Learning the Best K-th Channel for QoS Provisioning in Cognitive Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

A learning strategy for distributed channel selection in Cognitive Radio networks is proposed. The goal of the learning is quality of service (QoS) provisioning by which competing secondary users cooperatively converge to their rank-optimal channels while channel availability statistics are initially unknown. By this convergence, collisions reaches zero since users eventually work on their own channels. The proposed learning strategy, $k^{th} - MAB$, is inspired from the Multi-Armed Bandit problem but it converges to the k^{th} best arm. The rank-optimal channel for each user\player is identified based on the user's QoS demands. We believe that under this learning and allocation policy, cognitive users get services proportional to their QoS level since evaluation results represent order optimality in terms of average throughput.

1 Introduction

Due to extensive need for wireless spectrum and the inefficiency in utilizing it, Cognitive Radio(CR) technology is emerged to allow unlicensed\secondary users(SU) for opportunistic access to the spectrum when licensed\primary users(PU) are not active. To take advantage of the possible empty spaces in the spectrum, SUs sense a part of the spectrum and use it for transmission *if it is found free*. Thus, it is crucial for SUs to make optimal decisions about which part of the spectrum to sense at different times. This gives rise to the trade-off between exploration: sensing new channels in the hope of obtaining better availability and exploitation: ensuring successful transmission in the current time.

When there are multiple SUs, there is a competition among SUs to access the channel with the best availability. Hence collision is likely since there is no explicit information exchange among SUs about their observations and channel selection strategy. Moreover, SUs demand diverse levels of quality of service (QoS) requirements proportional to their traffic importance. To provision these requirements, SUs should cooperatively find their own unique rank-optimal channels and work on them.

The goal of this paper is proposing a learning strategy for channel selection by which SUs estimate the rank of channels with respect to their availabilities through sensing samples. This strategy helps SU-i with rank k to allocate itself to an orthogonal channel with k^{th} highest availability, in a distributed manner. In this regard, Multi-Armed Bandit (MAB) problem for finding the best channel is reviewed in Section 2. $k^{th} - MAB$ channel-selection strategy for finding the k^{th} best arm, is proposed in Section 3. Performance evaluation and conclusion of the paper are covered in Section 4 and 5 respectively.

2 Multi-Armed Bandit (MAB) problem

MAB problem formulizes exploitation\exploaration trade-off for choosing the best arm by selecting one out of M possible arms in each trial $t \in 1, ..., T$. For the chosen arm i in trial t, reward $x_i^{(t)}$ is drawn from some fixed but unknown distributions $D_1, D_2, ..., D_M$ while the rewards for other arms excluding i, i.e. $i \in \{1, ..., M\} \backslash i$, are not revealed. The appropriate strategy for the MAB problem, pursues the goal of maximizing the total reward up to the observation period T, i.e. $\sum_{t=1}^{T} x_i^{(t)}$ where the upper expected total reward is obtained by the best distribution D_i. The difference between this upper bound and the achieved total reward is defined as *regret*.

The exploration\exploitation trade-off is reflected on one hand by the necessity for trying all arms and on the other hand by the regret suffered by trying a non-optimal arm. Too little exploration might make a sub-optimal alternative look better than the optimal one because of random fluctuations while too much exploration prevents the algorithm from playing the optimal often enough which also result in a large regret.

2.1 Upper confidence bound (UCB) algorithm

Upper Confidence Bound (UCB) algorithm for solving the MAB problem, chooses arm $i^{(t)}$ in trial t as: $i^{(t)} = \arg\max_{i \in M} (\bar{x}_i^{(t)} + \sqrt{\frac{\zeta log(t)}{n_i^{(t)}}})$

UCB calculates weight of arm i based on $\bar{x}_i^{(t)} + \sigma_i^{(t)}$ when this arm has distributions D_i and expected reward R_i. The first term, $\bar{x}_i^{(t)}$, is the current average reward which is an estimate for the true expected reward R_i. And the second term, $\sigma_i^{(t)}$, corresponds to the confidence interval that both the true and average rewards fall in with high probability, i.e. $\bar{x}_i^{(t)} - \sigma_i^{(t)} \leq R_i \leq \bar{x}_i^{(t)} + \sigma_i^{(t)}$. With UCB, we may say that a trial is an exploitation trial if an alternative is chosen since $\bar{x}_i^{(t)}$ is large and that is an exploration trial if $\sigma_i^{(t)}$ is large. Since $\sigma_i^{(t)}$ decreases rapidly with each choice of arm i, the number of exploration trial is limited. Thus the use of UCB automatically trades off between exploration and exploitations. An improved version, UCB-V, considers the effect of the empirical variance, is proposed in [1] and estimates the best arm in trial t as following where ζ and c are constant coefficients:

$$\arg\max_{i \in M} (\bar{x}_i^{(t)} + \sqrt{\frac{(\bar{x}_i^{(t)} - (\bar{x}_i^{(t)})^2)\zeta log(t)}{n_i^{(t)}}} + \frac{c.log(t)}{n_i^{(t)}})$$

3 MAB problem and K-th best arm

3.1 System model

We assume that time is slotted and a time-slot on channel i is occupied by PUs with Bernoulli distribution with parameter μ_i, i.e. $W_i \sim B(\mu_i)$. There is a set of U cognitive users grabbing free time-slots from M independent and orthogonal channels on the premise of not interfering the operation of licensed PUs. Also, cognitive user i has a prior information about its own unique rank, k, among the rest of U-1 users. Here, users should learn channel mean availabilities, μ, in a distributed manner and converge to an appropriate channel while on one hand they do not exchange information on their decisions and observations and on the other hand they implement the pre-allocated rankings. Note that we use two terms of time-slot and trial interchangeably. Thus, the optimal channel selection strategy for a SU-i is the one that narrows operation of user i on the channel with k^{th} highest mean availability.

At the beginning of time-slot t, user i selects a channel, e.g. channel j, and keeps the history of its selections on $T_{i,j}$. User i then senses the selected channel j to find if PU has occupied this slot or not and keeps the history of sensing results regarding to channel j in $X_{i,j}$. User i approximate the mean availability of channel j as $\hat{\mu_j} = \frac{X_{i,j}}{T_{i,j}}$ where $T_{i,j}$ indicates the number of times that channel j is selected by user i so far. In time slot *t+1*, channel selection strategy of user i exploits previous observations as the form of $\hat{\mu_j}, j \in 1, ..., M$ to pick a channel for sensing. Note that although $X_{i,j} = 0$ indicates that this slot is free of PU transmission but it does not guarantee that user j is the sole transmitter in this slot. In fact, collision is likely since multiple users may select a common channel. However, a proper learning strategy eventually confines the operation of user i on the channel with k^{th} highest mean availability and consequently collision probability reaches zero in the course of time. To estimate the achievable throughput, user i receives an acknowledgement on whether its transmission on channel j was successful($Z_{i,j} = 1$) or not ($Z_{i,j} = 0$).

3.2 Greedy distributed learning under pre-allocation (ρ^{PRE})

Authors in [2] have proposed a distributed learning under pre-allocation, ρ^{PRE}, as a modified version of the $\epsilon - greedy$ strategy for finding the K-th best channel. Their general idea is that a SU should do a lot of experiments by selecting different channels to estimate their availability ratios and eventually settles down in the appropriate one. In ρ^{PRE}, SU-i with rank k, selects a uniformly random channel with probability $\epsilon_n = min(1, \beta/n)$ and selects the channel with k^{th} highest sample mean with probability $1 - \epsilon_n$. It means that, there is a finite probability ϵ_n for user i to not select the channel according to its rank and instead finds an opportunity to explore other channels to find better estimation about their sample means. The value of β defines the trade-off between exploitation and exploration and so the efficiency of ρ^{PRE} is highly sensitive to the appropriate choice of β. In the next section we propose a new approach, $k^{th} - MAB$ to solve the problem of finding the k^{th} best channel which is more efficient than ρ^{PRE} as evaluated in Section 4.

3.3 UCB for ordring ($k^{th} - MAB$)

In this section, a decentralized policy called $k^{th} - MAB$ is constructed by which a cognitive user with rank *k* finds the best *k* channels in order, and converges to the one with k^{th} highest mean availability. Note that the higher the value of μ_is, $i \in \{1, ..., M\}$, the more available a channel is. Without loss of generality, from now we suppose that user i has the i^{th} highest rank and channel j has the j^{th} highest μ. With this assumption, user 1 and user 2 want to converge to channel 1 and 2 respectively. The basic idea is that user i selects the best i channels in a hypothetical frame structure consists of i time-slots. The formal explanation of this policy is summarized in in Table 1.

As an example, consider a case of three cognitive users, i.e. U=3, in which SU-1 and SU-3 have the highest and lowest priorities respectively. For SU-1, the problem is simplified as the common MAB problem in which a player wants to find the best arm (channel). Thus, SU-1 always applies UCB-V to efficiently learn and select the best channel. SU-2, works in frames of two time-slots since it wants to find the best two channels. For this, in odd time-slot of each frame, it applies UCB-V to find the best channel. In order to find the second best channel in an even time-slot of the frame, it applies UCB-V policy to the remaining M-1 channels after removing the channel considered as the best one in the odd time-slot. SU-3 wants to find the best three channels and finally converges to channel 3. For this, it works in a hypothetical frame of three time-slots and considers finding the best channel in the first time-slot. Then it applies UCB-V to M-1 channels remained from the first time-slot, to find the second best channel. Finally, it estimates the third best channel by applying UCB-V policy to the list of M-1 channels resulted from the case that the second best channel is estimated. At the end of the third time-slot, the current frame of SU-3 is completed and this user resumes its channel selection pattern from the next frame.

Since the goal of user i is convergence to the i^{th} best channel, it switches to channel j with $\hat{\mu_j} > \hat{\mu_i}$ only with probability $P_{switch} \sim B(min(1, \frac{5.0}{\sqrt{t}}))$. This allows user i to smoothly converge to the i^{th} best channel. For example, SU-2 tries to find the best channel in odd time-slots only with probability P_{switch}. Similarly, in trials t, t%3 != 0, SU-3 estimates the best and the second-best channels with probability P_{switch} and estimates the third best one with probability $1 - P_{switch}$.

Table 1: $k^{th} - MAB$ for user i with k^{th} highest rank

- Init:
  - Slecting each channel $j, j \in 1, ..., M$ once and updates $X_{i,j}$ and $T_{i,j}$s.
  - Set K sub-sequences with $\hat{\mu_j} = \frac{X_{i,j}}{T_{i,j}}$.
- At trial t = 0 ,..., T :
  - Calculates j = t % k and $P_{switch} \sim B(min(1, \frac{5.0}{\sqrt{t}}))$
  - if $P_{switch} = 0$:
    - Applying UCB-V to the k^{th} sub-sequence,
    - Selecting the best channel, say h, and updates μ_h.
  - else if $P_{switch} = 1$:
    - if j = 0 :
      - Applying UCB-V to the k^{th} sub-sequence.
      - Selecting the best channel, say h, and updates μ_h,
    - else if j = 1 :
      - Applying UCB-V to the *first sub-sequence*,
      - Selecting the best channel, say h, and updates μ_h,
      - Updating the *second sub-sequence* with *first sub-sequence*$\setminus$ h
    - else if j = 2 :
      - Applying UCB-V to the *second sub-sequence*,
      - Selecting the best channel, say h, and updates μ_h,
      - Updating the *third sub-sequence* with *second sub-sequence*$\setminus$ h
    - ...
    - else if j = (k-1) :
      - Applying UCB-V to the $k-1^{th}$ *sub-sequence*,
      - Selecting the best channel, say h, and updates μ_h,
      - Updating the K^{th} *sub-sequence* with the $(k-1)^{th}$ *sub-sequence*$\setminus$ h

4 Performance evaluation

We present simulations for comparing efficiency of two discussed schemes ρ^{PRE} and $k^{th} - MAB$. A set of M=5 orthogonal channels with mean availabilities characterized by the following Bernoulli distributions is avaibale, CH1$\sim B(.8), CH2 \sim B(.6), CH3 \sim B(.4), CH4 \sim B(.2), CH5 \sim B(.1)$.

Note that in a presence of a centralized arbitrator, SU-i with rank k is *centrally assigned* to work on the k^{th} best channel. This assignment of SUs to their rank-optimal orthogonal channels makes collision occurrence is unlikely and guarantees the optimum throughput. However, without such an optimal allocation, users may not choose their desired channels and face with collision. For SU-i, three criteria are introduced which represent how well learning policies work in comparison to the optimal case:

1) $regret_i^T = T.\mu_h - \sum_{j=1}^{M} Z_{i,j}$ indicates how much throughput is lost up to an observation period T where 'h' is the k^{th} best channel with mean availability μ_h.

2) $rank_{opt}^i = \frac{T_{i,h}}{\sum_{j=1}^{M} T_{i,j}}$ gives an estimate about efficiency of the learning strategy in bounding the operation of user i on its desired channel 'h'.

3) $\overline{Throughput_i} = \frac{\sum_{j=1}^{M} Z_{i,j}}{\sum_{j=1}^{M} T_{i,j}}$ estimates the percentage of channel selections that lead to a successful packet transmission. Under the presence of a centralized arbitrator, the average throughput for SU-i would be μ_h.

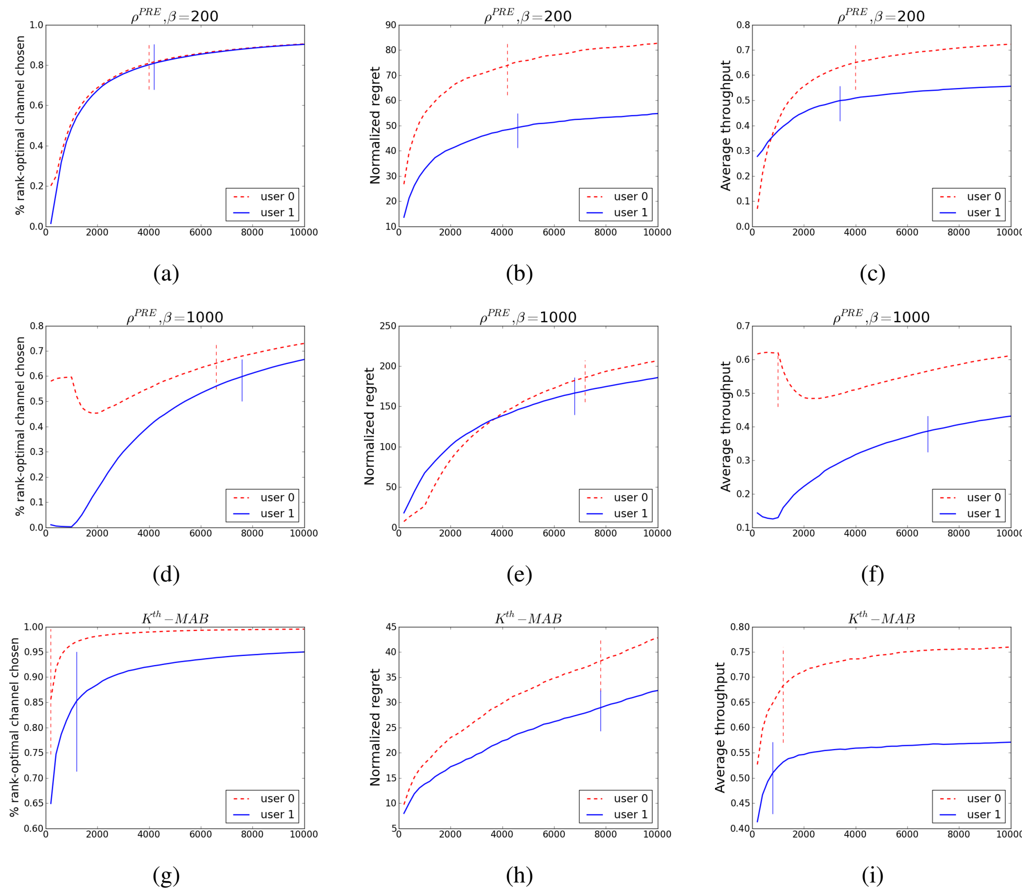

Figure 1: {(a)-(d)-(g)} is normalized regret, $\frac{regret_i^T}{log(T)}$, vs. T slots, {(b)-(e)-(h)} is $rank_{opt}^i$ vs. T slots, {(c)-(f)-(i)} is $\overline{Throughput_i}$ vs. T slots.

Fig. 1 and Fig. 2 represent the results for $k^{th} - MAB$, $\rho^{PRE}_{\beta=200}$ and $\rho^{PRE}_{\beta=1000}$ where two and three competitive SUs, i.e. U=2 and U=3, exist. SUs compete on a set of M=5 channels where SU-1 has the highest rank and its desired channel is CH1 with 80% mean availability and SU-3 has the lowest rank and its desired channel is CH3 with $\mu = 40\%$. To make the comparison easier, a vertical line corresponds to 0.9% of the final value, is added to each graph.

Worthwhile to mention that performance of ρ^{PRE} is evaluated under various values of β. It is empirically estimated that $\beta = 200$ gives the *best* results for $rank_{opt}^i$ and $\overline{Throughput_i}$. Therefore, results of $k^{th} - MAB$ are compared to the best empirical configuration of ρ^{PRE}. To emphasize that performance of ρ^{PRE} directly hinges on the configuration parameter β, results related to $\beta = 1000$ are also provided. In figures 1 and 2:

- Subfigures (a),(d) and (g) indicate how fast an applied learning strategy converges to the desired channel. Under $k^{th} - MAB$, after trial 2000 the desired channel is selected with probability higher than 80% while for $\rho^{PRE}_{\beta=200}$, similar situation happens after 4000 trials.
- Subfigures (b),(e) and (h) represent normalized regret up to time-slot T as $\frac{regret_i^T}{log(T)}$. Comparison of the results justifies that at each arbitrary trial T, SU-$i, i \in \{1, 2, 3\}$ suffers the least regret when it works based on $k^{th} - MAB$.
- Subfigures (c),(f) and (i) represent that under $k^{th} - MAB$ learning policy, $\overline{Throughput_i}$ reaches 0.9% of its highest value around trial T=2500 while similar results are obtained around trial T=4000 for the case of $\rho^{PRE}_{\beta=200}$.

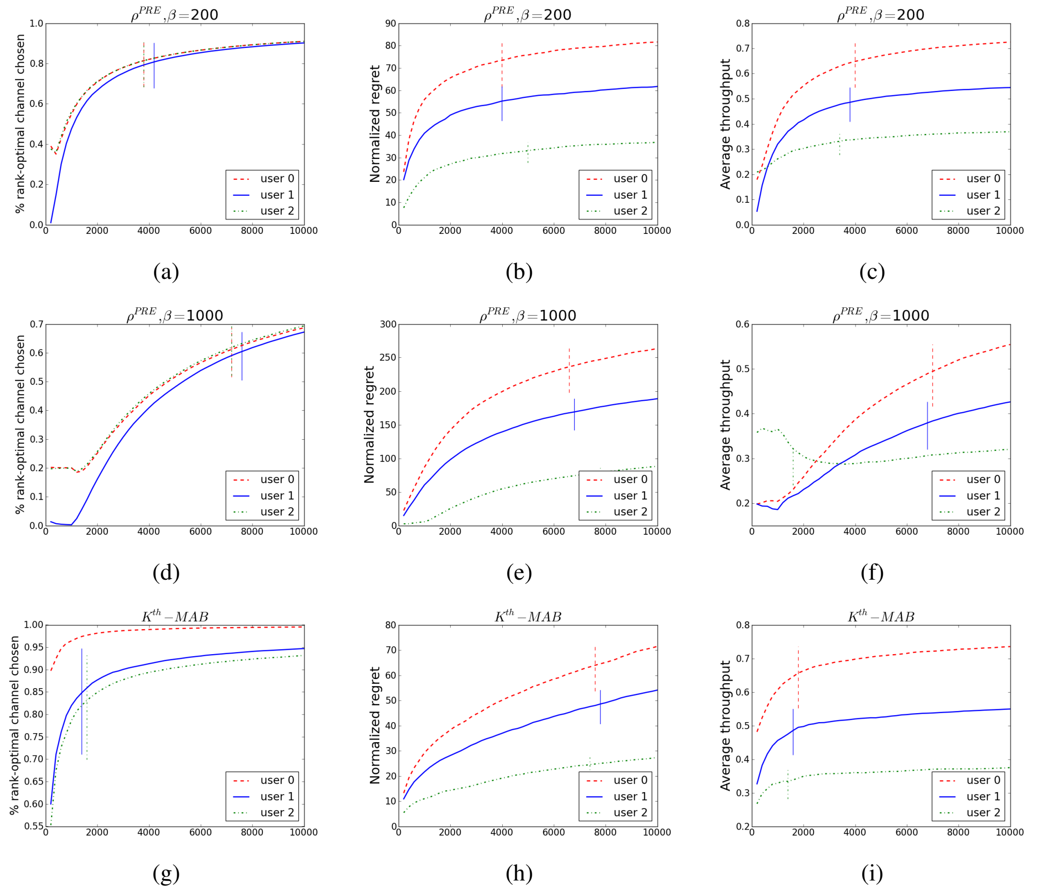

Figure 2: {(a)-(d)-(g)} is normalized regret, $\frac{regret_i^T}{log(T)}$, vs. T slots, {(b)-(e)-(h)} is $rank_{opt}^i$ vs. T slots, {(c)-(f)-(i)} is $\overline{Throughput_i}$ vs. T slots.

5 Conclusion

In this paper, we design a distributed learning policy by which SUs estimate channel statistics and cooperatively converge to their rank-optimal channels. Under this online learning strategy, achieved throughput for each SU would be proportional to the level of its QoS requirements. Simulation results represent that convergence rate to the desired channel is high and SUs get rank-based average throughput. We plan to extend this work for the non-greedy case in which SUs may have less traffic than channel availability of their desired channels. Thus, SUs can improve their average throughput by capturing leftover of channels with higher ranks.

References

[1] Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002) Finite-time Analysis of the Multiarmed Bandit Problem.*Journal of Machine Learning*.

[2] Anandkumar, A., Michael, N. & Tang, A. (2010) Opportunistic Spectrum Access with Multiple Users: Learning under Competition. *Proceedings of IEEE INFOCOM*.

[3] Liu, K. & Zhao, Q. (2010) Distributed learning in cognitive radio networks: Multi-armed bandit with distributed multiple players. *Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing*.