Channel Sensing Order for Cognitive Radio Networks Using Reinforcement Learning

Andre C. Mendes and Carlos Henrique P. Augusto and Marcel W. R. da Silva and
Raphael M. Guedes and Jose F. de Rezende
Grupo de Teleinformática e Automação
Universidade Federal do Rio de Janeiro (UFRJ)
P.O. Box 68.504 – Rio de Janeiro, RJ 21.945–970
Emails: {andre,chenrique,marcel,raphael,rezende}@gta.ufrj.br

***Abstract*—This work investigates the problem of channel sensing order used by a cognitive multichannel network, where each user is able to perform primary user detection on only one channel at a time. The sensing order indicates the sequence of channels sensed by the secondary users when searching for an available channel. When using an optimal sensing order, the secondary user can find faster a free channel with high quality. Brute-force algorithms may be used to find the optimal sensing order. However, this approach requires great computational effort. Even in scenarios where the secondary user knows the probability of each channel being available, the sensing order where the most available channels are sensed first is not ideal when using adaptive modulation. Therefore, we propose an approach using reinforcement learning to search dynamically for the optimal sensing order. Through simulations, we evaluated our proposal and compared its performance with other mechanisms, and the results obtained are close to the optimal value provided by the brute-force and superior to the other mechanisms in most of the scenarios.**

I. Introduction

The increasing demand for spectrum, along with the inefficient use of some licensed bands [1], pushed the idea of freeing those underutilized bands for dynamic and opportunistic spectrum access [2]. This kind of access requires reconfigurable network devices, called *cognitive radios*, capable of adapting their behavior in response to environmental stimuli [3]. Basically, cognitive radios can only access a certain band of the spectrum when the users primarily licensed to this band are inactive. For this, these cognitive devices, called secondary users, need to determine by spectrum sensing when primary users are active in order to avoid causing them a harmful interference. Therefore, spectrum sensing is of most importance in the correct operation of these devices.

In a single-radio and multiple-channel environment, secondary users can only sense one channel at a time to identify potential opportunities for transmissions. In this scenario, the channel sensing order can severely influence the performance of cognitive networks. Thus, the search of the optimal channel sensing order is a problem of great interest.

Recent works have tackled this problem [4]–[8]. Most of them use the optimal stopping theory [9] to find the best channel sensing order to be used. In these works, time is divided in slots. And, at each slot, a secondary user senses channels following a specific sequence until it finds a free channel. In this case, the user can use this channel for transmission for the remaining time of the slot, i.e. the effective time of the slot used for transmission. With a priori knowledge of the channel's achievable rates and availability probabilities, it is possible to calculate the expected reward of a specific sensing order in terms of the *effective transmission rate*, i.e. the product between the achievable rate and the effectiveness of a slot. Therefore, one can find the optimal sequence of channels by computing the expected reward of each possible sequence and by choosing the sequence that reaches the best reward. However, the complexity of this *brute force* approach for N channels is $O(N.N!)$, considering that the computation of the expected reward for each sequence is $O(1)$.

In order to reduce the computational complexity of this search for optimal sensing order, the authors in [7], [10] provide sub-optimal solutions. The first solution uses dynamic programming [7] and has complexity of $O(N.2^{N-1})$, while the second one uses decision trees [10] with a complexity of $O(N^3)$. The latter work evaluates the performance of these two solutions in face of different degrees of primary activity, and compare them to the randomly ordered sequence and to the sequence in decreasing order of channel availabilities, referred as "intuitive sequence" in [7].

In [8], the authors propose ordering the channels in descending order of their achievable rates. This eliminates the necessity of a priori knowledge about the activity of the primary radios. This work also demonstrates that if the secondary user always uses the first channel sensed as free, the best reward for this sequence is obtained. In the solution presented in [6], all channels have the same probability of availability. In this case, the problem of channel sensing order reduces to ordering the channels in the descending order of their achievable rates, as in [8]. All these works have the drawback of relying on a priori knowledge of channel's achievable rates and/or availability probabilities. Furthermore, those solutions are difficult to embed in cognitive radios since their computational complexity increases with the number of channels.

In this work, we propose a low complexity solution, based

This research was supported by FAPERJ, CAPES, CNPq and FINEP.

978-1-61284-928-7/11/$26.00 ©2011 IEEE

on a reinforcement learning machine, that follows the optimal stopping concepts and associated rewards. This solution assumes no prior knowledge of the moments of the random variables that represent the channel's availability and achievable rates. Further, it can dynamically adapt to variations of these moments. It also has low computational complexity, which makes it attractive to be embedded in cognitive radios. The proposed solution is evaluated and compared to the optimal an to other simple channel ordering solutions. The results show that our solution is at most 5% worse than the optimal sequence and it is far superior to the other solutions in most scenarios.

In the remainder of this paper, the following section describes the system model. Section III provides the basic concepts of reinforcement learning and presents our proposal. In Section IV, we describe the simulation environment and show the obtained results. Finally, Section V concludes the paper and lists future works.

II. System Model

In this section, we describe the system model, which is similar to the model presented in [7] which serve as basis for the design and implementation of our proposal. This model allows determining the optimal sequence for channel sensing through the application of optimal stopping theory. In this case, the goal is to decide when to stop sensing new channels with the purpose of maximizing the obtained reward. Therefore, this theory allows defining the stopping rule that maximizes the reward. Moreover, because the number of channels is finite, and it equals to N, the method of backward induction can be applied to find the expected reward of a sequence of N channels.

Consider a secondary user and a finite number of channels, N. This user operates based on time slots, i.e. time is divided into slots of duration T. In each slot, each channel is free of primary radios activity with a probability p_i. We assume that the state of a channel in a slot, free or busy, is independent of its previous state and the state of other channels. We also consider that the signal to noise ratio (SNR) obtained in a channel varies randomly for each slot due to fading effects. The SNR random variable is assumed to be i.i.d. among different slots and channels, and it follows an arbitrary distribution. If the secondary user decides to transmit on a channel sensed as free, c_i, the transmission rate that can be obtained is a function of the instantaneous SNR on that channel. This function, $F(SNR_i)$, is a monotonically increasing function that maps the SNR of the channel c_i in the obtained transmission rate.

Before deciding to use a channel in a particular slot, the secondary user must perform the sensing of this channel with the purpose of determining whether there is activity of primary users. In the model, we assume that the sensing process is error free. Since there is no prior knowledge about the status of the channels, the secondary user performs a sequential sensing of the N channels, following a predetermined order, $\{o_1, o_2, ..., o_N\}$. The efficiency of a given sensing order relates to the time spent sensing the channels and the transmission rate that the user obtains in the chosen channel.

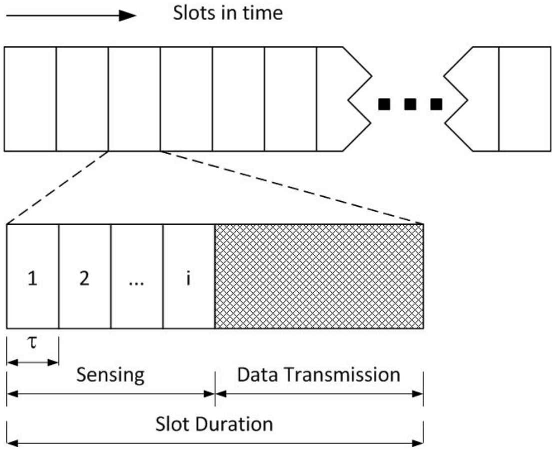


Fig. 1. Process of channels sensing in a slot.

Figure 1 exemplifies the activity of a secondary user in a slot, which has two phases: a sensing phase, and a data transmission phase. The value τ corresponds to the time required for sensing each channel. During the sensing phase, if the channel c_i is sensed as busy, the secondary user performs channel sensing in c_{i+1}, which is the next channel in the used sensing order. However, if the channel c_i is sensed as free, the effective transmission rate obtained by the secondary user in this channel for the remainder of the slot is given by $e_i \times F(SNR_i)$, where e_i is the effectiveness of transmission, calculated by the formula $e_i = \frac{T-i\tau}{T}$. Thus, the reward in the use of each channel in a sequence is given by:

$$r_i = \begin{cases} e_i F(SNR_i) & if \quad e_i F(SNR_i) > R_{i+1} \\ R_{i+1} & \text{otherwise} \end{cases} \tag{1}$$

where R_{i+1} for $i \leq N-1$ is the expected reward if the user decides to proceed in sensing. The expected reward is given by:

$$R_{i+1} = \begin{cases} p_{i+1}E[r_{i+1}] + (1-p_{i+1})R_{i+2} & if \quad i < N-1 \\ p_{i+1}E[r_{i+1}] & if \quad i = N-1 \end{cases} \tag{2}$$

Note that the set of expected rewards $\{R_1, R_2, ..., R_n\}$ can be obtained recursively from R_N through Eq. 1 and 2. Therefore, R_1 is the reward expected by the secondary user using a sequence of N channels. In general, R_i is the expected value of the reward in the use of a partial sequence of channels $(o_i, o_{i+1}, ..., o_N)$. Thus, the use of a channel sensed as free is profitable if the reward, r_i, in using the channel is greater than the expected reward from the remainder of the sequence, R_{i+1}. Otherwise, the secondary user must proceed sensing the next channel of the sequence, and it cannot return to the previous channels (recall).

III. Dynamic Sensing Order Using Reinforcement Learning

The proposed mechanism uses reinforcement learning to dynamically determine the sensing order used in each slot. One advantage of a mechanism based on reinforcement learning is that it requires no prior knowledge about the channel's availability probabilities, or the estimated quality of each channel through its average SNRs. Another key feature of the proposal relates to its adaptability to changes in channels characteristics, because it is able to learn from its past actions. Therefore, the mechanism gains immunity to possible changes in the channel's availabilities, which may occur due to changes in activity patterns of primary radios, and due to possible changes in channels qualities (average SNRs), which may occur due to mobility and large scale fading effects.

In the following subsections, we present the basics of reinforcement learning theory and our proposed mechanism for searching for the optimal sensing order.

A. Reinforcement Learning

Reinforcement learning [11] is a sort of learning machine concerned with how an agent choose its actions, according to cause and effect information obtained from the environment. Briefly, in this method, a given agent, which inspects a given state, performs an action that provides a reward. Based on the collected reward, the agent learns the quality of the chosen action. The problem is then to choose actions that maximize the total rewards obtained by the agent.

The *reinforcement learning method* adopts a simple approach to minimize the complexity of the model, leading to a low computational cost [12]. In contrast, this method may present a slow convergence. Among the many existing reinforcement learning techniques [11], we have adopted the *Q-learning* technique [13] due to its simplicity, and for that reason, it is described below in more detail.

The *Q-learning* method is an online algorithm, which determines the best action at each moment without prior knowledge of the environment. The basic *Q-learning* model consists of:

- decision epochs, which represent the moments of execution of the algorithm, denoted by $t \in T$, $T = \{1, 2, ...\}$;
- a set of states, which represent the modeled problem and are denoted by $s \in S$;
- a set of actions, which represent possible decisions that lead to new states, denoted by $a \in A$;
- rules that determine the reward of an action in a given state, denoted by $r_t(s, a)$;
- transitioning rules between states.

Each agent maintains a *Q-table*, which is a $|S| \times |A|$ matrix, where the rows represent the states and the columns indicate the actions. The elements of this matrix are the *Q-values*, $Q_t(s, a)$, which are updated by using the value of the collected reward, $r_t(s, a)$, whenever the agent takes the action a in state s.

The *Q-value* estimates the level of reward for the state-action pair. Therefore, changes in *Q-values* lead to changes in the decisions on what actions should be taken by agents. At every moment of decision t, the agent observes its current state (row) and chooses an action (column) in its *Q-table*. After the execution of an action, the agent receives a reward $r_t(s, a)$ that is relative to this performed action.

By using the received reward, the agent updates its respective entry on the *Q-table* at time $t + 1$ as:

$$Q_{t+1}(s, a) =$$

$$Q_t(s, a) + \alpha[r_t(s, a) + \gamma max_a Q_t(s_{t+1}, a) - Q_t(s, a)] \tag{3}$$

In Eq. 3, α is known as the *learning parameter*, and γ as the *discount factor*. Higher α values indicate that the agent gives more importance to the recent experiences than to the history. Higher γ values indicate that the agent values more the future reward instead of the immediate reward [14]. The *Q-learning* starts with the *Q-table* filled with zeros, and at each decision epoch, the agent selects an action based on an exploration strategy. A strategy commonly adopted is the $\varepsilon - greedy$ strategy [11], where the agent uses the probability ε to decide between *exploitation* of the *Q-table* or *exploration* of random states. Since ε is usually small, in most cases, the agent greedily selects the action that satisfies $max_a Q(s, a)$, i.e. the action representing the best the agent thinks it can do from state s [15]. However, occasionally, with probability ε, the agent selects a random action. This strategy intends to make agents to experiment all possible actions and its effects [16].

B. Proposal

One of the biggest challenges faced in the use of reinforcement learning at the problem of finding the optimal sensing order was the **states** and **actions** modelling. A careless modelling may lead to many states and actions, which would slow down the convergence of exploration. At our model, we define the state as the ordered pair (o_k, c_i), where o_k is the current position at the sensing order, and c_i is the channel that is sensed at that position. The possible actions of a secondary user at the state (o_k, c_i) correspond to the possible channels that could be sensed at the next position of the sensing order, o_{k+1}. According to that model, the *Q-table* will be a matrix of dimensions $N^2 \times N$ $(states \times actions)$. Note that at this model there is not a unique objective state to be reached by the reinforcement learning; instead, the model leads to a sequence of actions that maximize the immediate reward and create a dynamic sensing order.

At the moment of choosing an action and updating the *Q-table*, there are some import constraints that need to be considered. First, an action taken at any state $(o_k, *)$, with $1 \leq k \leq (N - 1)$, always leads to a state where the position at the sensing order is o_{k+1}. At the states $(o_N, *)$, which represent the last position in the sensing order, the actions indicate the first channel that will be sensed at the next slot and lead to a state $(o_1, *)$. When the secondary user decides to use a channel c_i and stops sensing at the position o_k, the sensing process at that slot finishes. At this case, the first channel that will be sensed at the next slot is determined by the best action at the state (o_N, c_i). Another constraint prevents the secondary user to return to a previously sensed channel (recall). In order to avoid recall, the secondary user needs to store the channels already sensed on the current slot. Therefore, before taking an action, the secondary user must eliminate the sensed channels from the available actions.

The **reward** obtained at each state is other important part of our model. The reinforcement learning uses this reward to update the *Q-table*, as described in Section II. When the channel c_i is sensed as free at the position o_k, the obtained reward, r_t, equals to the effective transmission rate at that channel, $e_k \times F(SNR_i)$. When the channel is sensed as busy, the *Q-value* representing that action needs to be decreased. For that reason, the model uses the parameter δ, which assumes values at the range $[0, 1]$ and multiplies the current *Q-value* of that action. Therefore, the model guarantees that an action that leads to a busy channel always has its corresponding *Q-value* reduced. By adopting this approach, the *Q-value* represents not only the effective transmission rate, but also the channel's availability. In summary, the *Q-table* updating process can be represented by the following:

$$Q_{t+1}(s,a) = \begin{cases} (1-\alpha) \times Q_t(s,a) + \alpha \times r_t(s,a) & if \text{ channel is free} \\ \delta \times Q_t(s,a) & if \text{ channel is busy} \end{cases} \quad (4)$$

The proposed mechanism is described in details in Algorithm 1. At the beginning, all state-action pairs at the *Q-table* are filled with zeros. Afterwards, the mechanism enters the learning phase, which is repeated during its entire period of operation. At this phase, it decides between exploration, where it chooses a random action, and exploitation, where it chooses the best action based on the *Q-table*. After executing the action, the mechanism is able to calculate the reward and to update the corresponding *Q-value*.

An important feature of our proposal concerns the utilization of the free channels. According to the model presented in Section II, the optimal stopping rule requires the comparison between the instant reward and the expected reward for the remainder of the sequence. It indicates that may not always be a good choice to use the first channel sensed as free. In a similar fashion, our proposal uses a stopping rule that consists in comparing the current reward, r_t, to the best *Q-value* among the possible actions available at that state. This way, the proposal can estimate if the current reward is greater than the expected reward of the best action available. Note that, even when the free channel is not used, the *Q-value* corresponding to that action is updated.

IV. Numerical Results

To evaluate the behavior of the reinforcement learning mechanism in solving the problem of sensing order, we have built a discrete events simulator using the Tcl language [17].

```
/* initializes Q-table */
foreach $s \in S,\ a \in A$ do
    Q(s,a) = 0;
while (1) do
    /* learning phase */
    draws a random number $x$ between 0 and 1;
    if ($x < \varepsilon$) then
        /* exploration */
        selects an action $a$ randomly;
    else
        /* exploitation */
        selects the action $a$ which possess the higher
        Q-value for the current state $s$;
    if (free channel) then
        /* channel $c_a$ from action $a$ */
        calculate reward $r_t(s,a)$;
        $Q_{t+1}(s,a) \leftarrow (1-\alpha) \times Q_t(s,a) + \alpha \times r_t(s,a)$;
        if $r_t(s,a) > maxQ(s',a')$ then
            /* use channel $c_a$ */
            slot finishes ;
        else
            /* do not use channel $c_a$ */
            continue sensing ;
    else
        /* channel $c_a$ is busy */
        $Q_{t+1}(s,a) \leftarrow \delta \times Q_t(s,a)$;
        continue sensing ;
    $s_t = s_{t+1}$;
```

Algorithm 1: Proposed mechanism based on reinforcement learning.

Through this simulator, we evaluate the performance of the following sensing orders: the dynamic sequence provided by our proposal (`RL`), the optimal sequence obtained by brute force (`OPTIMAL`), the sequence in descending order of the average achievable rates (`CAP`) [8], the sequence in descending order of channel's availability probabilities (`PROB`), the sequence in descending order of the product between the average achievable rates and the availability probabilities (`PROBxCAP`), and the sequence in random order of channels (`RANDOM`). It is noteworthy that all the above sequences, except in the case of `RL`, are static, i.e. they do not change during simulation. In the case of `RL`, the sequence varies during simulation due to the adaptations performed by the reinforcement learning mechanism. Moreover, all these sequences, except `RL` and `RANDOM`, assume a priori knowledge of the average achievable rates and/or channel's availability probabilities.

A. Simulation Model

At the beginning of each experiment, the simulator chooses randomly the average achievable rate, hereafter called capacity, and the availability probability of each channel c_i, with $i \in \{1, ..., N\}$. The average capacities of the channels follow an uniform distribution within the range $[FCH \times MAXCAP, MAXCAP]$, where FCH is a *factor of channel homogeneity* and $MAXCAP$ is the maximum average capacity of the channels. The parameter FCH assumes values in the interval $[0, 1]$. A high FCH value makes the average channel's capacities more homogeneous and closer to the $MAXCAP$. The channel's availability probabilities follow a uniform distribution within the range $[0, 1]$.

During an experiment, at each slot T, the channels are set randomly as free or busy according to their availability probability using either a uniform distribution or a exponentially distributed on-off model. The instantaneous capacity of each channel is also chosen randomly at each slot, by following a uniform distribution within the range $[MEANCAP \times (1 - FEV/2), MEANCAP \times (1 + FEV/2)]$. At this case, FEV is the *factor of environment variability* and $MEANCAP$ is the mean instantaneous capacity of each channel. With high FEV values, the instantaneous achievable rate on each channel presents high variations. The parameter FEV assumes values in the interval $[1, 2]$.

At each slot, the simulator computes the reward obtained by each implemented sequence using the same channel states and instantaneous capacities, i.e. under the same condition for the sake of fairness in the comparisons. The reward at each slot corresponds to the effective transmission rate (Section II). Each simulation run comprises the execution of X slots. At the end of each run, the simulator computes the average reward obtained by each sequence in all X slots.

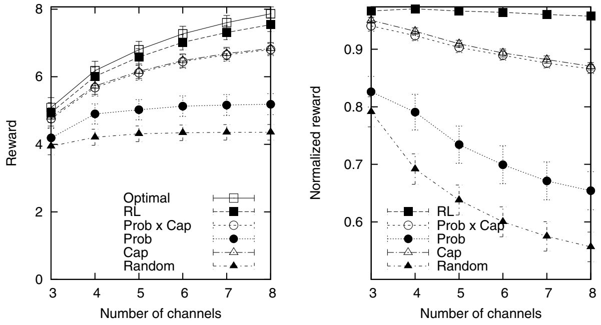
B. Results

We have performed simulations with 50.000 slots each and a number of channels varying from 3 to 8. We choose the values of the parameters that were most representative within its validity interval after a round of tests. Thus the exploration factor ε is set to 0.7 during the first slots of each simulation, which corresponds to 20% of the total number of slots. This ε changes to 0.1 after that exploration period until the end of the experiment. The parameter α of RL was set to 0.1. The size of the slot T is a integer multiple of the time required for sensing each channel (τ). We have conducted 200 simulation runs for each set of parameters. In all graphs, we present the average rewards obtained at each simulation run, with error bars corresponding to confidence intervals of 95%.

In the first set of simulations, channels availability is randomly chosen using a uniform distribution, i.e. they are set as busy or free at each slot according to their availability probability. In the other group of simulations, the channel state is set according to an exponentially distributed on-off model. In the simulation of this model, the channel stays in the OFF-state (i.e. busy) for a exponentially distributed time with a mean of t_{OFF}. Thus, the t_{ON} can be obtained by $t_{ON} = \frac{(1-u) \times t_{OFF}}{u}$, where u is the *channel utilization* by primary users, i.e. the channel probability of being unavailable. The results obtained for these two availability models are presented in the next two subsections.

Uniformly Distributed Channel Availability

Figure 2 presents the results as a function of the number of channels. In these simulations, both $MAXCAP$ and the slot size are set to 10, parameter δ is set to 0.95, and the FCH and FEV parameters are set to 0.1 and 2.0, respectively.



(a) absolute rewards

(b) normalized rewards

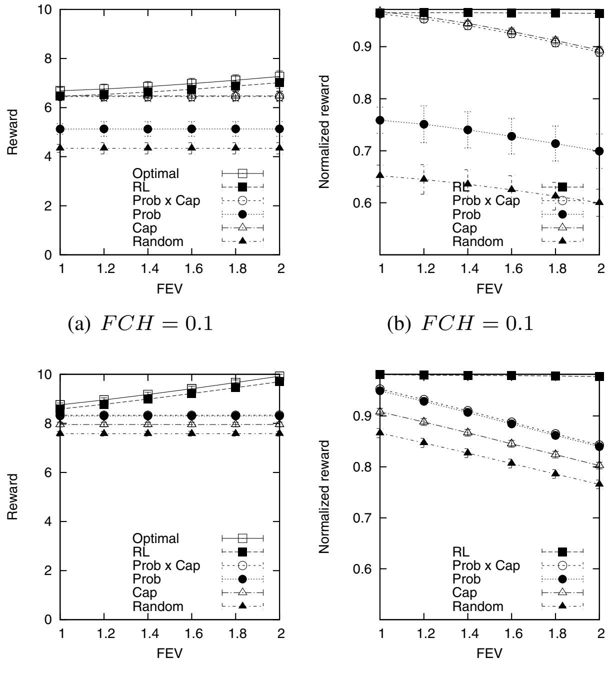
Fig. 2. Results for $FCH = 0.1$, $FEV = 2$, and $\delta = 0.95$.

Analyzing the absolute rewards presented in Figure 2(a), one can see that as the number of channels increases, the average reward for all simulated sequences also increases. This behavior occurs because with a higher number of channels the likelihood of channels with higher capacity and availability increases. Also for this reason, PROB, CAP, PROBxCAP, RL and OPTIMAL solutions, which are aware of the channel's achievable capacities and availability probabilities, achieve better performance gains than the RANDOM sequence.

Figure 2(b) presents obtained average rewards normalized to the OPTIMAL sequence performance. This performance comparison shows that our proposal, RL, is the one that achieves results closer to the OPTIMAL sequence. The performance of other sequences is worst because none of them use a stopping-rule based on a expected reward for remaining sequence, i.e. they stop sensing as soon as the first free channel is found. Thus, the RL, using past experiences stored in the *Q-table*, can determine efficiently whether it is advantageous to use a particular channel sensed as free. Another interesting observation about the curves in Figure 2(b) is that the performance of the PROB sequence is lower than the performance of CAP and PROBxCAP sequences. This indicates that the difference between the average channels capacities ($MEANCAP$) in this scenario is more important than the difference between their availability probabilities. Thus, it is better to sort channels in the decreasing order of their average capacities since this increases the likelihood of the first free channel to have a higher channel capacity.

Figures 3 and 4 show the absolute and normalized rewards as a function of FEV and FCH, respectively. The FEV parameter modifies the variability of the instantaneous channels capacity at each slot, which is representative of the dynamic variation of SNR. The PROB, CAP, PROBxCAP and

RANDOM are invariant with respect to this parameter due to these sequencesuse the average values . Thus, observing Figures 3(a) and 3(c), which considers the absolute reward, and Figures 3(b) and 3(b), which consider the normalized reward for all values of FEV, the sequences used are always the same, and the rewards tend to a constant average value. For OPTIMAL and RL sequences, the increase in the absolute reward occurs because these strategies only use a free channel if the instantaneous reward is greater than the expected reward computed for the remaining of the sequence. Thus, these strategies tend to use free channels with larger instantaneous capacity due to their greater variability. Other strategies always use a free channel, independent of their instantaneous capacity.



(a) $FCH = 0.1$

(b) $FCH = 0.1$

(c) $FCH = 0.9$

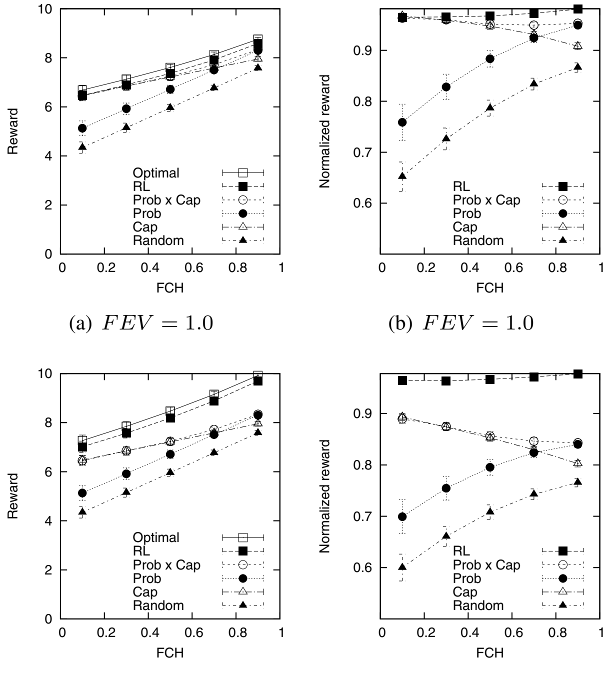
(d) $FCH = 0.9$

Fig. 3. Results for 6 channels, slot size = 10, channel capacity = 10.

FCH modifies the homogeneity of the channels with respect to their average capacities. By increasing the value of this parameter, the average capacities are closer to $MAXCAP$, increasing the average reward with the increase of FCH for all mechanisms, as shown in Figure 4. However, the performance of the sequences sorted by channel capacity, i.e. CAP and PROBxCAP, do not grow in the same proportion. The explanation for this lies in the fact that when the channels are more homogeneous, the lesser is the weight of the capacity on the choice of the sequence. Therefore, the PROB sequence improves in performance as channels become more homogeneous.

On-Off Channel Availability Model

In the simulations of this availability model, the channel utilization values at each slot is varied according to a uniformly distributed variable within the interval [0.1, 0.9] and the t_{OFF} is defined in number of slots.



(a) $FEV = 1.0$

(b) $FEV = 1.0$

(c) $FEV = 2.0$

(d) $FEV = 2.0$

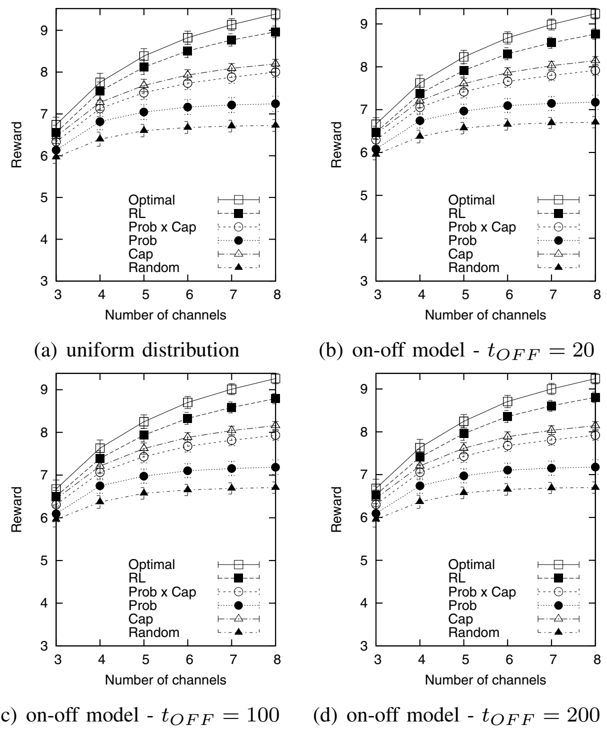
Fig. 4. Results for 6 channels, slot size = 10, channel capacity = 10.

In the first simulations, we evaluate the impact of using an on-off model to characterize the activity of the primary users. For these simulations the t_{OFF} assumes values equals to 20, 100 and 200. Also, the δ, FCH and FEV parameters are set to 0.95, 0.1 and 1.5, respectively. The absolute rewards are shown in Figure 5.

By comparing the results in Figures 5(b), 5(c) and 5(d) to the ones in Figure 5(a), one can observe that the performance of the RL ís the most impacted with the increase of t_{OFF}. However, the performance degradation is small, which demonstrates the robustness of the RL against different channel availability processes without any prior knowledge.

In the next round of simulations, we evaluated the influence of δ parameter (subsection III-B). This parameter dictates the penalization incurred by the choice of a busy channel in the sequence, leading to a reduced *Q-value* for that action. The t_{OFF} equals to 100 and FCH and FEV parameters assume values equal to 0.1 and 1.5, respectively. Figure 6 shows the absolute reward for different values of δ. The results show that the larger is the value of δ, the better is the performance of the RL. This occurs because in the on-off model, the unavailability of a channel occurs in bursts of slots. This way, the reward in the use of a busy channel can be very reduced, leading the RL to change of sequence more often.

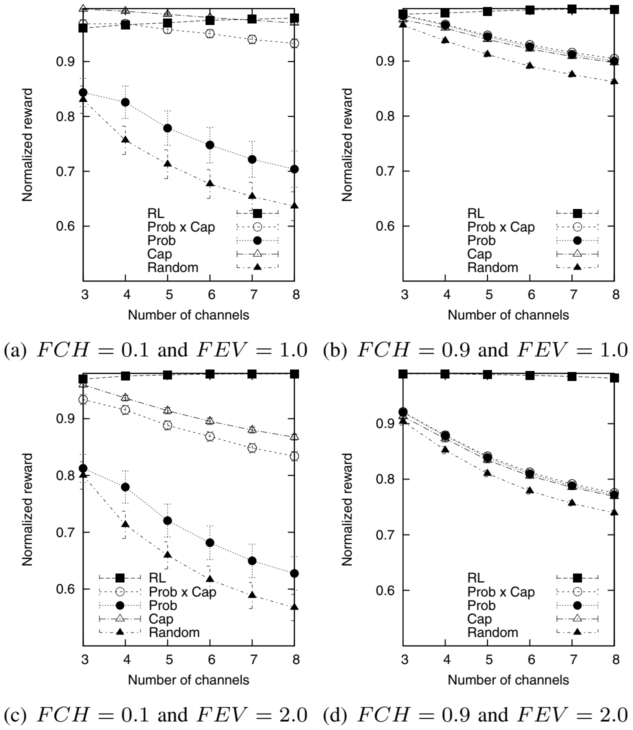
Finally, the last set of simulations evaluate the influence of the FCH and FEV parameters when the on-off channel availability model is used. The t_{OFF} equals to 100, and δ assumes the value of 0.99. The normalized rewards results are



(a) uniform distribution (b) on-off model - $t_{OFF} = 20$

(c) on-off model - $t_{OFF} = 100$ (d) on-off model - $t_{OFF} = 200$

Fig. 5. Uniform distribution vs on-off model.



(a) $FCH = 0.1$ and $FEV = 1.0$ (b) $FCH = 0.9$ and $FEV = 1.0$

(c) $FCH = 0.1$ and $FEV = 2.0$ (d) $FCH = 0.9$ and $FEV = 2.0$

Fig. 7. Influence of the channel heterogeneity and variability.

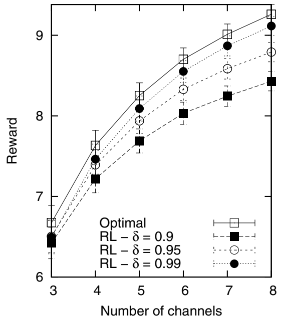


Fig. 6. Influence of the δ parameter.

shown in Figure 7. In the scenario with the larger heterogeneity and smaller variability of the channels ($FCH = 0.1$ and $FEV = 1.0$), the RL presents the worst performance relative to the OPTIMAL for a small number of channels. This occurs due to the dynamic nature of the RL. In these scenarios, any temporary change from the optimal order causes a significant performance degradation. In contrast, the RL achieves a performance very close to the OPTIMAL in the other scenarios.

V. Conclusions

The spectrum sensing is a critical task for the opportunistic use of licensed channels of the spectrum. Specifically, in scenarios where users have only a single transceiver, which has to sense one channel at a time to detect opportunities for use. In these cases, the channel sensing order can have great impact on performance. The optimal stopping theory can be used to model the problem and determine the optimal sensing order. However, this theory assumes prior knowledge about the availability probabilities and expected average capacity of each channel.

We propose a low complexity solution that uses the reinforcement learning machine. This solution requires no prior knowledge of the availability probability and expected average capacity of each channel, and can dynamically adjust itself to variations in these characteristics. Moreover, this solution has low computational complexity, which makes it attractive to be embedded in cognitive radios. For evaluate the performance of the proposed mechanism, we developed a simulator that emulates a secondary user that follows arbitrary sensing sequences. The simulation results show that the proposed mechanism achieves performance close to the optimal sensing order and superior to other pre-established sequences.

As future works, we intend to extend the evaluation for the case of multiple users and for scenarios where the availability probability and the average channel's capacities vary during the simulations. In addition, it would be interesting to evaluate how sensing errors can affect the proposed mechanism.

References

[1] M. A. McHenry, "NSF Spectrum Occupancy Measurements Project Summary," Shared Spectrum Company report, Tech. Rep., 2005.

[2] FCC, "FCC-03-322 - NOTICE OF PROPOSED RULE MAKING AND ORDER," Federal Communications Commission, Tech. Rep., 30 Dec. 2003.

[3] J. Mitola III and G. Q. Maguire Jr., "Cognitive Radio: Making Software Radio more Personal," *IEEE Pers. Communications*, vol. 6, no. 4, pp. 13–18, Aug. 1999.

[4] S. Guha, K. Munagala, and S. Sarkar, "Approximation Schemes for Information Acquisition and Exploitation in Multichannel Wireless Networks," in *44th. Allerton Conference*, 2006.

[5] H. Kim and K. G. Shin, “Fast Discovery of Spectrum Opportunities in Cognitive Radio Networks,” in *IEEE DySPAN*, 2008.

[6] J. Jia, Q. Zhang, and X. Shen, “HC-MAC: a Hardware Constrained Cognitive MAC for Efficient Spectrum Management,” *IEEE JSAC*, Jan. 2008.

[7] H. Jiang, L. Lai, R. Fan, and H. V. Poor, “Optimal Selection of Channel Sensing Order in Cognitive Radio,” *IEEE Transactions in Wireless Communications*, Jan. 2009.

[8] Ho Ting Cheng and Weihua Zhuang, “Simple Channel Sensing Order in Cognitive Radio Networks,” *IEEE Journal on Selected Areas in Communications*, Apr. 2011.

[9] Y. S. Chow, H. Robbins, and D. Siegmund, *Great Expectations: The Theory of Optimal Stopping*. Houghton Mifflin Company, 1971.

[10] Han Han, Jin-long Wang, Qi-hui Wu, and Yu-zhen Huang, “Optimal Wideband Spectrum Sensing Order Based on Decision-making Tree in Cognitive Radio,” *International Conference on Wireless Communications and Signal Processing (WCSP)*, 2010.

[11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: an Introduction*. MIT Press, 1998.

[12] Kok-Lim Alvin Yau, Peter Komisarczuk, and Paul D. Teal, “Applications of Reinforcement Learning to Cognitive Radio Networks,” in *IEEE International Conference in Communications (ICC)*, July 2010.

[13] C. J. C. H. Watkins, “Learning from Delayed Rewards,” Ph.D. dissertation, King’s College, May 1989.

[14] Kok-Lim Alvin Yau, Peter Komisarczuk, and Paul D. Teal, “Enhancing network performance in distributed cognitive radio networks using single-agent and multi-agent reinforcement learning,” in *IEEE Conference on Local Computer Networks (LCN)*, Oct. 2010.

[15] C. J. C. H. Watkins and P. Dayan, “Q-learning,” *Machine Learning*, no. 8, pp. 279–292, 1992.

[16] Jelle R. Kok and Nikos Vlassis, “Collaborative Multiagent Reinforcement Learning by Payoff Propagation,” *J. Mach. Learn. Res.*, vol. 7, pp. 1789–1828, December 2006. [Online]. Available: http://portal.acm.org/citation.cfm?id=1248547.1248612

[17] John Ousterhout, “Tcl - Tool Command Language,” 1988, http://www.stanford.edu/ ouster/cgi-bin/tclHistory.php.