Cooperation Reliability based on Reinforcement Learning for Cognitive Radio Networks

Nemanja Vučević[1]
Signal Theory and Communications Dpt.
Universitat Politècnica de Catalunya
Barcelona, 08034, Spain
Email: vucevic@tsc.upc.edu

Ian F. Akyildiz
Broadband Wireless Networking Lab.
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA, 30332, USA
Email: ian@ece.gatech.edu

Jordi Pérez-Romero
Signal Theory and Communications Dpt.
Universitat Politècnica de Catalunya
Barcelona, 08034, Spain
Email: jorperez@tsc.upc.edu

***Abstract* —The primary objective of cooperation in Cognitive Radio (CR) networks is to increase the efficiency and improve the network performance. However, CR users may act destructively and decrease both their own and others' performances. This can be due to Byzantine adversaries or unintentional erroneous conduct in cooperation. This work presents an autonomous cooperation solution for each CR user, i.e., each CR user decides with whom to cooperate. The objective of the proposed solution is to increase the spectrum access in cooperative CR networks. To realize this, a Reinforcement Learning (RL) algorithm is utilized to determine the suitability of the available cooperators and select the appropriate set of cooperators. In addition, the proposed solution determines the most appropriate number of cooperators to achieve the highest efficiency for spectrum access. Accordingly, the control communication overhead is reduced. The simulation results demonstrate the learning capabilities of the proposed to achieve reliable behavior under highly unreliable conditions.**

Keywords- cognitive radio networks, cooperation, reinforcement learning, reliability.

I. INTRODUCTION

Cognitive Radio (CR) technology, as the core of CR networks, is a promising solution to spectrum scarcity and low spectrum use by Primary Users (PUs) [1]. Unlike primary wireless networks that have predefined frequency bands and are focused only on optimization of the resource utilization, CR networks need to perceive the behavior of PUs and perform opportunistic spectrum access without or with minimal interference to them. In order to do so, CR networks need to be capable of learning from the environment and adapt to current conditions.

Opportunistic spectrum access (spectrum sensing, spectrum decision, spectrum sharing) requires that the CR users continuously gather and process information. With aim to increase the fairness and the performance of the network (e.g. time and energy consumption for information obtaining and processing) cooperation is introduced in CR networks [2]. Despite the fact that the purpose of cooperation is to improve the overall network performance, this may lead to malicious behavior and unreliability in CR networks.

The cooperative CR networks need not only to dynamically adapt to the changes in the radio environment, but also they need to be resistant to various types of threats and system errors. The cooperation in CR networks needs to be performed with unknown or known cooperators that may change their behavior, in conditions where the knowledge of the success in cooperation is limited. Therefore, the cooperative CR networks are vulnerable to so-called Byzantine threats (i.e., traitors – inside threats from the cooperators that use its privileges to achieve its own goals). The cooperation reliability in CR networks can additionally be jeopardized by unintentional malicious effects (i.e., from the lack of reliability of devices or systems). Autonomous solutions, which are capable to learn and react dynamically according to the grade of reliability, should be developed to deal with these issues in real time.

In this paper we address the reliability problem in cooperative CR networks. We present a solution that determines the reliability of the cooperators and selects which are appropriate for cooperation. The cooperator evaluation and selection in the proposed solution is based on the Reinforcement Learning (RL) [3] which is a branch of machine learning envisaged as a good candidate for dynamic control and adaptation in CR networks [4]. One of the threats to a CR network may be learning from false beliefs, thus, the learned information should not be permanent [5]. Our proposed solution uses exploratory properties of reinforcement learning to provide the ability to relearn the changes and react with reconfiguration.

The rest of this paper is organized as follows. In the next section, the detailed problem formulation is given. In section III, the framework for the solution is described, followed by the algorithm description in section IV. Section V contains simulation results. Finally, the main conclusions are summarized in section VI.

II. PROBLEM STATEMENT

CR user needs to cooperate with other CR users (cooperators) in order to perform opportunistic spectrum access. A CR user collects information (the advices) from cooperators and makes a decision about its next actuation. The most straightforward application of this approach is in cooperative spectrum sensing, where the cooperators provide information about spectrum availability and the CR user decides whether to access a given spectrum or not. In a general case, the actuation may be any activity that satisfies the conditions explained in the remaining of this section (e.g., decisions regarding spectrum decision, spectrum sharing, and spectrum mobility).

Cooperating CR users only exchange "hard decisions" on the unknown hypothesis, H_0 (resource access not possible), H_1 (resource access possible). Cooperator i makes a local decision, and forwards it as an advice (denoted by $x(i)$) to the CR user. In this paper we assume the local decision is H_1 with the advice $x(i)=1$, and the local decision is H_0 with the advice $x(i)=0$. Hard decisions lower the communication overhead. Hard decision can be derived from "soft decision". For example: if the signal power is lower than a threshold, consider that a PU is not there.

[1]This work was conducted during his stay at the BWN Lab, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, in 2009.

978-1-4244-7213-0/10/$26.00 ©2010 IEEE

The usual data fusion rule for hard decisions is "K out of N" rule, meaning that at least K advices have to be H_1, so the global decision is H_1 [6]. The global decision is H_1 by a particular CR user to access a resource when $X=1$. Otherwise, it is H_0 when $X=0$.

We assume that N cooperators exist. However, not all of them are necessary for cooperation. A CR user may select M ($M \leq N$) cooperators based on their suitability to achieve the desired goal. The suitability should include both aspects of reliability and accuracy of the cooperators. In this paper, dynamic tracking of this suitability is carried out by means of reinforcement learning.

Regarding the above aspects, the proposed method needs: to:

- decide how many cooperators to use (M of N),
- learn how to select reliable cooperators (which M),
- make a decision from collected advices (K out of M).

A. Malicious behavior in cooperation

The malicious conduct in cooperation may be both intentional and unintentional:

- The intentional malicious behavior is due to the Byzantine adversary that pretends to be a friend and uses its privilege to achieve its own desired goal.
- The unintentional incorrect behavior may be from:
  - Cooperators that are manipulated by other opponent systems and are unaware of their malicious effects (e.g., sensors in range of jammers);
  - Technological limitations (i.e. hardware and software errors, failures and limitations);
  - Environment conditions (e.g. high shadowing zone).

Additional complexity in the detection of the malicious cooperators comes from the fact that cooperators may change their behavior in time. Thus, a system needs to be capable of capturing possible dynamic changes.

B. Erroneus advices and decisions

Cooperator i can make erroneous positive advice when the resource access is not possible ($x(i)=1|H_0$) and erroneous negative advice when the resource access is possible ($x(i)=0|H_1$). Probabilities of these erroneous advices are:

$$p_{err}(i) = \Pr\{x(i) = 0 \mid H_1\} \quad (1)$$

$$q_{err}(i) = \Pr\{x(i) = 1 \mid H_0\} \quad (2)$$

The probabilities of erroneous advice (1-2) influence the final error probabilities of decision. The probabilities of erroneous non-actuation (P_{ERR}) and erroneous actuation (Q_{ERR}) are:

$$P_{ERR} = \Pr\{X = 0 \mid H_1\} \quad (3)$$

$$Q_{ERR} = \Pr\{X = 1 \mid H_0\} \quad (4)$$

An optimal system will tend to minimize both $P_{ERR} \rightarrow 0$ and $Q_{ERR} \rightarrow 0$. However, the erroneous resource access is often more destructive than erroneous non-actuation, because it can lead to interference with primaries. Therefore, Q_{ERR} must be kept within very low limits to assure none or the minimal interference as a higher priority than minimization of P_{ERR}.

An additional constraint in CR networks is that the evaluation of the actuation is only possible once the resource access has actually happened. This means that the erroneous resource access is perceived from the actuation itself (e.g., collision after accessing a spectrum when PU is active). Nevertheless, the erroneous non-actuation is usually not possible to detect (e.g. missed opportunities). In other words, there is no feedback when $X=0$. Thus, as a direct notion of P_{ERR} is not available, the way to minimize P_{ERR} is to maximize the resource access, i.e. to maximize $\Pr\{X=1\}$.

C. Challenges

The goal is to maximize $\Pr\{X=1\}$, but at the same time preserve Q_{ERR} within very low limits.

The following additional challenges are addressed:

1. The local decision of the cooperators is unknown and independent in each cooperator (cooperators are heterogeneous).
2. In this paper we consider a worst case scenario where probabilities $q_{err}(i)$ and $p_{err}(i)$ are unknown, uncorrelated, and can vary dynamically and independently of each other.
3. Probabilities $\Pr\{H_1\}$ and $\Pr\{H_0\}$ are unknown and may change with time.
4. There is no knowledge on the success of decisions not to actuate, i.e. when $X=0$.
5. Number of cooperators (N) can vary with time.

D. Related work

To the best of authors' knowledge, the problem with all the aforementioned assumptions and conditions has not been addressed in the literature up to date. However, as the proposed solution combines suitability evaluation, data fusion and decision making in unreliable conditions, some partially related studies are briefly discussed in the following.

The classification of how good are cooperators often results in trust evaluation and is commonly compared with human behavior characteristics [7]. The application of trustworthiness for reliable path selection in ad-hoc and wireless sensor networks is present in the literature, e.g. [8]. Existing studies distinguish between direct and indirect trusts and address influence of reputation, respect and rumors on trust. Similarly, our CR user also evaluates and assigns a grade of suitability to cooperators. The suitability in our work reflects reliability of the cooperators that are used to make further decisions.

Studies on expert advices address the problem of how to use several experts to guess hypothesis as close as possible to the best expert's guesses, e.g. [9]. However, assumptions taken in existing studies never take into account all the explained constraints and limitations at the same time (challenges 1-5). Moreover, the behavior of the best cooperator does not guarantee that $\Pr\{X=1\}$ is maximized and does not preserve Q_{ERR} within very low limits with higher priority.

The problem to maximize $\Pr\{X=1\}$ while Q_{ERR} is preserved within low limits appears as a Neyman-Pearson detection problem in the literature [6] - where the data fusion from sensors should optimize the decision following the "K out of N" rule. Due to their high complexity, the methods used for solving this problem have only limited applications in cases with small number of highly similar sensors [6]. These solutions are also not applicable for the entire set of challenges 1-5.

III. CONSIDERED FRAMEWORK

It is assumed that N cooperators are available to a CR user. Both the CR user and its cooperators may be a CR base station or a CR user in either centralized or ad-hoc CR networks. A CR user needs to determine which of the available cooperators to use in order to maximize its goals. Each cooperator has a

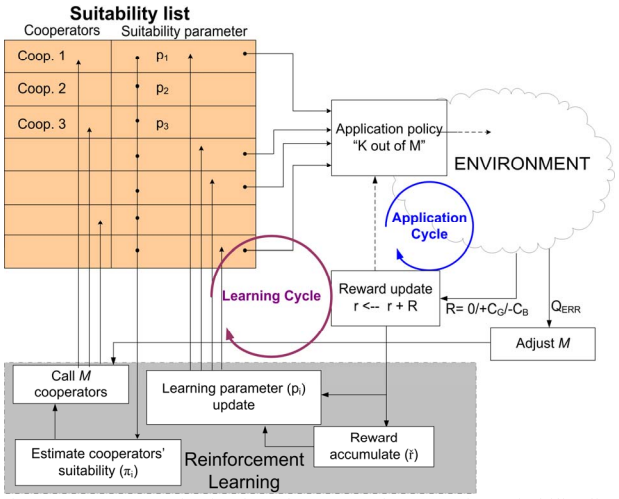

Fig. 1: Cooperator selection based on RL and cooperation suitability list.

grade of reliability that can change with time when the cooperator changes behavior or when the general conditions change. The evaluation of cooperation suitability depends on the final outcomes of Q_{ERR} and P_{ERR} when a cooperator is used. The proper learning mechanism should assure that the malicious cooperators are not used. Correct cooperator selection also reduces the signaling load of cooperation.

A. *Suitability list and cooperator selection*

Each CR user performs its own evaluation and selection based on personal experience. To this end, a CR user makes a list (i.e. the suitability list) of the N cooperators available to him at a given moment. The appearance and disappearance of cooperators (e.g., due to mobility) does not affect the algorithm, as they can easily be added or removed from the list. Each of these cooperators is characterized with a learning parameter p_i, which is used to define their suitability π_i.

In Fig. 1, the general framework is presented where CR user selects M cooperators from the suitability list and starts an application period (i.e., a period in which the cooperation is carried out with the selected group of M cooperators). During this period T_A application cycles are applied. In each application cycle the resource access decision depends on the current advices of the M cooperators in use. The decision is derived from the application policy (K out of M). After each decision, the reward r for that application period is updated.

Each application period is followed by one learning cycle. The learning cycle starts from the reward obtained in the predecessor application period and updates the values of the learned parameter p_i for each cooperator i used in that period. After this update, the reward accumulated $\check{r}$ from the learning algorithm is also updated. Finally, the suitability π_i is computed for all available cooperators. Based on π_i values, a new set of cooperators is formed for the cooperation in the following application period.

The learned values p_i are the key parameters in defining the suitability of each cooperator. Apart from the global reward value, these are the only values that need to be stored by the learning mechanism.

Even more flexibility in adaptation can be achieved by adapting M to CR user's needs through time. This change can be performed after the application period, independently from the RL algorithm that maintains the suitability list.

B. *Cooperation communication*

We assume the existence of a control channel over which the cooperation is performed. CR user either pools (invites) cooperators or listens to all cooperators and just ignores the ones denoted as unreliable. In the first case, the invitation to cooperate is performed only once after each application period (T_A application cycles) and only if there has been a change in the set of used cooperators.

In case when cooperators are invited for cooperation, the CR user also needs to perform occasional checkup of possibly new cooperators. This can be done only when the current performance is low for a long period, or when too many cooperators have disappeared. Note that the CR user can optionally set a maximum limit for the number of cooperators in a suitability list. With this option it can occasionally remove highly unreliable cooperators and replace them with the new ones.

We assume direct data gathering from the cooperators. In general, data gathering can also be indirect (i.e. through an intermediate cooperator). Then, two options are possible. In the first option, the intermediate cooperator makes a unique local decision from gathered data and forwards it to a CR user. This makes only the intermediate cooperator visible in suitability list. In the second option, all the gathered data is forwarded. This opens the possibility to data falsification [10], however, it would only change the original advice error probabilities of the indirect cooperators. In both cases, the proposed solution remains unchanged.

IV. Learning and Reasoning

The application policy and the learning mechanism are first described for a fixed number of used cooperators (M). Afterwards, the framework is extended for a variable M.

A. *Application policy*

The application policy defines the advice interpretation by the system. As the solution works with hard decisions, the applied policy decides to actuate ($X=1$) only if at least K out of M advices are $x(i)=1$ ("K out of M"):

$$X = \begin{cases} 1, & \sum_{i=1}^{M} x(i) \geq K \\ 0, & \sum_{i=1}^{M} x(i) < K \end{cases} \tag{5}$$

which makes the error probabilities P_{ERR} and Q_{ERR}, defined in eq. (3) and (4):

$$P_{ERR} = 1 - \sum_{j=K}^{M} \Pr\left\{ \left(\sum_{i=1}^{M} x(i) \right) = j \mid H_1 \right\} \tag{6}$$

$$Q_{ERR} = \sum_{j=K}^{M} \Pr\left\{ \left(\sum_{i=1}^{M} x(i) \right) = j \mid H_0 \right\} \tag{7}$$

The dependence of P_{ERR} and Q_{ERR} on M and K is discussed in the Appendix.

From equations (6) and (7) we see that for a given M, K closer to M decreases Q_{ERR}, but increases P_{ERR}. As the primary objective of our solution is to maintain Q_{ERR} very low, $K=M$ can be the safest choice (i.e., all the cooperators have to decide $x(i)=1$ in order to have $X=1$). However, we select $K=M-1$ in this paper so the learning mechanism can permit one cooperator to advise $x(i)=0$ and still to decide $X=1$. This permits the learning algorithm to perform occasional exploration and learn

faster on the cooperators that may be malicious and are among selected M cooperators. Then P_{ERR} and Q_{ERR} become:

$$P_{ERR} = 1 - \left(\prod_{i=1}^{M} (1 - p_{err}(i)) \right) \cdot \left(1 + \sum_{i=1}^{M} \frac{p_{err}(i)}{1 - p_{err}(i)} \right) \quad (8)$$

$$Q_{ERR} = \left(\prod_{i=1}^{M} q_{err}(i) \right) \left(1 + \sum_{i=1}^{M} \frac{1 - q_{err}(i)}{q_{err}(i)} \right) \quad (9)$$

where the error probabilities $p_{err}(i)$, $q_{err}(i)$, P_{ERR} and Q_{ERR} are defined in equations (1-4).

When $K=M-1$, than higher M increases P_{ERR} but decreases Q_{ERR} (see Appendix). However, the probability Q_{ERR} is low with only two cooperators with low $q_{err}(i)$ (equation (26) in Appendix). Thus, with few cooperators having low $q_{err}(i)$ and good cooperator selection, the Q_{ERR} can be maintained low with M that does not have to be high. When this is not the case, we account on the adaptation of M explained in subsection *C* to increase M and protect Q_{ERR} when necessary. Finally, it is left to the learning mechanism to encounter the best cooperators to maintain P_{ERR} value also as low as possible for a given M.

B. Cooperator selection

The learning capabilities of the proposed solution rely on reinforcement learning (RL) [3] algorithms to maintain the suitability parameter of the prospective cooperators. RL is a branch of machine learning where an agent learns through interaction with environment and decides on actions in order to maximize some long term reward.

This work starts from the actor-critic learning, in particular from the REINFORCE algorithm [11]. Actor critic methods require minimal computation in order to select actions. The critic is a function that takes the form of temporal difference error and "criticizes" the actor's behavior - evaluates if the results have gone better or worse than expected [3]. This is carried out based on the interaction with the environment through a reward function. Additionally, in this work, the correction to the global reward (critic) is introduced to lower the number of trials during exploration. This is similar to the supervisor that, by correcting the temporal difference error of the critic, contributes to faster learning [12] (Fig.2). In this paper the reward (critic) correction penalizes currently selected cooperators that do not contribute to correct actuation.

The global reward function r is accumulated throughout the application period. At each application cycle when the final decision is actuation ($X=1$) reward is updated as:

$$r \leftarrow r + X \cdot \left(\alpha \cdot C_G + (1 - \alpha) \cdot C_B \right) \quad (10)$$

Constants C_G and C_B define reward increment for correct and wrong actuations. Parameter $\alpha=1$ when $X=1|H_1$, or $\alpha=0$ when $X=1|H_0$.

The reward from equation (8) corresponds to the overall actuation performance. The reward correction ρ_i for each active cooperator i is also computed for every application period:

$$\rho_i = \sum_{t=1}^{T_A} (1 - x_t(i)) X_t \Big/ \sum_{t=1}^{T_A} X_t \quad (11)$$

This reward represents the number of times the cooperator i gave advice $x(i)=0$ when decision was $X=1$, divided with the number of decisions to actuate ($X=1$) in that application period.

After each application period a learning cycle follows. The learning parameter of each active cooperator i is updated:

$$p_i \leftarrow p_i + \beta \cdot (r - \breve{r} - \xi \cdot \rho_i) \cdot (1 - \pi_i) \quad (12)$$

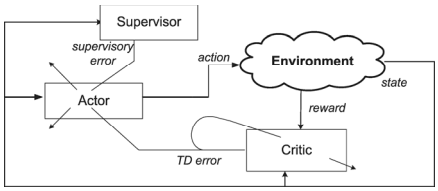

Fig. 2: Supervised actor-critic reinforcement learning.

Here, β and ξ are positive constant parameters and π_i is the suitability of the cooperator i to be selected for the cooperation. The parameter $\breve{r}$ is the global rewards accumulate. It is updated after p_i of all the active cooperators have been updated:

$$\breve{r} \leftarrow \gamma \cdot r + (1 - \gamma) \cdot \breve{r} \quad (13)$$

where the parameter γ is constant, $0<\gamma\leq 1$.

Finally, once the updates in the suitability parameter list are done, the suitability of each cooperator is obtained as:

$$\pi_i = e^{p_i} \Big/ \sum_{j=1}^{N} e^{p_j} \quad (14)$$

The suitability value π_i determines the probability of each of the cooperators to be selected as active for the following application period.

C. Variation of M

Aimed at having a more adaptive system, the number of used cooperators (M) can be variable. In other words, when Q_{ERR} is high, M should be incremented. However, when Q_{ERR} is low, M may be decremented so that the P_{ERR} may be decreased more easily (see Appendix). To this end, Q_{ERR} is tracked for each application period (i.e., percent of time $X=1|H_0$), and averaged for the last T application periods (learning cycles):

$$S(T) = \frac{1}{T} \sum_{s=-T}^{0} \left(\sum_{t=1}^{T_A} \left(X(s,t) \cdot H_0(s,t) \right) \Big/ \sum_{t=1}^{T_A} X(s,t) \right) \quad (15)$$

Here, s stands for the index of the application period, whereas t is the index of the application cycle.

The average $S(T)$ is independently tracked for the previous T_X and T_Y learning cycles. Two threshold values are defined to control M: T_H^U and T_H^L ($T_H^L < T_H^U$). M is changed as:

$$S(T_X) > T_H^U \rightarrow \quad M = M + 1 \quad (16)$$

$$S(T_Y) < T_H^L \rightarrow \quad M = M - 1 \quad (17)$$

Let us assume now that the number of the application periods from the last increment of M is T_X^*. Then, as long as $T_X^* < T_X$, instead of using condition from equation (16), the modified condition to increment M is used:

$$T_X^* < T_X \rightarrow \quad S(T_X^*) > T_H^U \cdot (T_X + T_X^*) / 2T_X^* \rightarrow \quad M = M + 1 \quad (18)$$

Similarly, if M has been decreased before T_Y^* application periods, and as long as $T_Y^* < T_Y$, instead of (17), the modified condition for the next decrement of M is:

$$T_Y^* < T_Y \rightarrow \quad S(T_Y^*) < T_H^L \cdot (2T_Y^* - T_Y) / T_Y^* \rightarrow \quad M = M - 1 \quad (19)$$

The previously explained use of conditions (18) and (19) is done in order to disable consecutive changes of M due to the values that already have contributed to a change of M. Thus, the learning algorithm has time to readapt itself to the new number of cooperators in use.

Whenever more than 3 cooperators are available ($N\geq 3$), we set the minimum number for M to be 3. This is due to the $K=M-1$, in order to have majority ($K>M/2$) when deciding to access spectrum (equation (5)). When this is not the case ($N<3$), one or two cooperators can also be used.

V. SIMULATION RESULTS

A. Learning and reconfiguration

The parameters in the simulation are: $\beta=0.01$, $\xi=0.1$, $\gamma=0.4$, $T_Y=100$, $T_X=10$, $T_H^U=0.001$, $T_H^L=T_H^U/100$, $T_A=20$, $C_G=2$ and $C_B=-20$. Initial values are $p_i(0)=0$, $M_0=5$ (for variable M). When new cooperator i appears, p_i takes a random value.

This set of results demonstrates the learning process. It is assumed that there are ten cooperators (*C1-C10*) available and that the error probabilities $p_{err}(i)$ and $q_{err}(i)$ change as given in Table I. After every *2000* application cycles some of the probabilities change. Notation "--" indicates that the cooperator is unavailable after that moment. For this set of results, the resource availability is randomly generated with $\Pr\{H_1\}=\Pr\{H_0\}=0.5$. Statistics are averaged over the last *200* application periods.

Fig.3a-c show an example of the adaptation for $M=5$. The probabilities to select the cooperators change through time with change of the error probabilities $p_{err}(i)$ and $q_{err}(i)$. More than five cooperators are performing well enough to be included in cooperation at the beginning. Each time one of the used cooperators decreases performances by increasing an error probability (either $p_{err}(i)$ or $q_{err}(i)$), RL tries to reconfigure if necessary. Appearance and disappearance of cooperators is also followed by similar expected reaction. Note that when cooperator i appears, learning parameter p_i is randomly assigned to it. So when *C1* and *C4* appear at the same time at the 18000th learning cycle, the probability to use *C4* is high, whereas the probability to use *C1* is low. However, the quality in the behavior of these two cooperators is perceived and these values are corrected fast. Also note that during the application periods from 6000th-8000th, there are only 3 cooperators with $p_{err}(i)=0.01$ and one with $p_{err}(i)=0.5$, whereas the rest of them have $p_{err}(i)=0.99$. Then, there is no combination of $M=5$ cooperators that can give lower P_{ERR} than *0.5* (see eq. (6)). Similarly, for the application periods between 16000th-18000th, the combination of available $q_{err}(i)$ values does not permit Q_{ERR} values as low as T_H^U for $M=5$ (see eq. (7)). Consequently, in this period average Q_{ERR} is $E(Q_{ERR})=0.0022>2T_H^U$ (Fig.3c).

Fig.3d presents the performances (P_{ERR} and Q_{ERR}) for the case $M=4$. Now, P_{ERR} is significantly lower between the 6000th-8000th application periods. However, as expected, having one cooperator less can be a limitation for Q_{ERR}. So, between the 16000th-18000th application periods, the average value of Q_{ERR} is even higher, $E(Q_{ERR})=0.0059\approx 6T_H^U$.

TABLE I. BEHAVIOR OF COOPERATORS THROUGH SIMULATION

Application period (Learning Cycle)		Cooperators [C1	C2	C3	C4	C5	C6	C7	C8	C9	C10]
0	p_{err} =	[0.01	0.50	0.01	0.01	0.01	0.01	0.01	0.01	0.99	--]
	q_{err} =	[0.50	0.50	0.01	0.01	0.01	0.01	0.01	0.50	0.01	--]
2000	p_{err} =	[0.01	0.50	0.01	0.01	0.01	0.01	0.99	0.01	0.99	--]
	q_{err} =	[0.50	0.50	0.01	0.01	0.01	0.01	0.01	0.50	0.01	--]
4000	p_{err} =	[--	0.50	0.01	0.01	0.01	0.99	0.99	0.01	0.99	--]
	q_{err} =	[--	0.50	0.01	0.01	0.01	0.01	0.01	0.50	0.01	--]
6000	p_{err} =	[--	0.50	0.01	0.01	0.01	0.99	0.99	0.99	0.99	--]
	q_{err} =	[--	0.50	0.01	0.01	0.01	0.01	0.01	0.50	0.01	--]
8000	p_{err} =	[--	0.50	0.01	0.01	0.01	0.99	0.99	0.99	0.99	0.01]
	q_{err} =	[--	0.50	0.01	0.01	0.01	0.01	0.01	0.50	0.01	0.01]
10000	p_{err} =	[--	0.50	0.01	0.01	0.01	0.99	0.99	0.99	0.01	0.01]
	q_{err} =	[--	0.50	0.01	0.01	0.01	0.01	0.01	0.50	0.01	0.01]
12000	p_{err} =	[--	0.50	0.01	--	0.01	0.99	0.99	0.99	0.01	0.01]
	q_{err} =	[--	0.50	0.01	--	0.01	0.01	0.01	0.50	0.01	0.01]
14000	p_{err} =	[--	0.50	0.01	--	0.01	0.01	0.99	0.01	0.01	0.01]
	q_{err} =	[--	0.50	0.01	--	0.01	0.50	0.50	0.50	0.50	0.01]
16000	p_{err} =	[--	0.50	0.01	--	0.01	0.01	0.99	0.01	0.01	0.01]
	q_{err} =	[--	0.50	0.01	--	0.50	0.50	0.50	0.50	0.50	0.01]
18000	p_{err} =	[0.01	0.50	0.01	0.99	0.01	0.01	0.99	0.01	0.01	0.01]
	q_{err} =	[0.01	0.50	0.01	0.50	0.01	0.50	0.50	0.50	0.50	0.01]

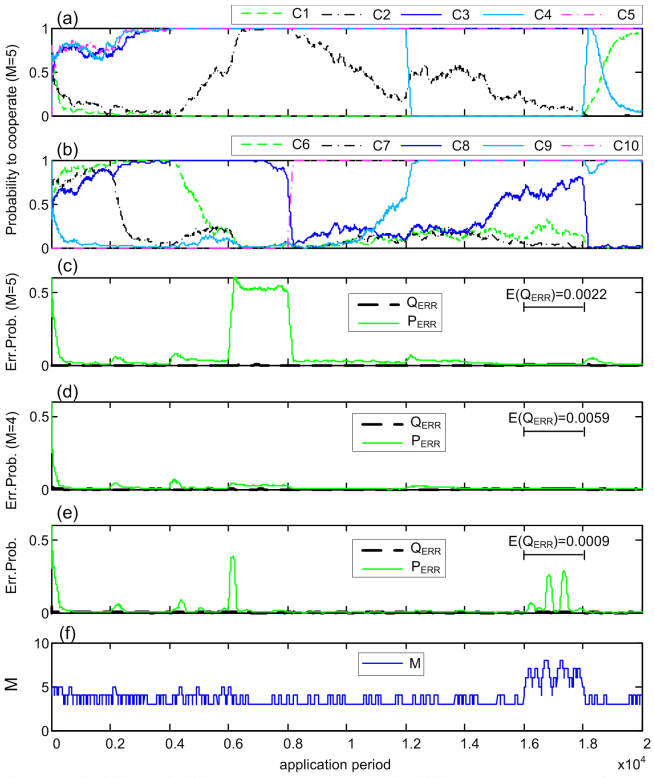

Fig. 3: Suitability of different cooperators (probability to cooperate)(a-b) and performance results for $M=5$ (c); performance results for $M=4$ (d); performance results when M is variable (e) and corresponding M value (f).

Finally, Fig.3e-f give the results for variable M. Now, the variation of M allows reducing P_{ERR} in periods in which Q_{ERR} is not close to the threshold (between 6000th-8000th application periods). In the periods from 16000th-18000th, the values of P_{ERR} are higher than in both simulations with fixed M. Nevertheless, this tradeoff is done so the average value of Q_{ERR} is now below the T_H^U ($E(Q_{ERR})=0.0009$ in Fig3e).

As it is presented in section II, the primary objective was to maintain Q_{ERR} very low and then to minimize P_{ERR} as much as possible. The presented results show how the mechanism maintains Q_{ERR} low even in transitional periods (i.e., simulation initialization and preference changes when conditions worsen). At the same time, fast convergence of the RL algorithm lowers P_{ERR} relatively fast (e.g., after only *~300* application periods, even for $M=5$, P_{ERR} average is lowered below *0.05* without any previous knowledge - at session initialization).

B. Performance evaluation

In this section the total number of cooperators is $N=15$. Resource availability is simulated as two states of exponentially distributed length with mean of *500* application periods. The spectrum access is randomly available with probabilities $\Pr\{H_1\}=0.2$ and $\Pr\{H_1\}=0.8$ in the two states. The simulation length is $2\cdot 10^5$ application periods for each point.

All cooperators change their behavior independently for $p_{err}(i)$ and $q_{err}(i)$ within separate time intervals. Each change happens after an exponentially generated number of application periods with mean $1/\lambda_p=1/\lambda_q=2000$. Two behavior types are distinguished in this example - defined by the range from

which error rates $p_{err}(i)$ or $q_{err}(i)$ take value from. When cooperator i wants to change $p_{err}(i)$ or $q_{err}(i)$, it first selects range interval: I_{1P} (I_{1Q}) with probability P_1 (Q_1) or I_{2P} (I_{2Q}) with probability P_2 (Q_2). Afterwards, the error rate $p_{err}(i)$ or $q_{err}(i)$ is uniformly selected from the chosen interval.

For simplicity reasons, we set $P_1=Q_1$ ($P_2=Q_2$), whereas behavior intervals are:

Case 1: $(I_{1Q},I_{1P},I_{2Q},I_{2P})=(0\text{-}0.05, 0\text{-}0.05, 0.95\text{-}1.00, 0.95\text{-}1.00)$,

Case 2: $(I_{1Q},I_{1P},I_{2Q},I_{2P})=(0\text{-}0.05, 0\text{-}0.05, 0.95\text{-}1.00, 0.05\text{-}1.00)$,

Case 3: $(I_{1Q},I_{1P},I_{2Q},I_{2P})=(0\text{-}0.05, 0\text{-}0.05, 0.05\text{-}1.00, 0.95\text{-}1.00)$,

Case 4: includes two cooperators with constant perfect behavior (i.e., $p_{err}(i)=0$, $q_{err}(i)=0$), whereas the rest 13 cooperators behave as in case 1.

The results are presented in Fig.4. The proposed algorithm is compared with fixed "K out of N" solution. For each set (Q_1,Q_2,P_1,P_2) best "K out of N" result is taken from $K=1,..,N$. We consider that the valid results are those that preserve Q_{ERR} close to the used threshold ($<1.2\cdot T_H^U$, as this is the upper limit within which proposed solution maintains Q_{ERR}). Thus, best "K out of N" solution is the one that is valid and has lowest P_{ERR}. Depending on the conditions this may be for different K. For example for $P_2=Q_2=0.1$ best $K=7$ for cases 1 and 2, best $K=8$ for case 3 and best $K=6$ for case 4; whereas for the $P_2=Q_2=0.4$ best $K=12, 11, 13$ and 10 for the cases 1,2,3 and 4, respectively.

In all the cases for $P_2=Q_2>0.2$, the proposed solution demonstrates significantly higher robustness. Cases 2 and 3 do not change general tendencies in behavior significantly when compared to case 1. With two perfect cooperators, best K in simulations for "K out of N" gives still high P_{ERR} values, similar to the cases 2 and 3. However, for the case 4, the proposed solution lowers P_{ERR} to 0. As expected from section IV, the learning mechanism manages to identify the cooperators to achieve good performances while reducing M, which leads to a considerable improvement of performance.

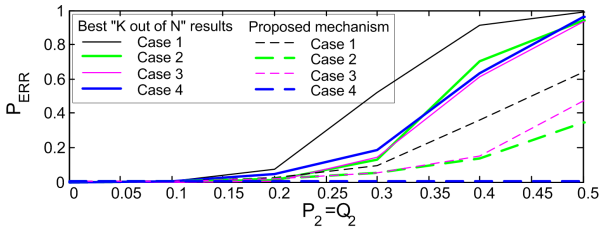

Fig. 4: P_{ERR} as a function of $P_2=Q_2$.

VI. CONCLUSIONS

This paper has addressed reliability in cooperative cognitive radio networks. The problem is how to maximize opportunistic radio access, with minimal or no interference to primary users, when cooperators advising on resource access are unreliable or malicious. We proposed a reinforcement learning-based solution that maintains a suitability list of cooperators and selects with which of them to cooperate. Results demonstrate the capability of the proposed solution to successfully learn and act in dynamic hostile environments. Additionally, the proposed solution adapts the number of used cooperators in accordance with their performances. When there are few highly accurate cooperators, the proposed solution successfully identifies them and bases the spectrum access decision only on their advices to achieve very high performances.

APPENDIX

Some additional expressions regarding dependence of P_{ERR} and Q_{ERR} on M and K are derived here. From the general expressions for the error probabilities (6,7), it is clear that when K is closer to M, the probability P_{ERR} decreases, whereas Q_{ERR} increases. Special case of (6,7) when $K=M-1$ are eq. (8,9).

If the eq. (8,9) are now presented as:

$$P_{ERR}(M)=1-A_M\left(1+B_M\right) \tag{20}$$

$$Q_{ERR}(M)=C_M\left(1+D_M\right) \tag{21}$$

where A_M, B_M, C_M and D_M are:

$$A_M=\prod_{i=1}^{M}\left(1-p_{err}(i)\right),\quad B_M=\sum_{i=1}^{M}\frac{p_{err}(i)}{1-p_{err}(i)} \tag{22}$$

$$C_M=\prod_{i=1}^{M}q_{err}(i),\quad D_M=\sum_{i=1}^{M}\frac{1-q_{err}(i)}{q_{err}(i)} \tag{23}$$

than recursive expressions, when one more cooperator is included, $(M+1)^{\text{th}}$, can be derived as follows:

$$P_{ERR}(M+1)=1-A_M\left(1+B_M\left(1-p_{err}^{*}\right)\right) \tag{24}$$

$$Q_{ERR}(M+1)=C_M\left(1+D_M q_{err}^{*}\right) \tag{25}$$

where the error probabilities of the $(M+1)^{\text{th}}$ cooperator are $p_{err}(M+1)=p_{err}^{*}$, and $q_{err}(M+1)=q_{\text{err}}^{*}$.

From the previous equations it is clear that P_{ERR} increases and Q_{ERR} decreases no matter how good or bad values p_{err}^{*} and q_{err}^{*} are. Additionally, this may be presented for Q_{ERR} as:

$$Q_{ERR}(M)\le\min\left\{Q_{ERR}(m)\right\},\quad \forall m, 2\le m<M \tag{26}$$

ACKNOWLEDGMENT

This work is supported by the US National Science Foundation under contract ECCS-0900930; by the Spanish Research Council and FEDER funds under COGNOS grant (ref. TEC 2007-60985); and MEC-FPU Scholarship (AP2005-3739).

REFERENCES

[1] I.F. Akyildiz, W.Y. Lee, M.C. Vuran, S. Mohanty, "Next Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Networks: A Survey", *Computer Networks*, vol. 50, issue 13, Sept. 2006

[2] S.M. Mishra, A. Sahai, R.W. Brodersen, "Cooperative sensing among cognitive radios", IEEE ICC, Jun 2006

[3] R.S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction*, A Bradford Book, MIT Press, Cambridge, MA 1998

[4] R.W. Thomas, D.H. Friend, L.A. DaSilva, A.B. MacKenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives", *IEEE Communication Magazine*, vol 44, iss 12, Dec. 2006

[5] T. Clancy, N. Goergen, "Security in Cognitive Radio Networks: Threats and Mitigation", IEEE CrownCom, May 2008

[6] Q. Zhang, P.K. Varshney, R.D. Wesel, "Optimal bi-level quantization of i.i.d. sensor observations for binary hypotesis testing", *IEEE Transactions on Information Theory*, vol 48, no 7., July 2002

[7] J.L. Burbank, W.T.M. Kasch, "The Application of Human and Social Behavioral-Inspired Security Models for Self-aware Collaborative Cognitive Radio Networks", CollaborateCom, Nov. 2008

[8] G. Han, D. Choi, W. Lim, "A reliable and efficient approach of establishing trust for wireless sensor networks", IEEE ICCP, Sept. 2007

[9] N. Cesa-Bianchi, *et al.*, "How to use expert advice", *Journal of the ACM*, vol. 44, issue 3, May 1997

[10] Chen Ruiliang, Park Jung-Min, Y.T. Hou, J.H. Reed, "Toward secure distributed spectrum sensing in cognitive radio networks", *IEEE Communications Magazine*, vol. 46, issue 4, April 2008

[11] R.J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine Learning*, vol. 8, 1992

[12] M.T. Rosenstein, A.G. Barto, "Supervised Actor-Critic Reinforcement Learning", in *Learning and Approximate Dynamic Programming: Scaling Up to the Real World*, John Wiley & Sons, New York, 2004