Figure 1):

1) The environment in which the CA operates is stochastic and unknown.
2) The CR device has multiple objectives that involve trade-off. For instance a CR should minimize its power consumption, while maximizing its broadcasting range.

   To the environment model we add user oriented communication constraints that should influence the cognitive agent's design:
3) A CR equipment must behave at least as well as current non-CR equipments.
4) The CA should lead to a service that improves over time.
5) The algorithms used by the CA should be able to operate with limited memory and computational resources in order to be embedded in a CR equipment.

To answer some of these challenges, several papers have proposed to use different tools borrowed from the "Artificial Intelligence" community such as genetic algorithms [9] or adaptive neuronal networks [10]. Moreover, the evaluation of the best *configuration* for a CR terminal or a CR network lead some researchers to use fuzzy logic [11] or Bayesian approaches [12] combined with other optimization techniques.

In this paper we propose to borrow algorithms introduced for solving MAB problems to design cognitive agents.

B. *Cognitive agent*

In this paper, we assume that the CA can only take actions at discrete time instants $t = 0, 1, 2, \ldots$. At every instant t, the CA observes its environment and collects different kind of information (e.g., signal to noise rate, throughput, position, etc.). All the information collected by the CA up to instant t is supposed to be gathered in a vector i_t. We assume that the CA has to select at every instant t an action a_t in the discrete set $\mathcal{A}$. Without loss of generality, the behavior of the CA can be seen as a function π that maps the information vector i_t into the action $a_t \in \mathcal{A}$, that is:

$$a_t = \pi(i_t) \tag{1}$$

We denote by Π the set of all possible functions π that map the information vector into an action. Designing a good CA for a specific CR application can be seen as finding a function in Π offering good performances. There are two main difficulties related to the search for a good policy in Π when designing a CA for a CR device. First, since a CA must satisfy at best several objectives and constraints, it may be difficult to grade the performance of a policy or even to distinguish between two policies $\pi_1, \pi_2 \in \Pi$ which one is the best. The second one is related to the fact that it is difficult to predict how well a specific policy will be able to take appropriate decisions when used in real-life.

To specifically address these two problems, we will assume that for many CR related decision issues:

1) The CA can define at instant $t+1$ based on the information contained in the vector i_{t+1} a numerical reward r_t which scores its behavior over the interval $]t, t+1]$ - it represents how well the different objectives and constraints have been satisfied over this time interval.
2) The environment in which the CA evolves in such that every reward r_t can be seen as the realization of a random function which depends on the current action a_t and of some environment-dependent parameters.

Intuitively a good policy should be able to overcome the lack of information by exploiting in an appropriate way the information gathered by the CR on the environment.

Under these assumptions, the problem of finding a good policy π can be seen as the problem of finding a good policy for a MAB problem, a well-studied machine learning problem. Multi-armed bandit problems will be presented in the next two sections (Section III and IV) and algorithms proposed for solving these problems will be applied to the design of a CA for an academic radio device in Section V.

III. Multi-armed bandit

In Section II, we suggested that many CR related decision making issues may be formalized as a MAB problem. In this section we detail the MAB framework and discuss the notion of regret, of consistency as well as order optimality of a policy. Later in Section IV, we will detail some algorithms proposed by the machine learning community for solving MAB problems.

A. *Multi-armed bandit framework*

A multi-armed bandit is a simple machine learning problem based on an analogy with the traditional slot machine (one armed bandit) but with more than one lever. When pulled at a time $t = 0, 1, 2, \ldots$, each lever (or machine) $k \in \{k = 1, \ldots, K\}$ provides a reward r_t drawn from a distribution θ_k associated to that specific lever. The objective of the gambler is to maximize the collected reward sum through iterative pulls. It is classically assumed that the gambler has no initial knowledge about the levers. However it is important to understand that many CR applications provide substantial information that shall be used to design better policies. The crucial tradeoff the gambler faces at each trial is between "exploitation" of the lever that has the highest expected payoff and "exploration" to get more information about the expected payoffs of the other levers. In this paper we assume that the different payoffs drawn from a machine are independent and identically distributed (i.i.d.) and that the independence of the rewards holds between the machines. However the different machines reward distributions $\{\theta_1, \theta_2, ..., \theta_K\}$ are not supposed to be the same.

B. Regret

Let I_t denote the machine selected at a time t and $i_t = [I_0, r_0, I_1, r_1, \ldots, I_{t-1}, r_{t-1}]$ the information vector available to the gambler at instant t. We assume that the gambler uses a policy π to select at instant t the arm I_t s.t.

$$I_t = \pi(i_t)$$

Moreover $\forall k \in \{1, ..., K\}$, let $\mu_k \overset{\Delta}{=} E(\theta_k)$

The regret of a policy $\pi \in \Pi$ at time t (after t pulls) is defined as follows:

$$R_t^\pi = t.\mu^* - \sum_{m=0}^{t-1} r_m \tag{2}$$

where $\mu^* = \max_k\{\mu_k\}$ refers to the expected reward of the optimal arm.

The general idea behind the "regret" can be summarized as follows: if the gambler knew *a priori* which one was the best arm, he would only pull that one and hence maximize the expectancy of the collected rewards. However, since he lacks that essential information he will suffer unavoidable loss due to suboptimal pulls.

We seek to find a policy that minimizes the expected cumulated regret (Equation (3)), while having no information on the reward distributions. The expected cumulated regret can be written as follows:

$$E[R_t^\pi] = \sum_{k=1}^{K} \Delta_k.E[T_k(t)] \tag{3}$$

where $\Delta_k = \mu^* - \mu_k$ and $T_k(t)$ refers to the number of times the machine k has been played from instant 0 to instant $t-1$.

C. Consistency and order optimality

A policy π is said to be *β-consistent*, $0 < \beta \leq 1$, if it satisfies:

$$\lim_{t\to\infty} \frac{E[R_t^\pi]}{t^\beta} = 0 \tag{4}$$

We expect our policies to be at least *1-consistent*. As a matter of fact, this property ensures that asymptotically the mean expected reward is optimal, i.e.:

$$\lim_{t\to\infty} \frac{\sum_{m=0}^{t-1} r_m}{t} = \mu^* \tag{5}$$

Considering a MAB problem Robbins and Lai have shown [13] that if we consider families of distributions indexed with a single real parameter, then there exist *β-consistent* policies for all $\beta > 0$. Moreover, they explicitly developed such policies for particular distributions (Bernoulli, Gaussian, Exponential, etc.) and proved that the expected cumulated regret of optimal policies grows at least as a logarithmic function of the number of steps t. However the policies introduced in [13] where based on complex indexes whose computation is burdensome. The machine learning community dealing with MAB issues aim at finding simple policies π verifying at least:

$$\lim_{t\to\infty} \frac{E[R_t^\pi]}{\ln(t)} \leq C \tag{6}$$

where C is a positive real function of the arm distributions parameters (e.g., expected mean, variances, etc.). Policies verifying this property are referred to as "order optimal" policies. Moreover, a policy that satisfies the following inequality:

$$E[R_t^\pi] \leq C.\ln(t) \tag{7}$$

for every t is said to be "order optimal uniformly over time".

IV. Sample mean based upper confidence bound algorithms

As we have explained earlier, building a CA for a CR device requires to find a policy π for this agent that offers good performances. In this section, we will propose an approach for designing well-performing policies π when the CA faces a decision problem that can be formalized as a MAB problem, as we believe it is often the case for cognitive radio's decision making problems. The approach is based on the computation of Upper Confidence Bound (UCB) indexes introduced first in [14]. From the UCB indexes computed from the information vector i_t at time t, the action a_t can be inferred in a straightforward way. As we will see later in this section, these UCB based policies offer good performance guarantees and lend themselves to software implementations compliant with the limited computational resources of a CA embedded in a CR device. Later in Section V, the empirical performances of these policies will be evaluated on an academic CR problem.

At every instant t, an upper confidence bound index is computed for every machine k. This upper confidence bound index, denoted by $B_{k,t,T_k(t)}$, is computed from i_t and gives an optimistic estimation of the expected reward of machine k.

Let $B_{k,t,T_k(t)}$ denote the index of the policies we are dealing with:

$$B_{k,t,T_k(t)} = \overline{X}_{k,T_k(t)} + A_{k,t,T_k(t)} \tag{8}$$

where $\overline{X}_{k,T_k(t)}$ is the sample mean of the machine k after been played $T_k(t)$ times at the step t, and $A_{k,t,T_k(t)}$ is an upper confidence bias added to the sample mean.

A policy π computes from i_t these indexes from which it deduces an action a_t as follows:

$$a_t = \pi(i_t) = \arg\max_k(B_{k,t,T_k(t)}) \tag{9}$$

We describe hereafter two specific upper confidence biases $A_{k,t,T_k(t)}$ that will be used in our simulations and discuss the theoretical properties of the policies associated to these indexes.

```
Parameters: $K$, exploration coefficient $\alpha$
Input: $i_t = [I_0, r_0, I_1, r_1, \dots, I_{t-1}, r_{t-1}]$
Output: $a_t$
Algorithm:

If: $t \leq K$ return $a_t = t + 1$
Else:
  • $T_k(t) \leftarrow \sum_{m=0}^{t-1} \mathbf{1}_{\{I_m=k\}}$,[1] $\forall k$
  • $A_{k,t,T_k(t)} \leftarrow \sqrt{\frac{\alpha.\ln(t)}{T_k(t)}}$, $\forall k$
  • $B_{k,t,T_k(t)} \leftarrow \frac{\sum_{m=0}^{t-1} r_m.\mathbf{1}_{\{I_m=k\}}}{T_k(t)} + A_{k,t,T_k(t)}$, $\forall k$
  • return $a_t = \arg\max_k(B_{k,t,T_k(t)})$
```

Fig. 2. A tabular version of a $\pi(i_t)$ policy using a UCB_1 algorithm for computing actions a_t.

1) UCB_1: When using the following upper confidence bias:

$$A_{k,t,T_k(t)} = \sqrt{\frac{\alpha.\ln(t)}{T_k(t)}} \tag{10}$$

with $\alpha > 1$, we obtain an upper confidence bound index referred to as UCB_1 in the literature. A fully detailed version of the algorithm using UCB_1 indexes to select actions is given in Figure 2. Under some mild assumptions, given in the following theorem, a UCB policy using this index is order optimal uniformly over time.

Theorem 1: (cf. [15] for proofs) For all $K \geq 2$, if policy $UCB_1(\alpha > 1)$ is run on K machines/arms having arbitrary reward distributions $\theta_1, ..., \theta_K$ with support in [0,1], then:

$$E[R_t^{\pi=UCB_1}] \leq \sum_{k:\Delta_k>0} \frac{4.\alpha}{\Delta_k}.\ln(t) \tag{11}$$

Notice that a similar theorem could be written if the reward distributions had a bounded support rather than a support in [0,1].

2) UCB_V: A UCB_V policy refers to a policy which uses as upper confidence bias:

$$A_{k,t,T_k(t)} = \sqrt{\frac{2\xi.V_k(t).\ln(t)}{T_k(t)}} + \frac{3.c.\xi.\ln(t)}{T_k(t)} \tag{12}$$

The UCB_V upper confidence bound index was first introduced in [3]. In the same research paper, the authors have also proven the theorem given hereafter which shows that UCB_V policies are also order optimal uniformly over time.

Theorem 2: (cf. [15] for proofs) For all $K \geq 2$, if policy $UCB_V(\xi \geq 1, c = 1)$ is run on K machines/arms having arbitrary reward distributions $\theta_1, ..., \theta_K$ with support in [0,1], then $\exists C_\xi > 0$ s.t.

$$E[R_t^{\pi=UCB_V}] \leq C_\xi \sum_{k:\Delta_k>0} (\frac{\sigma_k^2}{\Delta_k} + 2).\ln(t) \tag{13}$$

Actually a similar result would still hold if $c \neq 1$ but satisfies nonetheless $3.\xi.c > 1$.

By anticipating on the simulation results that will be reported in the next section, the UCB_V index performs better on our test problem that the UCB_1 index. This is due to the fact that by adapting its behavior according to the empirical variance of every arm (see the term $\sqrt{\frac{2\xi.V_k(t).\ln(t)}{T_k(t)}}$ in Equation (12)), a UCB_V based policy is able to better address the exploration-exploitation tradeoff. Other authors have also noticed that by using upper confidence bound indexes based on the empirical variance, better performances can be obtained (see, e.g., [6]).

A cognitive agent which exploits the tabular version of the UCB_1 algorithm given in Figure 2 or its UCB_V counterpart will have at every instant t to carry out a number of operations which is proportional to t and store an information vector whose length grows linearly with t. Therefore, after a certain time of interaction with its environment, a CA having limited computational and memory resources is going to be unable to store the information vector i_t and to process it fast enough. To overcome this problem, one can implement the UCB_1 and UCB_V policies in such a way that part of the solution computed at time t can be used at time $t+1$, and so that the time required to compute a new solution can be bounded and the memory requirements independent from t.

This can be achieved by noticing that the upper confidence bound index from which the action a_t is computed at time t (see Equations (10) and (12)) are functions of the arguments $\overline{X}_{k,T_k(t)}$, $T_k(t)$ and also $V_k(t)$ for UCB_V and that these arguments can be computed from the only knowledge of their values at time $t-1$, I_{t-1} and r_{t-1}. Indeed if $k = I_{t-1}$ we have:

$$T_k(t) = T_k(t-1) + 1 \tag{14}$$

$$D \triangleq r_{t-1} - \overline{X}_{k,T_k(t-1)} \tag{15}$$

$$\overline{X}_{k,T_k(t)} = \overline{X}_{k,T_k(t-1)} + \frac{D}{T_k(t)} \tag{16}$$

$$V_k(t) = \frac{V_k(t-1) + D.(r_{t-1} - \overline{X}_{k,T_k(t)})}{T_k(t-1)} \tag{17}$$

and if $k \neq I_{t-1}$, $T_k(t)$, $\overline{X}_{k,T_k(t)}$, $V_k(t)$ are equal to $T_k(t-1)$, $\overline{X}_{k,T_k(t-1)}$, $V_k(t-1)$, respectively.

V. Application to radio configuration

A. Dynamic configuration adaptation problem

In this section we consider CR equipments which have the flexibility to use K different configurations. Each configuration is defined by a set of parameters (e.g., modulation/coding rate, equalizers, etc.). The goal of a CA is to select the best configuration. We call this problem the "Dynamic Configuration

[1]Indicator function: $\mathbf{1}_{\{logical\_expression\}}$={1 if logical_expression=true ; 0 if logical_expression=false}.

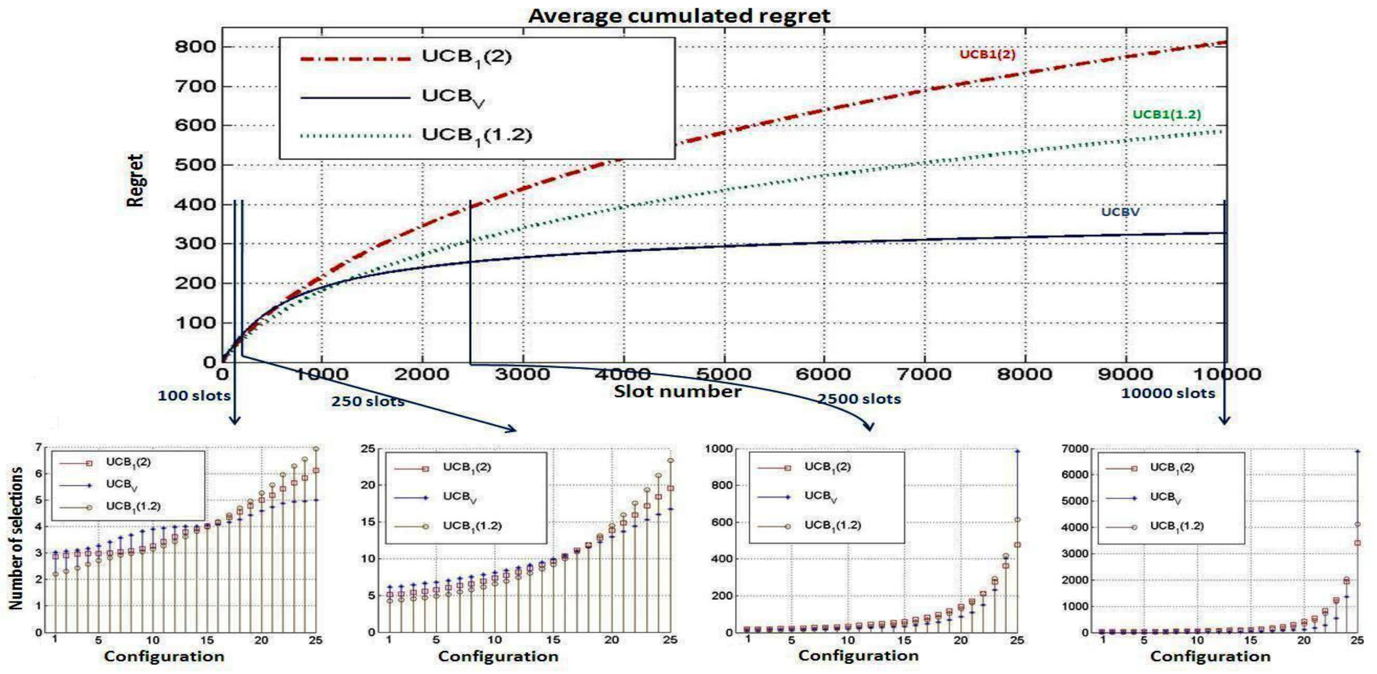

Fig. 3. UCB based policies and dynamic configuration problem: simulation results. Figure on top plots the average cumulated reward as a function of the number of slots for the different UCB based policies. The figures on the bottom represent the number of times every configuration has been selected after a specific number of slots. From the left to the right, 100 slots, 250 slots, 2500 slots and 5000 slots.

1-CR equipment:

- *K possible configurations C_k, $k \in \{k = 1, ..., K\}$, verifying the operational constraints but with unknown performances.*
- *A cognitive agent: can learn and make decisions to help the CR equipment to improve its behavior.*

2-Time representation:

- *Time divided into slots $t = 0, 1, 2, ...$ (Figure 5)*
- *At the beginning of every slot t, the CA decides to reconfigure or not the CR equipment.*

3-Environment and performance evaluation:

- *Typical observations: SNR, BER, network load, throughput, spectrum bands, etc.*
- *A numerical signal is computed at the end of every slot t and informs the CA of the performance of the CR equipment. The numerical signal obtained when using configuration C_k is a function of the observations and the configurations.*
- *The numerical results computed with a configuration C_k are assumed to be i.i.d. and drawn from an unknown stochastic distribution θ_k.*

Fig. 4. Description of the Dynamic Configuration Adaptation problem.

Adaptation" (DCA) problem. The DCA problem is further described in Figure 4. In the rest of this section a particular instance of the DCA problem is analyzed.

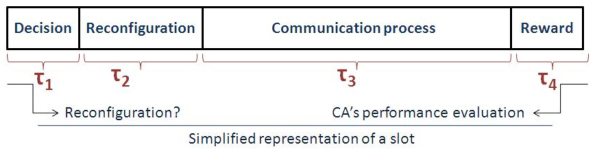

Fig. 5. Slot representation for a radio equipment controlled by a CA. A slot is divided into 4 periods. During the first period, the CA chooses the next configuration. If the new configuration is different from the current one, a reconfiguration is carried out during the second period before communicating. If a reconfiguration is not needed, the CR equipment keeps the current configuration to communicate. At the end of every slot, the CA computes a reward that evaluates its performance during the communication process. It is assumed here that $\tau_1 + \tau_2 + \tau_4$ are small with respect to τ_3.

B. Experimental protocol

The CA can choose between 25 configurations. To every of these configurations is associated a reward distribution which is Gaussian distribution truncated to the interval [0,1]. The pdf of such a distribution is given by

$$\frac{Gauss_{(\mu,\sigma^2)}(\cdot)}{E_{X\sim Gauss_{(\mu,\sigma^2)}}[\mathbf{1}_{\{X\in[0,1]\}}]} \tag{18}$$

where $Gauss_{(\mu,\sigma^2)}(\cdot)$ refers to the pdf of a non-truncated Gaussian distribution having a mean μ and a standard deviation σ. The parameter σ has always been chosen equal to 0.1 in our simulations. The parameter μ differs from one distribution to another and has been selected by drawing a number at random and with uniform probability in $[0, 1]$ for every configuration.

Every numerical result reported hereafter is the average of the values obtained over 100 experiments. For each experi-

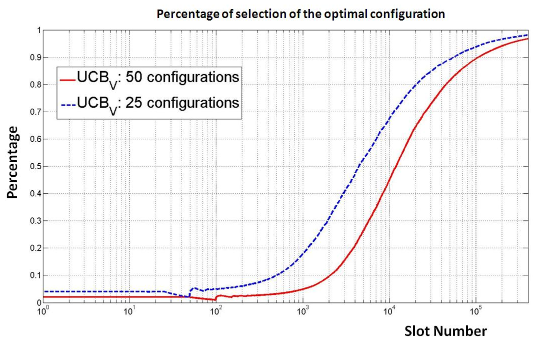

Fig. 6. Percentage of times a UCB-based policy selects the optimal configuration.

ment, new reward distributions are first generated. To ease the presentation of the results, we will in each experiment refer by k the configuration to which is associated the reward distribution having the kth smallest mean.

In this section, the parameter α of the UCB_1 algorithm is chosen either equal to 1.2 (in which case the algorithm is referred to as $UCB_1(1.2)$) or to 2 (referred to as $UCB_1(2)$). The parameters ξ and c of the UCB_V algorithm are equal to 1 and 0.4, respectively.[2]

C. Simulation results

Figure 3-top shows the evolution of the average cumulated regret for the different UCB policies. For all three policies, the cumulated regret first increases rather rapidly with the slot number and then more and more slowly. This shows that UCB policies are able to process the past information in an appropriate way such that configurations leading to high rewards are favored with time. This is further illustrated by the four graphics on the bottom of Figure 3. These graphics show the number of times every individual configuration has been selected after a specific slot number. As we observe, the UCB policies indeed select more often the best configurations when the slot number increases. The coefficient α seems to affect significantly the performance of the UCB_1 policies (see Figure 3). This suggests that tuning well the parameters of UCB based policies is important. With respect to this particular parameter α, Theorem 1 suggests to take α as close as possible to 1 to have the smallest possible upper bound on the expected cumulated regret. As we can observe, we have indeed obtained better results with the smallest value of α.

In this academic problem, the number of possible configuration was relatively small (25). One may wonder how the UCB policies would scale up to larger sets of configurations. In an attempt to answer this question, we have run simulations by using 50 configurations. The results are reported on Figure 6. The bold curve represents for different number of slots the percentage of times the optimal configuration was selected when using 50 configurations. The dashed curve shows the results obtained with 25 configurations. As we observe, the dashed curve stands well-above the plain one when the number of slots is small. When the numbers of slots starts growing, the distance between both curves decreases and almost vanishes after a large number of slots. These results suggest that when dealing with larger set of configurations, UCB based policies may still lead to acceptable performances if the number of slots is large enough.

[2]With such values for c and ξ, the condition $3.\xi.c > 1$ is satisfied and the bound on the expected cumulated regret given by Equation (13) still holds.

VI. CONCLUSION

In this paper, we have proposed a new approach inspired from work done on the MAB problem - a well-studied problem in machine learning - for tackling cognitive radio's decision making issues. Though this research is still in its infancy, we believe that this approach might have a bright future in the field of Cognitive Radio and compare favorably with those using other techniques borrowed from the machine learning/optimization community (Bayesian approaches, meta heuristic optimization techniques, reinforcement learning, etc.).

ACKNOWLEDGMENT

Damien Ernst is a Research Associate of the Belgian FRS-FNRS of which he acknowledges the financial support.

REFERENCES

[1] J. Mitola and G.Q. Maguire. Cognitive radio: making software radios more personal. *Personal Communications, IEEE*, 6:13–18, August 1999.
[2] A.A. Feldbaum. Dual control theory, parts i. *Automation and Remote Control*, 21, No. 9:874–880, April 1961.
[3] A.A. Feldbaum. Dual control theory, parts ii. *Automation and Remote Control*, 21, No. 11:1033–1039, May 1961.
[4] A.N. Burnetas and M.N. Katehakis. Optimal adaptive policies for markov decision processes. *Mathematics of Operations Research*, 22(1):222–255, 1997.
[5] H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of American Mathematical Society*, 58:527–535, 1952.
[6] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite time analysis of multi-armed bandit problems. *Machine learning*, 47(2/3):235–256, 2002.
[7] J.F. Hren and R. Munos. Optimistic planning in deterministic systems. *European Workshop on Reinforcement Learning*, pages 151–164, 2008.
[8] L. Kocsis and Cs. Szepesvari. Bandit based monte-carlo planning. *In 15th European Conference on Machine Learning (ECML)*, pages 282–293, 2006.
[9] C.J. Rieser. *Biologically Inspired Cognitive Radio Engine Model Utilizing Distributed Genetic Algorithms for Secure and Robust Wireless Communications and Networking*. PhD thesis, Virginia Tech, 2004.
[10] N. Colson, A. Kountouris, A. Wautier, and L. Husson. Cognitive decision making process supervising the radio dynamic reconfiguration. In *Proceedings of Cognitive Radio Oriented Wireless Networks and Communications*, page 7, 2008.
[11] N. Baldo and M. Zorzi. Fuzzy logic for cross-layer optimization in cognitive radio networks. *IEEE Consumer Communications and Networking Conference*, January 2007.
[12] L. Lai, H.E. Gamal, H.J. Jiang, and V. Poor. Cognitive medium access: Exploration, exploitation and competition. *[Online]. Available: http://arxiv.org/abs/0710.1385*.
[13] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.
[14] R. Agrawal. Sample mean based index policies with o(log(n)) regret for the multi-armed bandit problem. *Advances in Applied Probability*, 27:1054–1078, 1995.
[15] J.-Y. Audibert, R. Munos, and C. Szepesvári. Tuning bandit algorithms in stochastic environments. In *Proceedings of the 18th international conference on Algorithmic Learning Theory*, 2007.