Learning Interference Strategies in Cognitive ARQ Networks

Sina Firouzabadi, Marco Levorato, Daniel O'Neill and Andrea Goldsmith
Dept. of Electrical Engineering, Stanford University, Stanford, CA 94305 USA.
e-mail: {cna, levorato, dconeill, andreag}@stanford.edu.

***Abstract*—Cognitive radios, which enable the coexistence on the same bandwidth of licensed primary and unlicensed secondary users, have the potential for dramatically increasing the efficiency of wireless networks. In this paper, we propose an on line learning algorithm to optimize the transmission strategy of secondary users in interference mitigation scenarios, where the secondary users are allowed to superimpose their transmission onto those of the primary users. Due to practical limitations, the secondary users have access to only a fraction of the current state of the primary users' network. Therefore, the strategy of the secondary users is defined on a reduced state space. Numerical results show that the proposed practical learning algorithm operates close to the performance of the system under full knowledge.**

I. Introduction

Cognitive networks offer the promise of greater efficiency and aggregate network throughput than existing spectrum allocation methods [1]–[4]. In cognitive networks, unlicensed secondary users opportunistically access radio bandwidth owned by licensed primary users in order to maximize their performance, while limiting interference to primary users' communications.

Much prior work focuses on a *white space* approach [5], where the secondary users are allowed to access only those time/frequency slots left unused by the users. The strategy of the secondary users is then a *sensing* strategy, meaning that the secondary users decide whether or not to access the channel by performing a binary hypothesis test based on the outcome of channel sensing [6]. If idealized sensing and slotted time are assumed,[1] then a white space approach guarantees that secondary users' activity does not interfere with the primary user. However, due to noise and fading, sensing errors are inevitable. Therefore, in practical scenarios, the secondary users' activity results in a non-zero collision probability, which can be measured and used as a constraint for the optimization problem [6]. Other work investigates the coexistence of primary/secondary signals in the same time/frequency band by focusing on physical layer methods for static scenarios, *e.g.*, [7], [8]. While white space approaches address the recognition of dynamic idle/busy channel patterns, this latter class of approaches finds a practical application in broadcasting primary users' networks.

Recent work [9]–[11] proposes an *interference mitigation* approach, where secondary users are allowed to superimpose their transmissions onto those of the primary users in dynamic wireless networks[2] with constraints on the degradation of primary users' average performance metrics (*e.g.*, average throughput and packet delivery probability). In this approach the behavior and internal state of the primary transmitter is observed by the secondary transmitter, which adapts its behavior to maximize its performance, while limiting its impact on the primary. The resultant secondary transmission policy is optimal over the possible states of the primary and secondary transmitters.

However, this approach assumes that the secondary transmitter has complete and perfect knowledge of the current state and probabilistic model of the primary transmitter/receiver pair, limiting its applicability. For example, while it is likely that the secondary might read ACKs for the primary system, it is unlikely that the secondary will have knowledge of the pending workload of packets at the primary transmitter or will know the distribution of packet arrivals at the primary transmitter.

In this paper we address this limitation by developing an on line learning approach that uses only observable state information and that approximately converges to the optimal secondary control policy. The observable state information is a subset of the complete state of the primary system and thus will result in a loss in performance when compared to interference mitigation approach under full information. Simulations suggest however that this loss in secondary performance is relatively small.

We center our analysis on a network with a single primary user storing packets in a finite-buffer and implementing Automatic Retransmission reQuest (ARQ). We will show that when the secondary user has access to the observable state of the network, an on line algorithm can obtain performance similar to an off line algorithm with complete state information.

The rest of this paper is organized as follows. Section II describes the network model. The Markov process modeling the state of the network and the optimization problem are provided in Section II-B. The reduced state space representing the perceived network state at the secondary user is defined in Section III. Section IV presents the on line learning algorithm. Numerical results are discussed in Section V. Section VI concludes the paper.

II. Network Model, Objective and Constraints

We consider an *interference mitigation* scenario, where the secondary users are allowed to superimpose their transmissions

[1] If communications are asynchronous, a user may access the channel before a secondary user's transmission ends. In this case, although the secondary user correctly sensed an idle channel before starting the transmission, a collision may occur.

[2] By dynamic networks, we mean networks characterized by a stochastic evolution of the state of the nodes due to random events, such as packet arrivals, packet failures, etc.

onto those of the primary users, and the network dynamics are modeled by means of a homogeneous stochastic process. In interference mitigation the secondary user is constrained to no more than a fixed maximum degradation of the primary's performance. This is in contrast to the white space approach which places a maximum on the probability of collision, but not on the overall impact of the secondary user's degradation to the performance of the primary. In the following, constraints on the minimum packet delivery probability and minimum average throughput achieved by the secondary users are defined.

These approaches illustrate two fundamental aspects of cognitive networking in dynamic environments:

- control protocols implemented by primary users *react* to the interference generated by the secondary users, and therefore the stochastic process modeling primary users' state depends on secondary users' strategy;
- while in traditional white space approaches the strategy is based on a binary idle/active representation of the primary user network, if primary users implement control mechanisms, then the optimal strategy must be defined on the, typically non-binary, state space induced by those mechanisms.

Figure 1 depicts our model. The primary has a finite buffer and implements an ARQ protocol. This protocol reacts to the interference from the secondary. the secondary in turn senses this reaction and adjusts its behavior. The result is a stochastic interference process that is a function of the protocols implemented by both the primary and the secondary.

Section II-A, II-B and II-C describe in detail primary and secondary link operations and protocols, the stochastic model of the network and the optimization problem, respectively.

A. Network Description

The primary source stores packets to be sent to the primary receiver in a finite buffer of size B packets. Packets arrive in the buffer according to a Poisson process of intensity λ. In order to improve packet delivery probability, the primary source implements an ARQ error control protocol with finite retransmissions. Therefore, decoding failure at the primary receiver triggers retransmission of the packet currently being served by the primary source, unless the maximum number of packet's transmissions F has been reached. After F transmissions, a packet is removed from the buffer and returned to the higher layers, either if it was successfully delivered or the receiver failed the last retransmission. The primary receiver transmits a short ARQ feedback packet reporting successful/failed decoding of each primary source's data packet transmission.

For the sake of simplicity, the secondary source is assumed to be backlogged. Moreover, according to the common characterization of secondary users as *best effort* opportunistic users, it is assumed in the following that the secondary source transmits its own packets only once.[3] After each secondary source's transmission, the secondary receiver feeds back whether or not it successfully decoded its intended packet.

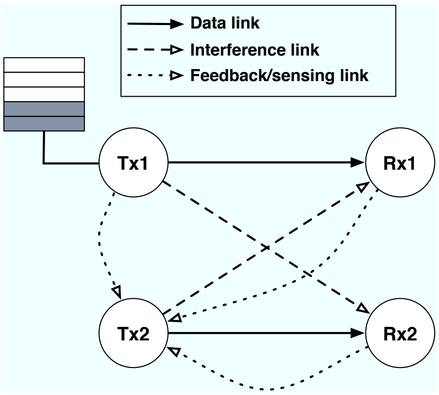

Figure 1. Model of the network. Solid, dashed and dotted arrows denote data, interference and feedback/sensing links, respectively.

The system operates in slotted time $t{=}\{1,2,3,\ldots\}$, where slot t is associated to the time interval $\big((t{-}1)\tau, t\tau\big]$ and τ denotes the duration of a slot. Packets have a fixed size of L-bits, and transmission of a packet plus its associated feedback message fits the duration of a slot.[4] We assume that the primary source accesses the channel in slot t to transmit a packet if its buffer at time $(t-1)\tau^+$ is non-empty.

The probability that the primary receiver fails to decode a primary source's packet if the secondary source is idle and the secondary source transmit is equal to ρ and ρ^*, respectively. Analogously, the probability that the secondary receiver fails to decode a secondary source's packet if the primary source is idle or if the primary source transmits is equal to ν and ν^*, respectively.[5]

B. Stochastic Model

The network can be modeled as a Markov process $\mathbf{\Phi}{=}\{\Phi_0,\Phi_1,\Phi_2,\ldots\}$, where Φ_t, $t{=}1,2,\ldots$, takes values in the state space $\mathcal{X}$. The state Φ_t is described by the pair (b_t,f_t), with $b_t{\in}\{0,1,\ldots,B\}$ and $f_t{=}\{0,1,\ldots,F\}$, where b_t is the number of packets in the buffer at the beginning of slot t and f_t is the ARQ transmission index of the packet currently being served by the primary source.[6] Thus, the state space is $\mathcal{X}{=}\{1,\ldots,B\}{\times}\{1,\ldots,F\}\cup\{0,0\}$, where state $\{0,0\}$ corresponds to an empty buffer. Since the secondary source is assumed to have a packet to transmit in every slot and does not implement ARQ, then its state does not need to be tracked.

The state transition probabilities are described by

$$P(\phi,\phi',u)=\mathcal{P}(\Phi_{t+1}=\phi'|\Phi_t{=}\phi,u_t{=}u),\ \forall\phi,\phi'{\in}\mathcal{X}, \quad (1)$$

where $u_t{\in}\{0,1\}{=}\mathcal{U}$ is the action of the secondary source in slot t, and $u_t{=}0$ and $u_t{=}1$ correspond to idleness and transmission of the secondary source, respectively.

From state $(0,0)$, $\mathbf{\Phi}$ moves to $(b',1)$ if $b'{=}1,\ldots,B$ packets arrive in the primary source's buffer. From state (b,f), $f{<}F$, the process moves to $(b',f{+}1)$ if $b'{-}b$ packets arrived and the primary source's transmission failed. If the transmission

[3] Extensions to this model can be investigated without significant changes to the proposed framework.

[4] We remark that a slot can be accessed by both the sources simultaneously.

[5] This corresponds, for instance, to fixed transmission rate and random *i.i.d.* channels.

[6] For instance, $f_t{=}3$ corresponds to a packet already transmitted twice, that will be transmitted for the third time in slot t.

is successful, then the process moves to $(b',1)$, $b'{>}0$ if $b'{-}b+1$ packets arrived, and to $(0,0)$ if $b{=}1$ and no packets arrived. If $f{=}F$, then the process moves to either $(0,0)$ or $(b,1)$ according to the queue length and packet arrivals. The transition kernel, then, is straightforwardly derived from the packet arrival rate and failure probability.

C. Optimization Problem

Define the cost functions $x_i(\phi,u) : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ as the average cost incurred by the Markov process in state $\phi{\in}\mathcal{X}$ if action $u{\in}\mathcal{U}$ is chosen, and the time averages

$$X_i(\boldsymbol{u}){=} \lim_{n\to+\infty} \inf \frac{1}{n}\sum_{t=1}^{n} \mathbb{E}\big[x_i\left(\Phi_t,u_t,\epsilon_t(\Phi_t,u_t)\right)\big], \qquad (2)$$

where $\boldsymbol{u}{=}\{u_1,u_2,\ldots\}$ is the sequence of actions of the secondary source and $\epsilon_t(\Phi_t,u_t)$ is an exogenous random variable that depends both on the state and the action at time slot t.

In particular, we define

$$x_0(\phi,u,\epsilon) = \begin{cases} 1 & \text{if } u{=}0,\ \forall\phi\in\mathcal{X} \\ B_\nu & \text{if } u{=}1, \phi{=}(0,0) \\ B_{\nu^*} & \text{if } u{=}1, \phi{\neq}(0,0), \end{cases} \qquad x_1(\phi,u,\epsilon) = \begin{cases} 1 & \text{if } \phi=(0,0) \\ B_\rho & \text{if } u{=}0, \phi{\neq}(0,0) \\ B_{\rho^*} & \text{if } u{=}1, \phi{\neq}(0,0). \end{cases} \qquad (3)$$

where B_ν and B_{ν^*} are binary random variables with parameters ν and ν^* that represent the failure probability of the secondary user under the transmission or no-transmission of the primary user respectively. It is easy to see that $X_0(\boldsymbol{u})$ is equal to $1{-}\Theta_\text{s}$, where Θ_s is the normalized throughput achieved by the secondary source under control $\boldsymbol{u}$. Similarly, $X_1(\boldsymbol{u})$ is $1{-}\Theta_\text{p}$, where Θ_p is the normalized throughput achieved by the primary source under control $\boldsymbol{u}$.

We also define

$$x_2(\phi,u,\epsilon) = \begin{cases} B_\rho & \text{if } u{=}0, \phi=(b,F) \\ B_{\rho^*} & \text{if } u{=}1, \phi=(b,F) \\ 0 & \text{otherwise,} \end{cases} \qquad x_3(\phi,u,\epsilon) = \begin{cases} 1 & \text{if } \phi=(b,1) \\ 0 & \text{otherwise.} \end{cases} \qquad (4)$$

$$(5)$$

Thus, $X_2(\boldsymbol{u})$ can be interpreted as the fraction of time slots in which the primary source fails the last allowed transmission and the packet would not be delivered and $X_3(\boldsymbol{u})$[7] is the fraction of time slots where the primary begins the service of a new packet. In this paper we define the failure probability as the average ratio of dropped packets after F retransmission, to the total number of new packets sent[8], one can see that $X_2(\boldsymbol{u})/X_3(\boldsymbol{u})$ is equivalent to the failure probability of the primary source's packets.[9] The optimization problem is then defined as

$$\min X_0(\boldsymbol{u}),\ \text{s.t. } X_1(\boldsymbol{u}) < \gamma_1,\ \frac{X_2(\boldsymbol{u})}{X_3(\boldsymbol{u})} < \gamma_2. \qquad (6)$$

Therefore, the secondary source selects the control sequence $\boldsymbol{u}$ for maximizing its own throughput, with constraints on the minimum normalized primary source's throughput of $1-\gamma_1$ and the maximum failure probability of primary source's packets γ_2.

It is shown in [12] that the optimization problem in Eq. (6) is solved by a stationary randomized policy $\mu(\phi,u){:}\mathcal{X}{\times}\mathcal{U}{\to}\mathbb{R}$ with a number of *randomizations* equal to or fewer than the number of independent constraints. Here $\mu(\phi,u)$ corresponds to the probability that action $u{\in}\mathcal{U}$ is selected in state $\phi{\in}\mathcal{X}$. The optimal policy can be found as the solution of an appositely defined Linear Program [10], [12].

III. State Knowledge

The offline solution of the optimization problem requires full knowledge of state Φ_t, which corresponds to the ARQ and queue state of the primary source, as well as knowledge of the transition probabilities and cost functions. However, the full knowledge of Φ_t requires an explicit exchange of information.

We address this limitation in two steps. First, by assuming that the secondary only has information about what can be directly observed about the primary, and second, by using an on line learning technique that *learns* the necessary parameters without requiring knowledge of the transition probabilities.

By sensing the channel, the secondary source detects the existence of the primary source's signal. Since the primary source would be idle in slot t only if $\Phi_t{=}(b_t,f_t){=}(0,0)$, *i.e.*, it has an empty buffer, the secondary source can distinguish state $(0,0)$ from any other state. Therefore, channel sensing provides to the secondary source a binary representation of the buffer in slot t, *i.e.*, $b_t{=}0$ if the primary source is idle and $b_t{>}0$ if the primary source transmits.

Moreover, if $b_t{>}0$, the primary source can retrieve the ARQ state f_t by overhearing the header of the packet sent by the primary source. In fact, the header includes the sequence number of the packet, which increases if the transmitted packet is a new one and remains the same if it is a retransmission. Therefore, the secondary source can exactly estimate f_t by spending a small fraction of the slot overhearing the primary source's transmission.

As a consequence of the previous discussion, in practical scenarios the secondary source perceives a *reduced* state space induced by a binary empty/non-empty representation of the queue state, and an accurate representation of the ARQ state. In particular, the secondary source bases its strategy on the state space $\{0,1,\ldots,F\}$, where state 0 corresponds to $(0,0)$ and state $f{>}0$ is associated with the state aggregate (b,f), with $b{>}0$.

Finally, the functions $x_i(\Phi_t,u_t,\epsilon(\Phi_t,u_t))$ are directly observable at the secondary source by ACK/NACK messages decoding.

It is shown in the following section via numerical results that this partial knowledge is sufficient to implement a learning algorithm operating close to the limit provided by full state knowledge.

[7] Note that $X_3(\boldsymbol{u})$ is not a function of ϵ

[8] Failure probability can also be also seen as the probability that the F–th transmissions of a new packet fails.

[9] Note that $X_2(\boldsymbol{u})$ alone is not the failure probability. Consider the case where the primary sends a single packet in the first F time slots and all of these transmissions fail (let $\rho = \rho^* = 1$) and it stays quiet ever after. In this case X_2 which is the expected portion of time slots that primary fails would be zero (simply because its one divided by infinity), while the failure probability is not zero(in fact it is one).

We remark that by *partial knowledge*, we do not mean that the secondary source has a noisy observation of the state, but rather that, due to practical limitations, the secondary source has exact knowledge of only a fraction of the state. Noisy feedback, sensing and header decoding may limit the accuracy of state and cost estimation.

IV. Learning Algorithm

Most approaches to optimal control require knowledge of an underlying probabilistic model of the system dynamics which requires certain assumptions to be made, and this entails a separate estimation step to estimate the parameters of the model. In particular, in our optimization paradigm (6), the optimal randomized stationary policy can be found if the failure probabilities $\rho,\ \rho^*,\ \nu,\ \nu^*$ are known to the secondary user, together with the full knowledge of state Φ_t. The other implicit assumption here is that these failure probabilities are fixed and they are not changing in time. This assumption also might not be true in general, especially when the users are mobile or when the channel conditions change in time. In this section we describe how we can use an adaptive learning algorithm called R-learning to find the optimal policy without a priori knowledge about our probabilistic model.

The R-learning algorithm is a long-term average reward reinforcement learning technique. It works by learning an action-value function $R_t(\phi, u)$ that gives the expected utility of taking a given action u in a given state ϕ and following a fixed policy thereafter. Intuitively, the R-function captures the relative cost of the choice of a particular allocation for the next time-step at a given state, assuming that an optimal policy is used for all future time steps. Like its counterpart Q-learning, R-learning is based on the adaptive iterative learning of R factors. In order to use R-learning for our problem, we need to convert the constrained MDP problem (6) to an ordinary MDP with no constraints by solving the dual problem with the introduction of Lagrange multipliers for each constraint. Define

$$J^*(l_1, l_2) = \min_{\boldsymbol{u}} X_0 + l_1(X_1(\boldsymbol{u}) - \gamma_1) + l_2(X_2(\boldsymbol{u}) - \gamma_2 X_3(\boldsymbol{u})) \tag{7}$$

where l_1, l_2 are the Lagrange multipliers for the primary sources throughput constraint and constraint on the maximum failure probability of the primary sources, respectively. These Lagrange multipliers can be interpreted as the prices that we put for violating constraints. Now the dual problem is to maximize $J^*(l_1, l_2)$ with respect to prices l_1, l_2. First note that for a particular choice of prices l_1, l_2, the optimal policy for (7) can be found by following the standard R-learning algorithm. Note that when we fix l_1, l_2, the solution of (7) is equivalent to finding the optimal stationary policy for an infinite horizon average cost problem of the form

$$\min_{\boldsymbol{u}} \lim_{n\to+\infty} \inf \frac{1}{n} \sum_{t=1}^{n} \mathbb{E}\big[c_i\left(\Phi_t, u_t, \epsilon_t(\Phi_t, u_t)\right)\big], \tag{8}$$

where

$$c_i\left(\Phi_t, u_t, \epsilon_t\right) \overset{def}{=} x_0\left(\Phi_t, u_t, \epsilon_t\right) + l_1(x_1\left(\Phi_t, u_t, \epsilon_t\right) - \gamma_1) + \\ + \ l_2(x_2\left(\Phi_t, u_t, \epsilon_t\right) - \gamma_2 x_3\left(\Phi_t, u_t, \epsilon_t\right)). \tag{9}$$

Here, $c_i(.)$ is the immediate cost function at time slot t for the action u_t, given that we are in state Φ_t. The R-learning algorithm for solving (8) consists of the following steps[s1]:

1) Let time step t=0. Initialize all the $R_t(\Phi, u)$ values (say to 0). Let the current state be Φ.
2) Choose the action u that has the highest $R_t(\Phi, u)$ value[10] with some probability, say $1 - \alpha_t$, else let u be a random exploratory action. In other words let
$$u_t = \arg\max_u R_t(\Phi, u) \tag{10}$$
3) Carry out action u. Let the next state be Φ', and the cost from (9) be $c_i\left(\Phi_t, u_t, \epsilon_t\right)$. Update the R values and the average cost $\bar{C}$ using the following rules:
$$R_{t+1}(\Phi, u) = R_t(\Phi, u)(1-\beta_r) + \\ +\beta_r\left(c_i(\Phi_t, u_t, \epsilon_t) - \bar{C}_t + \min_u R_t(\Phi', u)\right) \tag{11}$$
$$\bar{C}_{t+1} = \bar{C}_t(1-\beta_c) + \\ +\beta_c\left(c_i(\Phi_t, u_t, \epsilon_t) + \min_u R_t(\Phi', u) - \min_u R_t(\Phi, u)\right) \tag{12}$$
4) Set the current state to Φ' and go to step 2.

Here $0 \le \beta_r \le 1$ is the learning rate controlling how quickly errors in the estimated action values are corrected, and $0 < \beta_c < 1$ is the learning rate for updating $\bar{C}$. Convergence of R-learning requires that all action-state pairs should be visited infinitely often. Hence, all potentially important action-state pairs must be explored and this is usually done in practice via the exploratory factor α_t in step 2. Note that the average cost $\bar{C}$ is updated only when a non-exploratory action is performed. After the convergence, the optimal decision that the algorithm takes at state Φ_t is equal to $\arg\max_u R(\Phi, u)$. For fixed values of the prices l_1, l_2, if each action-state pair is visited an infinite number times and the learning rates β_r, β_c are decayed appropriately, the above learning algorithm will converge to the optimal strategy [13], [14]. Now in order to solve the dual problem completely, we need to find the optimal prices l_1, l_2 that can be done on line with a minor change in step 3 of the algorithm. Utilizing the stochastic subgradient method, we can update the prices at each iteration by adding these updates at step 3:

$$l_1 = \left[l_1 + \beta_{l_1}\left(x_1\left(\Phi_t, u_t, \epsilon_t\right) - \gamma_1\right)\right]^+ \tag{13}$$

$$l_2 = \left[l_2 + \beta_{l_2}\left(x_2\left(\Phi_t, u_t, \epsilon_t\right) - \gamma_2 x_3\left(\Phi_t, u_t, \epsilon_t\right)\right)\right]^+, \tag{14}$$

where, $[\ \cdot\ ]^+$ is equivalent to $\max(\ \cdot\ , 0)$ and $\beta_{l_1},\ \beta_{l_2}$ are constant step sizes for updating the Lagrange multipliers. In the next section, we discuss briefly why we choose constant step sizes instead of the more typical shrinking step sizes.

V. Numerical Results

In this section we illustrate the performance of the proposed learning algorithm in Section IV. We will also discuss some implementation problems of the proposed R-learning algorithm. We use the network model described in Section

[10] Basically, in a given state ϕ, $R_t(\phi, u)$ gives the expected utility/reward of taking a given action u and by maximizing $R_t(\phi, u)$ over u we are essentially choosing the best action with highest expected reward.

II and run the proposed learning algorithm in Section IV for two different scenarios. One is the case when the secondary user can completely observe the state of the system, i.e. it has access to both b_t and f_t. The other case is when the secondary source can only detects the existence of the primary source signals plus the retransmission state f_t through sensing the channel. This later scenario is more practical since it can be implemented without any exchange of information as we discussed in Section III. The goal is to compare the performance of the learning algorithm in these two scenarios and to see how much we lose by having partial information about the buffer state of the primary.

Throughout the simulation we assume that the buffer size of the primary source is $B = 6$ and the maximum retransmission time is $F = 4$. We set the failure probabilities for the transmission of the primary source $\rho = 0.2$, $\rho^* = 0.5$, depending on the fact that secondary is silent or not, respectively. Similarly, the failure probabilities of the secondary source are set to be $\nu = 0.3$ and $\nu^* = 0.5$. Note that these failure probabilities are not known at the secondary source and it has to learn the optimal policy without any assumption on these parameters in advance. We also restrict the failure probability of the primary source to be less than 0.02 and the goal is to maximize the throughput of the secondary source.

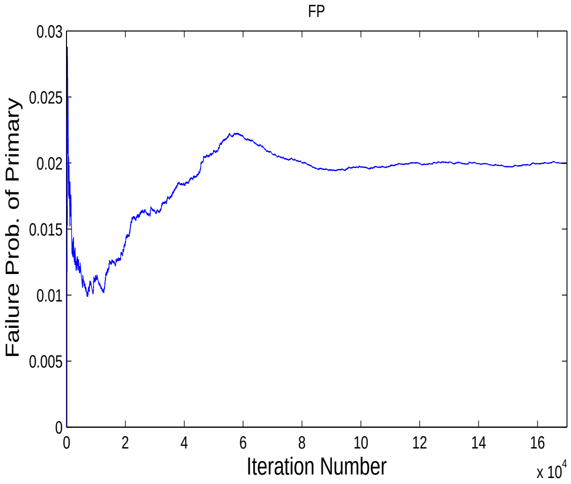

Figure 2. Failure probability of the primary user.

Figure 2 shows the convergence of the failure probability for the primary source throughout the iterations in the first scenario when the learning algorithm has full knowledge of the state when the arrival rate of the packets was set to 0.4. Figure 3 illustrates the value of the Lagrange multiplier associated with the constraint we have for failure probability of the primary source. According to [13], the optimal random policy of an MDP with one constraint is a probabilistic mixture of two deterministic polices taken with probabilities p_1 and $1 - p_1$ which depends on the constraint in the optimization problem. One observation that we had in our simulations, that can be seen also in Figure 2, is that the Lagrange multiplier does not converge fully and it starts bouncing back and force in an interval. We conjecture that this behavior leads to an indeterministic policy in time that might be close to the optimal randomized policy of the constrained MDP. Verifying this observation and obtaining the conditions under which we can find the optimal randomized policy with this method is a subject of our current research.

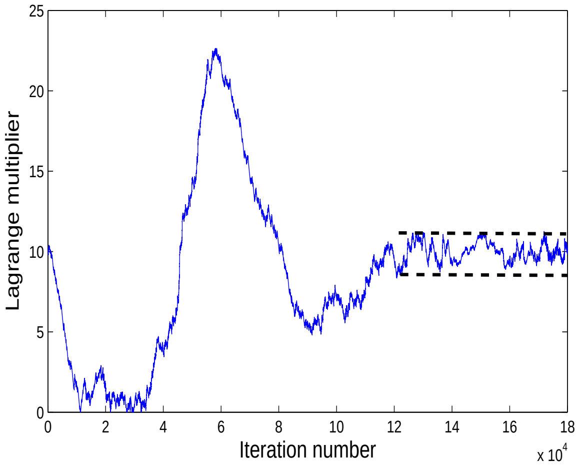

Figure 3. Lagrange multiplier associated with the failure probability of the primary.

Figure 4 compares the performance of the learning algorithm in two scenarios when we have full knowledge of the state space with the case that the secondary source has only a binary representation of the buffer at each time slot. For both scenarios, the achieved throughput of the secondary source is plotted for different arrival rates. Note that the performance of the secondary source, in term of throughput, decreases with increasing the arrival rate of the primary source. This is mainly because of the fact that increasing the traffic of the primary sources will decrease the opportunity of the cognitive network for utilizing white spaces.

As we can see in Figure. 4, in later case where we have limited observation, the average throughput loss compared to the full state knowledge case is near two percent. It can be observed that the cost functions only the empty/non-empty binary representation of the queue, which is directly observable by the secondary source. Therefore, the approximation lies in the steady-state distribution of the state aggregates resulting in a certain cost function measured by the secondary source. However, the relatively small difference between the performance of the learning algorithm under full and partial state knowledge shows that the reduced state space is a good representation of the full state space for the performance metrics considered herein.

VI. Conclusions

We propose an on line learning approach to interference mitigation for cognitive networks. Our approach relies only on the observable states of the primary transmitter and uses R-learning to converge to nearly optimal secondary transmitter control policies. We show that how the secondary source can retrieve an important part of the state by combined channel

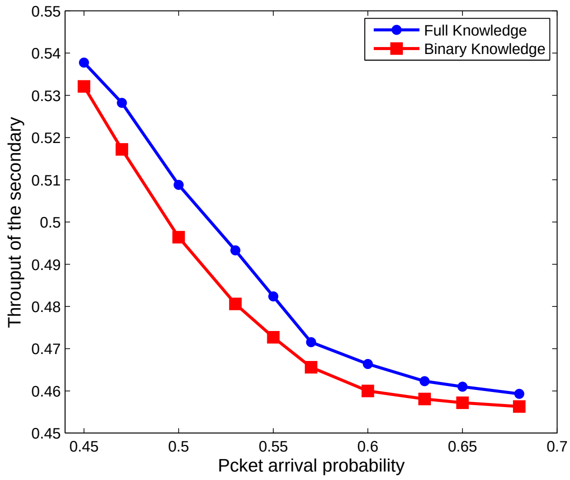

Figure 4. Comparison of the performance of the learning algorithm with and without full knowledge of the buffer state of the primary.

sensing, with reception of primary packets header and ARQ messaging. Numerical simulations suggest that this approach offers performance that is close to the performance of the system when complete system state information is known. This work is a stepping stone towards designing a practical MAC protocol for the cognitive users with limited and possibly noisy observations of the state of the primary network.

References

[1] Q. Zhao and L. Tong and A. Swami and Y. Chen, "Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: a POMPD framework," *IEEE J. Select. Areas Commun.*, vol. 25, no. 3, pp. 589–600, Apr. 2007.

[2] O. Simeone and Y. Bar-Ness and U. Spagnolini, "Stable throughput of cognitive radios with and without relaying capability," *IEEE Trans. Wireless Commun.*, vol. 55, no. 12, pp. 2351–2360, Dec. 2007.

[3] H. Su and X. Zhang, "Cross–layer based opportunistic MAC protocols for QoS provisioning over cognitive radio wireless networks," *IEEE J. Select. Areas Commun.*, vol. 26, no. 1, pp. 118–129, Jan. 2008.

[4] S. Haykin, "Cognitive radio: brain–empowered wireless communications," *IEEE J. Select. Areas Commun.*, vol. 23, no. 2, pp. 201–220, Feb. 2005.

[5] S. Geirhofer and L. Tong and B. M. Sadler, "Dynamic Spectrum access in the time domain: modeling and exploiting white space," *IEEE Commun. Mag.*, vol. 45, no. 5, pp. 66–87, May 2007.

[6] Y. Chen and Q. Zhao and A. Swami, "Joint design and separation principle for opportunistic spectrum access in the presence of sensing errors," *IEEE Trans. Inform. Theory*, vol. 54, no. 4, pp. 2053–2071, May 2008.

[7] W. Zhang and U. Mitra, "A spectrum-shaping perspective on cognitive radio: uncoded primary transmission case," in *Proc. of IEEE ISIT*, Toronto, Ontario, Canada, July 2008.

[8] Y. Xing and C. N. Mathur and M. A. Haleem and R. Chandramouli and K. P. Subbalakshmi, "Dynamic spectrum access with QoS and interference temperature constraints," *IEEE Trans. Mobile Comput.*, vol. 6, no. 4, pp. 423–433, Apr. 2007.

[9] M. Levorato, U. Mitra, and M. Zorzi, "On optimal control of wireless networks with multiuser detection, hybrid ARQ and distortion constraints," in *Proc. of the 28th IEEE Conference on Computer Communications (IEEE INFOCOM)*, Rio de Janeiro, Brazil, Apr. 2009.

[10] ——, "Cognitive interference management in retransmission-based wireless networks," in *Proc. of the 47th Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, Sept. 2009, pp. 94–101.

[11] F. Lapiccirella, S. Huang, X. Liu, and Z. Ding, "Feedback-based access and power control for distributed multiuser cognitive networks," in *Information Theory and Applications Workshop, 2009*, 8-13 2009, pp. 85 –89.

[12] K. W. Ross, "Randomized and past-dependent policies for Markov decision processes with multiple constraints," Operations Research, vol. 37, no. 3, pp. 474–477, May-June 1989.

[13] S. Mahadevan , "Average reward reinforcement learning: foundations, algorithms, and empirical results," Machine Learning, vol. 22, no. 1-3, pp. 159–195, Jan. 1996.

[14] D. P. Bertsekas and J. Tsitsiklis, *Neuro-dynamic programming*. Belmont, MA: Athena Scientific, 1996.