Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 00 (2015) 000–000

Procedia
Computer Science

www.elsevier.com/locate/procedia

The International Conference on Advanced Wireless, Information, and Communication Technologies (AWICT 2015)

Cognitive Radio Jamming Mitigation using Markov Decision Process and Reinforcement Learning

Feten Slimeni[a,*], Bart Scheers[b], Zied Chtourou[a], Vincent Le Nir[b], Rabah Attia[c]

[a] *VRIT Lab - Military Academy of Tunisia, Nabeul 8000, Tunisia*
[b] *CISS Departement - Royal Military Academy (RMA), Brussels 1000, Belgium*
[c] *SERCOM Lab - EPT University of Carthage, Marsa 2078, Tunisia*

Abstract

The Cognitive radio technology is a promising solution to the imbalance between scarcity and under utilization of the spectrum. However, this technology is susceptible to both classical and advanced jamming attacks which can prevent it from the efficient exploitation of the free frequency bands. In this paper, we explain how a cognitive radio can exploit its ability of dynamic spectrum access and its learning capabilities to avoid jammed channels. We start by the definition of jamming attacks in cognitive radio networks and we give a review of its potential countermeasures. Then, we model the cognitive radio behavior in the suspicious environment as a markov decision process. To solve this optimization problem, we implement the Q-learning algorithm in order to learn the jammer strategy and to pro-actively avoid jammed channels. We present the limits of this algorithm in cognitive radio context and we propose a modified version to speed up learning a safe strategy. The effectiveness of this modified algorithm is evaluated by simulations and compared to the original Q-learning algorithm.

Keywords: Cognitive radio network, jamming attack, Q-learning algorithm

1. Introduction

Cognitive Radio (CR) technology is recognized as an intelligent radio that is able of learning and reconfigurability in order to automatically detect available channels in wireless spectrum and perform a real time adaptation to the environment modifications [1,2]. Its ability of dynamic spectrum management makes it a promising solution to overcome the problems of scarcity and inefficient utilization of the radio spectrum, but makes it more susceptible to be jammed. Furthermore, cognitive radio networks (CRNs) are characterized by dynamic spectrum access (DSA) and by mainly distributed architectures which make it difficult to implement effective jamming countermeasures.The jammers can be classified according to the following criteria:

- Spot/Sweep/Barrage jamming
  Spot jamming consists in attacking a specific frequency, while a sweep jammer will sweep across an available frequency band. A barrage jammer will jam a range of frequencies at once.
- Single/Collaborative jamming

The jamming attack can be done by a single jammer or in a coordinated way between several jammers to gain more knowledge about the network and to efficiently reduce the throughput of the cognitive users.

- Constant/Random jamming
  The jammer can either send jamming signals continuously on a specific channel or alternate between jamming and sleeping.
- Deceptive/Reactive jamming
  A deceptive jammer continuously transmits signals in order to imitate a legitimate or primary user. A reactive jammer transmits only when it detects busy channel to cause collisions.

In this paper, we start by a review of the common anti-jamming techniques in CRNs. Then, we model the CR behavior in the suspicious environment as a markov decision process (MDP). To solve this optimization problem, we explain in section 4 how the Q-learning can be used to learn the jamming strategy and to pro-actively avoid the jammed frequencies. However, using the standard version of this algorithm present several limits in CRNs, so we present a modified version making the learning process safer and faster. We evaluate the effectiveness of this modified algorithm in the presence of different jamming strategies. The simulation results are compared to the original Q-learning algorithm applied to the same scenarios.

2. Review of CR jamming attack countermeasures

The jamming attack has been widely exploited as strategic maneuver in military wireless communications. This problem has been intensively researched for traditional wireless networks but it is still a challenging issue in CRNs.

We present in this section an overview of the proposed anti-jamming techniques in CRNs. We start by the traditional anti-jamming solutions used in wireless networks, which consist in spread spectrum techniques by the use of either frequency hopping (FH) or direct-sequence spread spectrum (DS-SS) methods[3]. These solutions can be enhanced to mitigate the jamming attack in CRNs. Then, we present another class of anti-jamming techniques which aim to correct the already jammed data during transmission. There are solutions to deceive the jammers instead of escaping from it or repairing its effect on the transmitted data. And finally, we present how game theory is used in related papers to model the jamming attack in CRNs and to find optimal anti-jamming strategies.

2.1. Frequency hopping

The CR is characterized by its ability of dynamic spectrum access to use the spectrum in opportunistic way. This ability can be exploited to overcome jamming attacks since the CR can change its operating frequency to avoid the jammers. However, the exploitation of frequency hopping in CRN anti-jamming approaches present a trade-off between the resource consumption every time to change the jammed frequency and the jamming impact if the CR still using the same frequency even jammed.

Recently, diverse CRN frequency hopping defense strategies, were analyzed.[4] presented proactive or impetuous hopping (selecting a new set of frequencies at every slot, irrespective of the jamming) and reactive or conservative hopping (unjammed users keep the same frequencies for the next slot, while the jammed users choose a set of new unused frequencies that exclude the jammed ones). The authors proposed a multi-tier proxy based cooperative defense strategy, in which users form tiers to exploit the temporal and spatial diversity to avoid jamming.

2.2. Direct-sequence spread spectrum (DS-SS)

This spread spectrum technique consists in spreading the signal over several pieces of non-overlapping channels. It can be exploited as an anti-jamming technique because the jammer will have to choose either to jam a large number of channels with negligible jamming effect in each one or to jam only few channels with important effect. The authors in[5], proposed an uncoordinated spread spectrum technique that enables anti-jamming broadcast communication without predefined shared secrets. They aimed to improve the common spread spectrum which depends on secret pairwise or group keys shared between the sender and the receivers before the communication, to adapt it for critical applications such as emergency alert broadcasts and military communications.

A random channel sharing was proposed in[6], for broadcast CR communication to mitigate the insider jamming attack which resist to spread spectrum techniques. The proposed idea is to organize receivers into multiple broadcast classified trusted/suspicious groups and use different channels for different groups. This ensures that a compromised receiver can only affect the members of the group it has been assigned to.

2.3. Coding anti-jamming techniques

In addition to approaches trying to evade the jammers, the CR can use coding techniques to mitigate the effect of the jamming attack on the transmitted signal. For example, the authors in[7] combine random linear network coding with random channel hopping sequences to overcome the jamming effect on the transmitted control packets. Their proposed algorithm is called jamming evasive network coding neighbor discovery algorithm (JENNA). Another coding approach is presented in[8], it consists in a hybrid forward error correction (FEC) code to mitigate the jamming impact on the transmitted data. The code is a concatenation of the raptor code to recover data loss due to jamming, and the secure hash algorithm (SHA-2) to verify the integrity of the received data.

2.4. Anti-jamming technique by deceiving the jammer

Instead of using coding technique to repair the already jammed data, the concept of honeynode has been shown in[9] to be effective in deceiving jammers about the transmitting nodes. In this reference, a single honeynode is dynamically selected for each transmitting period, to act as a normal transmitting CR in order to attract the jammer to a specific channel.

2.5. Game theory

The behaviors of the CR, doing transitions between available frequencies, and the jammer, trying to prevent it from efficiently utilizing the spectrum, can be modeled using game theory. In this context, several works have been using game models and learning algorithms to find optimal anti-jamming strategy for the CR. For example in[10], the authors model the CRN jamming scenario as zero-sum game because of the opposite CR and jammer objectives. Furthermore, they implement the minimax-Q learning algorithm to find the optimal defense policy. Recently, the problem is formulated as a non-zero-sum game in[11], by taking into account different hopping and transmission costs, as well as diverse reward factors for both the transmitter and the jammer side. Authors make use of fictitious play learning algorithm to learn optimal defense strategy. Closed-form expressions to the jamming probabilities and the throughput of the CRN under various jamming attack models, was determined in[12] using the concept of Markov chain.

In this paper, we will consider a fixed jamming strategy which means that the only player is the CR who tries to find the better strategy against a jammer with fixed behavior.

3. Using MDP to model CR jamming attack

Markov decision process (MDP) has been widely exploited as a stochastic tool to model the CR decision making problem in jamming scenarios with fixed strategy, i.e. assuming that the jammer preserves the same tactic.

The MDP is a discrete time stochastic control process. It provides a mathematical framework to model the decision problem faced by an agent to optimize his outcome. The goal of solving the MDP is to find the optimal strategy for the considered agent. In CRN jamming scenario, it means finding the best actions (to hop or to stay) for the CR to avoid the jammed frequency. An MDP is defined by four essential components:

- A finite set of states S.
- A finite set of actions A.
- $P_a(s, s') = Pr(s_{t+1} = s'|s_t = s, a_t = a)$ the transition probability from an old state s to a new state s' when taking action a.
- $R_a(s, s')$ the immediate reward after transition to state s' from state s when taking action a.

The process is played in a sequence of stages (timesteps). At every stage, the agent is in one state and at the end of that stage he selects an action, then the process moves to a new random state with the corresponding transition probability. The agent receives a payoff, also called reward, which depends on the current state and the taken action. He continues to play stages until finding the optimal policy, which is the mapping from states to actions that maximizes the state values.

Let's define a markov decision process to model the CR's available states and actions, with the consideration of unknown transition probabilities and unknown immediate rewards of the taken actions. We consider a fixed jamming strategy to solve the decision making problem from the side of the CR trying to find an anti-jamming strategy.

Assume there are M available channels for the CR and there is a jammer trying to prevent it from an efficient exploitation of these channels. As a defense strategy, the CR have to choose at every timeslot either to keep transmitting over the same channel or to hop to another one. The challenge is to learn how to escape from jammed channels without scarifying a long training period to learn the jammer's strategy. Lets define the finite set of possible states, the finite set of possible actions at each state and the resultant rewards after taking these actions.

The state of the CR is defined by a pair of parameters: its current operating frequency and the number of successive timeslots staying in this frequency. Therefore, its state at a timeslot i is represented by the pair $s_i = (f_i, k)$, where f_i is its operating frequency at this timeslot i and k is the number of successive timeslots using this frequency. We have opt for mixing spatial and temporal properties in the state space definition to get a Markovian evolution of the environment.

At every state, the CR should choose an action to move to another state, which means that it has to choose its future frequency. Therefore, we define its possible actions as a set of M actions, which are the M available channels: $\{f_1, f_2, ..., f_M\}$. An example of the Q matrix composed by these states and actions is given in Table 1.

Assume the reward is zero $R_a(s, s') = 0$ whenever the new frequency after choosing the action a is not jammed and $R_a(s, s') = -1$ when the CR takes an action a resulting to a jammed frequency. We consider that the jammed state as a failure and a situation that should be avoided.

4. Using Q-learning to counter the jammer

Learning algorithms can be used as a model-free simulation tool for determining the optimal policy π^* without initially knowing the action rewards and the transition probabilities. Autonomous reinforcement learning (RL) is completely based on interactive experience to update the information step by step, and based on this derive an estimate to the optimal policy. The most popular RL method is the Q-learning algorithm. It was exploited in [13,14] to solve the jamming problem in CRNs. In the first paper, the authors start by deriving a frequency hopping defense strategy for the CR using an MDP model under the assumption of perfect knowledge, in terms of transition probabilities and rewards. Further, they propose two learning schemes for CRs to gain knowledge of adversaries to handle cases of imperfect knowledge: maximum likelihood estimation (MLE), and an adapted version of the Q-learning algorithm. However the modified Q-learning algorithm is given without discussion or simulation results. The second paper gives an MDP model of the CRN jamming scenario and proposes a modified Q-learning algorithm to solve it. Again, as in the previous reference no details are given on how to implement the described theoretical anti-jamming scheme.

As first introduced by Watkins in [15] 1989, the Q-learning algorithm is a simple way for agents to learn how to act optimally by successively improving its evaluations of the quality of different actions at every state. The goal is finding a mapping from state/action pairs to Q-values. This result can be represented by a matrix of N_s lines, where N_s is the number of states s, and N_a columns corresponding to possible actions a. The Q-values are calculated in this algorithm by an iterative process; at every timeslot the algorithm measures the feedback rewards of taking an action a in a state s, and updates the corresponding $Q(s, a)$:

$$Q[s,a] \leftarrow Q[s,a] + \alpha \left[R_a(s,s') + \gamma \, max_a Q(s',a) - Q[s,a]\right] \tag{1}$$

which gives:

$$Q[s,a] \leftarrow (1-\alpha)Q[s,a] + \alpha \left[R_a(s,s') + \gamma \, max_a Q(s',a)\right] \tag{2}$$

where $0 < \alpha \leq 1$ is a learning rate that controls how quickly new estimates are blended into old estimates, and γ is the discount factor that controls how much effect future rewards have on the optimal decisions. Small values of γ emphasizing near-term gain and larger values giving significant weight to later rewards.

4.1. The standard version of the Q-learning algorithm

The Q-learning algorithm updates the values of $Q(s, a)$ through many episodes (trials) until convergence to optimal Q^* values; this is known as the training/learning stage of the algorithm. Each episode starts from a random initial state s_1 and consists on a sequence of timeslots during which the agent goes from state to another and updates the corresponding Q value. Each time the agent reaches the goal state, which have to be defined depending on the scenario, the episode ends and he starts a new trial. The convergence to the optimal Q^* matrix requires visiting every state-action pair as many times as needed. In simulation, this problem is known as the exploration issue. Random exploration takes too long to focus on the best actions which leads to a long training period of many episodes. Furthermore, it does not guarantee that all states will be visited enough, as a result the learner would not expect the trained Q function to exactly match the ideal optimal Q^* matrix for the MDP[16].

Two main characteristics of the standard Q-learning algorithm are: (i) it is said to be an asynchronous process since at each timeslot the agent updates a single $Q(s, a)$ value (one matrix cell), corresponding to his current state s (line s) and his action a (column a) taken at this timeslot[17]. (ii) The Q-learning method does not specify what action a the agent should take at each timeslot during the learning period, therefore it is called OFF-policy algorithm allowing arbitrary experimentation until convergence to stationary Q values[18].

The optimal Q^* matrix resulting from the learning period will be exploited by the agent as the best policy. During the exploitation phase, when he is in a state s, he has to take the action corresponding to the maximum value in the matrix line $Q^*(s, :)$. However, an off-line application of this technique seems to be inefficient for the CR, because until the convergence of the Q-learning algorithm other jammers may emerge and legacy spectrum holders (primary users) activity may change. During the training phase of the Q-learning algorithm, the CR can already exploit the communication link, denoted as on-line learning, but it may lose many data packets because of the random learning trials.

As a solution to these challenges, we propose in the next paragraph a modified version of the Q-learning algorithm, and we will denote this version as ON-policy synchronous Q-learning (OPSQ-learning) algorithm.

4.2. The On-policy synchronous Q-learning algorithm

We present in algorithm 1, a modified version of the Q-learning process denoted as the ON-policy synchronous Q-learning (OPSQ-learning), because of the two following modifications: (i) We have replaced the OFF-policy characterizing the standard Q-learning algorithm by an ON-policy, i.e. at each timeslot, the CR follows a greedy strategy by selecting the best action corresponding to $max_a Q(s, a)$ instead of trying random action. (ii) We have exploited the CR ability of doing wideband spectrum sensing, to do synchronous update of M Q-values instead of the asynchronous update of only one cell in the Q matrix, i.e. the CR after going to a next state can, using its wideband sensing capability, detect the frequency of the jammer at that moment and hence do an update of all state-action pairs, corresponding to the possible actions which can be taken from its previous state s (update of all columns of the Q matrix line $Q(s, :)$). Due to the second modification (the synchronous Q-values update), the modified Q-learning algorithm is no longer a model-free technique but it can be seen as a model-based technique, i.e. the CR can learn without actually apply the action.

We have to mention that we are assuming perfect spectrum sensing and full observations for simplicity. But we cite some interesting references dealing with the influence of the radio channel in the estimation of the detected signal. For example,[19] develops and analyzes an adaptive spectrum sensing scheme according to the variation of time-varying channels,[20] studies the cooperative spectrum sensing for a cognitive radio system operating in AWGN, correlated/uncorrelated shadowing, and in channels featuring composite large-scale and small scale fading. Also,[21] provides a comprehensive overview of the propagation channel models that will be used for the design of cognitive radio systems and deals with the time variations of the channel response which determine how often potential interference levels have to be estimated and, thus, how often transmission strategies may have to be adapted.

To evaluate the effectiveness of the proposed solution, we have applied both the standard version of the Q-learning algorithm (characterized by OFF-policy and asynchronous update) and the modified ON-policy synchronous Q-learning algorithm to the described MDP model. Note that in this algorithm, our episode starts from a random frequency, going from one state to another by taking the best action at every timeslot, and ends whenever the CR goes to a jammed frequency. The next section presents the simulation results in the presence of various jamming strategies.

Algorithm 1 Pseudocode of ON-policy synchronous Q-learning

```
Set $\gamma$ and $\epsilon$ values.
Initialize matrix $Q_1$ to zero matrix.
Select a random initial state $s = s_1$
Set n=1, timeslot=1
while n<Nepisodes do
  $Q_{n-1} = Q_n$, $R_a(s, s') = 0 \ \forall \ a,s,s'$
  Calculate the learning coefficient $\alpha = 1/timeslot$
  Select an action $a$ verifying $max_a Q_{n-1}(s, a)$
  Taking $a$, go to the new state $s'$ at frequency $f'$
  Find the new jammed frequency $f_{jam}$ %(due to wideband spectrum sensing)
  Update all $Q_n$ values of the previous state $s$ by doing:
  for $i = 1 : M$ do
    observe the fictive state $s_{tmp}$ of taking fictive action $f_i$
    if $f_i = f_{jam}$ then
      $R_{f_i}(s, s_{tmp}) = -1$
    else
      $R_{f_i}(s, s_{tmp}) = 0$
    end if
    Compute $Q_n(s, f_i) = (1 - \alpha)Q_{n-1}(s, f_i) + \alpha[R_{f_i}(s, s_{tmp}) + \gamma \ max_a Q_{n-1}(s_{tmp}, a)]$
  end for
  if $f' = f_{jam}$ %(end of episode) then
    n=n+1
    timeslot=1
    Select a random initial state $s = s_1$
  else
    $s = s'$
    timeslot=timeslot+1
  end if
  if $(abs(Q_n(s, a) - Q_{n-1}(s, a)) < \epsilon) \ \forall \ s,a$ then
    break
  end if
end while
```

5. Simulation results

We have considered in the simulations four available frequencies ($M = 4$) for the CR. Using Matlab, we have implemented both the standard and the modified versions of the Q-learning algorithm, under sweeping, reactive and pseudo random jamming strategies.

We started by the implementation of the standard version of Q-learning algorithm. We found, by averaging over many simulations, that it takes about one hundred episodes to converge to the matrix Q^*. Then, we have implemented the modified Q-learning version (OPSQ-learning) and we give the results in the following paragraphs. The following figures display the anti-jamming strategy in the exploitation phase, after running the learning algorithm. We are using the red color to indicate the jammed frequencies and the blue color to indicate the CR frequencies for an exploitation period of twenty timeslots.

5.1. Scenario with a sweeping jammer

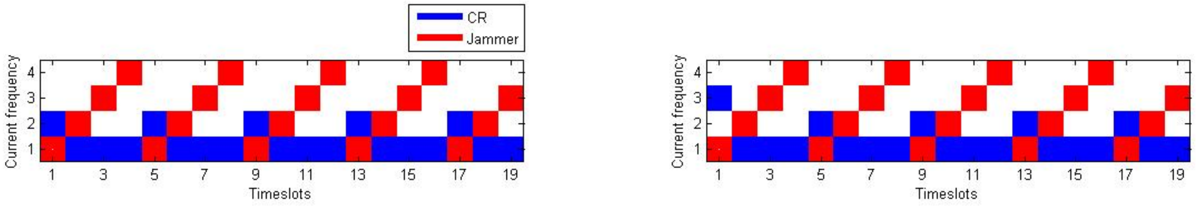
As a first scenario, we consider a jammer sweeping over the available spectrum frequencies by attacking at each timeslot one frequency. The OPSQ-learning algorithm converges after only one or two episodes depending on the initial state. The Q^* matrix is given in Table 1. The strategy given by this resulting Q^* matrix is shown in Fig. 1, when the CR starts as initial random state s_1 from the frequencies f_2 and f_3 respectively.

5.2. Scenario with a reactive jammer

In this scenario, we consider a reactive jammer. We suppose that this jammer needs a duration of two timeslots before jamming the detected frequency, because it has to do the spectrum sensing, then make the decision and finally hop to the detected frequency. The OPSQ-learning algorithm converges in this scenario after four episodes. The CR succeeds to learn that it has to change its operating frequency every two timeslots to escape from the reactive jammer. The learned strategy is given in Fig. 2 when the CR starts respectively from the frequencies f_2 and f_3 as initial state s_1.

Table 1. The Q^* matrix in a sweeping jammer scenario

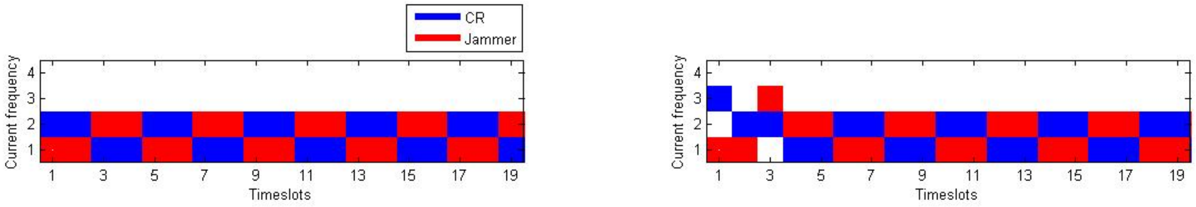
State \ Action	f_1	f_2	f_3	f_4
(f_1,1)	0	0	-0.8356	0
(f_1,2)	0	0	0	-0.6768
(f_1,3)	-0.5770	0	0	0
⋮	⋮	⋮	⋮	⋮
(f_2,1)	0	-0.3822	0	0
⋮	⋮	⋮	⋮	⋮
(f_3,1)	0	-1	0	0
⋮	⋮	⋮	⋮	⋮
(f_4,1)	0	0	0	0
⋮	⋮	⋮	⋮	⋮



(a) $s_1 = (f_2, 1)$

(b) $s_1 = (f_3, 1)$

Fig. 1. Exploitation of the learned policy against a sweeping jammer



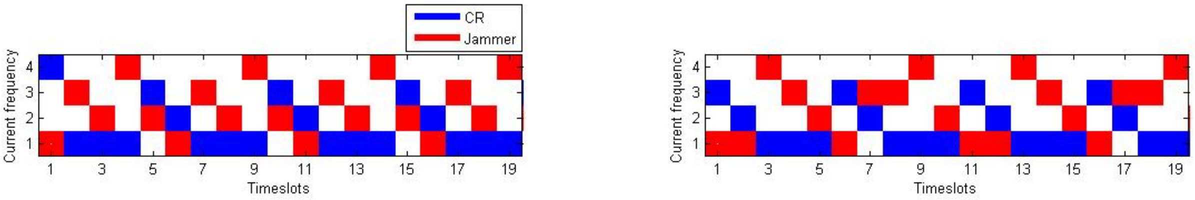
(a) $s_1 = (f_2, 1)$

(b) $s_1 = (f_3, 1)$

Fig. 2. Exploitation of the learned policy against a reactive jammer

5.3. Scenario with a pseudo random jammer

In this scenario, we consider a jammer with a pseudo random strategy. We suppose that at every timeslot, this jammer attacks randomly one of the four frequencies, and after a period T it repeats the same sequence of the attacked frequencies. We started with a period $T = 5$ during which the random sequence is $(1, 3, 2, 4, 2)$, we found that the OPSQ-learning algorithm converges in this scenario after four episodes. Then, we considered a period $T = 10$ during which the random sequence is $(1, 1, 4, 3, 2, 1, 3, 3, 4, 2)$, we found that the OPSQ-learning algorithm converges in this scenario after five episodes. The CR succeeds to learn the pseudo random strategy of the jammer, and the learned anti-jamming strategies are given in Fig. 3 when the periods of the pseudo random jamming sequences are respectively $T = 5$ and $T = 10$ timeslots.



(a) $T = 5$

(b) $T = 10$

Fig. 3. Exploitation of the learned policy against a pseudo random jammer

5.4. Discussion

The standard Q-learning algorithm converges after about one hundred episodes. Each episode starts from a random frequency, going randomly from one frequency to another taking random decisions until collision with the jammer. The CR applying this technique have to either wait for all this training period to get an anti-jamming strategy or to use it during real time communication and sacrifice about hundred lost packets.

The ON-policy synchronous Q-learning algorithm converges faster than the standard Q-learning algorithm; it gives a suitable defense strategy with the less possible frequency hopping after about four training episodes against sweeping, reactive and even pseudo random jammers. This is due to the synchronous update of all Q-values of possible actions from a current state, which helps the CR to faster improve its beliefs about all decisions without trying all of the actions. Furthermore, the choice of taking at every timeslot the best action (until the actual moment) promotes the real time exploitation of the OPSQ-learning algorithm during the CR communication. Because the OPSQ algorithm learns the safe strategy (it takes the action selection method into account when learning), it receives a higher average reward per trial than Q-Learning as given by Fig. 4. But, we should mention that the proposed OPSQ-learning algorithm doesnt optimize the entire matrix Q, it just optimizes the Q-values of state/action pairs that the CR goes through until finding an anti-jamming strategy.

6. Conclusion

The jamming attack is a challenging issue in cognitive radio networks because it can hinder the efficient exploitation of the spectrum. In this paper, we model the jamming mitigation problem as an MDP model and we start by using the standard version of the Q-learning algorithm to sole it. However, this algorithm takes a long training period before the convergence to an anti-jamming strategy which is inefficient to the real time working of the CRNs. We have proposed a modified version to overcome the limits of the Q-learning algorithm, and we call the proposed algorithm as the ON-policy synchronous Q-learning (OPSQ-learning) algorithm. We have presented the simulation results of the application of both the standard Q-learning and the OPSQ-learning algorithm under sweeping, reactive and pseudo random jamming strategies. We can conclude that the OPSQ-learning version speeds up the learning period and can be applied during CRN real time communication. As future work, the presented solution will be tested in real platform and real environment.

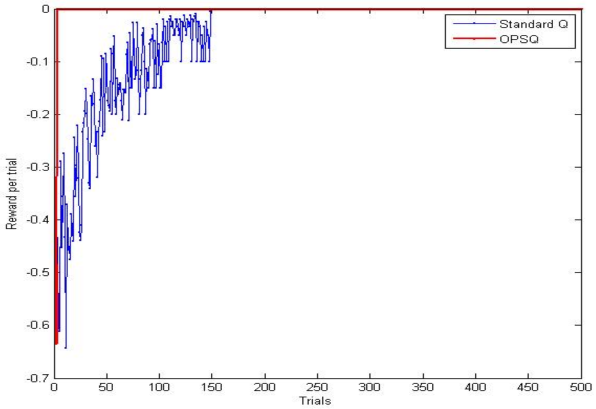


Fig. 4. Comparison between Q-learning and OPSQ-learning

References

1. Mitola III, J., G.Q. Maguire Jr, . Cognitive radio: making software radios more personal. *IEEE Personal Communications Magazine* Aug. 1999;**6**(4):13–18.
2. Mahmoud, Q.. *Cognitive Networks: Towards Self-Aware Networks*. John Wiley and Sons; 2007.
3. Ponuratinam, G., Patel, B., Elleithy, S.S.R.K.M.. Improvement in the spread spectrum system in DSSS, FHSS, and CDMA 2013;.
4. Wang, W., Bhattacharjee, S., Chatterjee, M., Kwiat, K.. Collaborative jamming and collaborative defense in cognitive radio networks. *Pervasive and Mobile Computing* 2013;**9**(4):572–587.
5. Pöpper, C., Strasser, M., Čapkun, S.. Anti-jamming broadcast communication using uncoordinated spread spectrum techniques. *IEEE Journal on Selected Areas in Communications* 2010;.
6. Dong, Q., Liu, D., Wright, M.. Mitigating jamming attacks in wireless broadcast systems. *Wireless Networks* 2013;.
7. Asterjadhi, A., Zorzi, M.. JENNA: a jamming evasive network-coding neighbor-discovery algorithm for cognitive radio networks. *IEEE Wireless Communications* 2010;**17**(4):24–32.
8. Balogun, V.. Anti-jamming performance of hybrid FEC code in the presence of CRN random jammers. *International Journal of Novel Research in Engineering and Applied Sciences (IJNREAS)* 2014;**1**(1).
9. Bhunia, S., Su, X., Sengupta, S., Vázquez-Abad, F.J.. Stochastic model for cognitive radio networks under jamming attacks and honeypot-based prevention. In: *Distributed Computing and Networking - 15th International Conference (ICDCN '14), Coimbatore, India*. January 4-7 2014, p. 438–452.
10. Wang, B., Wu, Y., Liu, K.J.R., Clancy, T.C.. An anti-jamming stochastic game for cognitive radio networks. *IEEE Journal on Selected Areas in Communications* 2011;.
11. Dabcevic, K., Betancourt, A., Marcenaro, L., Regazzoni, C.S.. A fictitious play-based game-theoretical approach to alleviating jamming attacks for cognitive radios. In: *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference*. 2014, .
12. Wednel Cadeau Xiaohua Li, C.X.. Markov model based jamming and anti-jamming performance analysis for cognitive radio networks. Communications and Network; 2014, .
13. Wu, Y., Wang, B., Ray Liu, K.J.. Optimal defense against jamming attacks in cognitive radio networks using the markov decision process approach. In: *GLOBECOM'10*. 2010, p. 1–5.
14. Chen, C., Song, M., Xin, C., Backens, J.. A game-theoretical anti-jamming scheme for cognitive radio networks. *IEEE Network* 2013; **27**(3):22–27.
15. Watkins, C.J.C.H.. *Learning from Delayed Rewards*. Ph.D. thesis; King's College; Cambridge, UK; 1989.
16. Tesauro, G.. Extending Q-learning to general adaptive multi-agent systems. In: *NIPS*. MIT Press; 2003, .
17. Abounadi, J., Bertsekas, D.P., Borkar, V.S.. Stochastic approximation for nonexpansive maps: Application to Q-learning algorithms. *SIAM J Control and Optimization* 2002;**41**(1):1–22.
18. Even-Dar, E., Mansour, Y.. Learning rates for Q-learning. *Journal of Machine Learning Research* 2003;**5**:1–25.
19. A.Arokkiaraj, , T.Jayasankar, . OFDM based spectrum sensing in time varying channel. *International Refereed Journal of Engineering and Science (IRJES)* 2014;**3**(4):50–55.
20. Spyros Kyperountas, N.C., Shi, Q.. A comparison of fusion rules for cooperative spectrum sensing in fading channels. EMS Research, Motorola.
21. Andreas F. Molisch, L.J.G., Shafi, M.. Propagation issues for cognitive radio. *Proceedings of the IEEE* 2009;**97**(5).