Distributed Learning Approach for Channel Selection in Cognitive Radio Networks

Chowdhury Sayeed Hyder and Li Xiao
Department of Computer Science & Engineering
Michigan State University
East Lansing, Michigan 48823
Email: \{hydercho, lxiao\}@cse.msu.edu

I. INTRODUCTION

Cognitive Radio Network (CRN) is an emerging networking concept that has been introduced to solve the spectrum scarcity problem. There are typically two kinds of users in CRN - primary users (PU) and secondary users (SU). SUs can opportunistically use licensed channels without interfering with the ongoing transmission of PUs. The successful transmission depends mostly on the selection of idle channels by SUs. Hence, channel selection under the aforementioned constraints is a significant research problem.

Accordingly, many researchers have developed channel selection algorithms based on different variations of PU channel usage models (e.g. Markov model) or adopted learning algorithms. However, in real scenarios, SUs do not know the activity of PUs in advance and channel usage by PUs may not exactly follow any mathematical model. On the other hand, existing learning algorithms (e.g. \[1\],\[2\]) either need a central base station or exchange of message passing between neighbors for cooperation and collision avoidance. Due to the unique characteristics of CRN, the centralized solution may not always be feasible and message passing among nodes will introduce excessive overhead.

The learning algorithm proposed in \[3\] overcomes these problems. In this approach, channels are ranked according to g-statistics. Although this learning algorithm achieves logarithmic regret, the sum regret under this approach is still high and it does not scale well with the increase of users. As a result, performance degrades when the ratio between the number of users and channels exceeds a certain threshold. The aggravated delay and packet loss introduced from channel switching may degrade the performance of the overall network. In the case of CRN, the effect can be more intense. Since every learning algorithm involves exploration and exploitation, ignoring channel switching cost may radically impact the performance. Furthermore, the work in \[3\] did not consider dynamic channel idle rates. In real life, primary users' (e.g. cell phone users) activity varies with time, so channels' idle rates also vary accordingly. Our algorithm adapts to such realistic changing environment and achieves same logarithmic regret.

In this paper, we address the channel selection problem with switching cost and propose a distributed learning approach that minimizes the sum regret while ensuring quick convergence to an optimal solution and logarithmic regret. Our algorithm is adaptive in the sense that it adapts to the changing idle status of channels and achieves logarithmic regret even in a dynamic environment. The experimental result shows that our algorithm outperforms the algorithm in \[3\] in terms of regret, scalability and channel switching cost.

II. PROBLEM FORMULATION

We consider a slotted network where every SU has only one radio and cannot sense and transmit at a time. So, each SU goes through a cycle of sensing channel and transmitting data if it is free. Consider a network where \(U \geq 1\) be the number of secondary users and \(C \geq U\) be the available channels. We assume that each channel \(i\) has a fixed probability \(\mu_i\) of being idle (or, not being occupied by any PU).

Every secondary user can sense only one channel in each transmission slot and transmits if the channel is idle. A successful transmission is considered as reward. Hence, SU receives reward if the channel is idle and no other SU transmits over the same channel.

We consider regret bound over the time averaged throughput or utilization rate to evaluate the performance of our algorithm because even a sub-linear regret (with respect to time) implies optimal average throughput \[3\]. In respect to our work, we define regret as the expected loss in achieving reward due to not selecting one of the top available channels compared to knowing the channel statistics perfectly.

Ideally if the users know about the channel availability, they will select only the top channels. In this case, the expected reward, \(S(\rho, t)\) will be:

\[
S(\rho, t) = t \cdot \sum_{i=1}^{U} r_i \cdot \mu_i^*
\]

where \(\mu_i^*\) denotes idle probability of \(i^{th}\) ranked (in terms of availability) channel and \(r_i\) denotes the data rate of \(i^{th}\) channel. For simplicity, we assume data rate of all channels are equal and commonly represent as \(r\).

Any learning approach except \(\rho\) tries to select a channel with higher availability through exploration. Let, \(S(\hat{\rho}, t)\) denotes the expected reward (i.e. number of successful data transmissions) by a learning approach \(\hat{\rho}\). It is obvious that \(S(\hat{\rho}, t) \leq S(\rho, t)\).
The expected regret $R(\hat{\rho}, t)$ can be determined using following equation.

$$R(\hat{\rho}, t) = S(\rho, t) - S(\hat{\rho}, t) + \Gamma(\hat{\rho}, t)$$  \hspace{1em} (1)

where $S(\rho, t) - S(\hat{\rho}, t)$ and $\Gamma(\hat{\rho}, t)$ denote the expected exploitation regret and the channel switching regret respectively.

1) Calculation of $\Gamma(\hat{\rho}, t)$: Let, $\gamma(t)$ denotes the total number of switches up to $t$, $\gamma(t) = \sum_{j=1}^{U} \gamma_j(t)$ where $\gamma_j(t)$ denotes the number of switches by user $j$.

$$\gamma_j(t) = \sum_{i=1}^{C} F(a_j(i), a_j(i+1))$$

where,

$$F(a, b) = \begin{cases} 1, & a \neq b \\ 0, & otherwise \end{cases}$$

Accordingly, we define switching cost as the number of packets that could have been transmitted within the time if it did not switch the channel. The switching cost defined in [4] is

switching cost = \frac{estimated packet transmission time}{switching delay}$

Assuming packet size $d$ and switching delay $\tau$, we formally define switching cost $c$ as

$$c = \frac{\tau * r}{d}$$  \hspace{1em} (2)

2) Calculation of $S(\hat{\rho}, t)$: In a distributed approach, there is a possibility that more than one SU select the same channel. As a result, no user receives any reward. In that case, the number of collision depends on the underlying collision mechanism. Assuming the collision channel model with no avoidance mechanism, the expected reward following a distributed approach $\hat{\rho}_{dist}$ defined in [3] is,

$$S(\hat{\rho}_{dist}, t) = r \sum_{i=1}^{C} \sum_{j=1}^{U} \mu_i \mathbb{E}[\Theta_{i,j}(t)]$$  \hspace{1em} (3)

where $\Theta_{i,j}(t)$ denotes the number of times user $j$ has the sole access on channel $i$ up to transmission slot $t$. The expected regret in distributed approach will be

$$R(\hat{\rho}_{dist}, t) = r \sum_{i=1}^{C} \sum_{j=1}^{U} \mu_i^2 - r \sum_{i=1}^{C} \sum_{j=1}^{U} \mu_i \mathbb{E}[\Theta_{i,j}(t)] + \gamma(t) * c$$  \hspace{1em} (4)

III. DISTRIBUTED LEARNING ALGORITHM

We formulate the channel selection problem as a multi-arm bandit problem with multiple play and switching cost, and develop a distributed block learning algorithm ($\rho^{D\text{BLA}}$).

A. Design of the Algorithm

At the beginning of each interval, the user picks a channel and compares it with the currently selected one. The channel selection process is periodic i.e. each channel is selected for comparison after $C$ blocks. Then it continues to use that channel until collision occurs. If a collision occurs, it randomly selects a channel among the top $U$ channels based on its current idle estimates of channels. Considering a user whose present selected channel is $p$ and the channel to be compared is $q$ at the end of a block, the user chooses channel $q$ over $p$ if

- $p$ is one of the top $U$ channels and $q$ is not. But upper confidence bound of $q$ is greater than the estimated idle rate of $p$.
- $q$ is one of the top $U$ channels but $p$ is not.

Each user selects a channel and updates its idle estimate and upper bound at each time slot throughout the time period.

1) Calculation of Block and Frame Size: We divide transmission slots into frames. Each frame (except frame 0 which consists of first $C$ slots) is then divided into multiple equal length blocks. Block length increases with frame number. Each block length is set equal to the frame number and frame length is set according to the following equation.

$$l_f = C * 2^{N_l}$$

where $l_f$ and $N_l$ denotes frame length and frame number.

2) Computation of Upper Confidence Bound of Channel: We use the interval estimation strategy [5] to compute the upper confidence bound of channels. The strategy works based on expected reward estimates of channels and confidence estimates of data. In our case, it is sensing data. It uses the standard statistical technique of constructing confidence interval estimates. If $x_{i,j}(t)$ be the number of successes arising from a series of Bernoulli trials ($\pi_{i,j}(t)$) with probability $p$, the upper bound of a $100(1 - \alpha)%$ confidence interval for $p$ can be approximated by $\beta_{i,j}(t)$.

$$\beta_{i,j}(t) = \frac{x_{i,j}(t) + Z_{\alpha/2}^2 / 2 \pi_{i,j}(t)}{1 + Z_{\alpha/2}^2 / 2 \pi_{i,j}(t)} + \frac{Z_{\alpha/2}^2}{4 \pi_{i,j}(t)}$$  \hspace{1em} (5)

Here, the parameter $Z_{\alpha/2}$ denotes the value that will be exceeded by the standard normal variable with probability $\alpha/2$. It controls the size of the confidence intervals. We follow the evaluation function stated in Equation 5 to compute the upper bound of each channel. In our algorithm, we set $\alpha = 2.5\%$ and the corresponding $Z_{\alpha/2}$ value is 1.96. In respect to our algorithm, $x_{i,j}(t)$ represents the number of times user $j$ sees channel $i$ idle up to transmission slot $t$. Initially each of the channels will have an upper bound of 1. As more channels are explored, more the bounds will be adjusted close to its exact value.

B. Non-stationary Environments

We extend our algorithm so that it can adapt in non-stationary environments. If the idle rate of the current channel in the current frame is not consistent with its overall idle rate, we increase the upper bound of every alternative channels to a certain value. In this way, if the selected channel status changes abruptly, a user will experience less availability of the current channel. This will lead to increase the upper bound
of alternative channels and thus probability of selecting a channel from these alternatives will increase. As soon as a user switches to another channel, it again starts acting accordingly.

IV. SIMULATION RESULTS

In this section, we present simulation results to evaluate the performance of our learning approach. We compare performance of our algorithm $\rho^{DBLA}$ with the learning approach $\rho^{RAND}$ proposed in [3]. We set up the identical simulation environment. To get the average result, we run the simulation for 10 times under the same conditions and each time our algorithm runs for 50,000 time slots. We also implement our centralized approach $\rho^{CBLA}$. This will provide a lower bound of regret achievable by SUs.

Performance Comparison with $\rho^{RAND}$: We present the comparison results of our algorithm (denoted as $\rho^{DBLA}$) and the algorithm (denoted as $\rho^{RAND}$) in [3]. The simulation result in Figure 1 shows the normalized regret for both the approaches with respect to number of transmission slots without considering switching cost. Clearly, our algorithm ensures faster convergence with less regret compared to [3]. We also plot our centralized algorithm ($\rho^{CBLA}$) to show the lower bound of regret.

We then compare the performance of both the algorithms with varying switching cost (see Figure 2). As expected, regret increases almost linearly with switching cost in $\rho^{RAND}$. On the other hand, regret increases slightly with switching cost in our algorithm $\rho^{DBLA}$.

Performance with Varying Number of Users: We compare the performance of $\rho^{RAND}$ and our $\rho^{DBLA}$ in terms of regret and number of channel switching. The results are presented in Figure 3(a) and Figure 3(b) respectively. In both figures (regret and channel switching), the rate of increase for $\rho^{RAND}$ is much higher than $\rho^{DBLA}$. In former algorithm, collision between users increase with the increase of number of users because in every slot, it tries to select the top channel and it may lead to more channel switching and more collision with other users. On the other hand, our algorithm performs better because it carefully decides about channel switching after each block.

Performance in Non-stationary Environments: To simulate a non-stationary environment, we assume that there are 4 secondary users and 9 channels in the network. Initially, channels’ idle rates follow a Bernoulli distribution with evenly spaced probability from 0.1 to 0.9. We run the simulation for 50,000 slots. After each 10,000 slots, we randomly assign each channel a different probability from 0.1 to 0.9. Figure 4 shows that our algorithm achieves logarithmic regret even in the dynamically changing channel states.

V. CONCLUSION & FUTURE WORK

In this paper, we propose an efficient distributed learning algorithm that minimizes the sum regret and channel switching cost, and ensures logarithmic regret. Our algorithm also adapts quickly in a dynamic environment and converges to an optimal solution. We compare our algorithm with existing approach and our algorithm achieves significantly lower regret and switching cost than existing ones. We will extend our work to support users’ different QoS requirements and channel bandwidth.

REFERENCES


