Route Aware Dynamic Channel Scheduling and Selection for Multi-Hop Cognitive Radio Networks

Erald Troja, Kenneth Ezirim, Suman Bhunia
Dept. of Comp. Science, The Graduate Center, CUNY
Email: etroja, kezirim, sbhunia@gc.cuny.edu

***Abstract*—The complexity of the Dynamic Spectrum Access (DSA) framework and risk of disruption due to primary-secondary spectrum etiquette pose serious link layer challenges in multi-hop cognitive radio networks (MH-CRNs). In order to properly harness the DSA potential, cognitive radio (CR) channel scheduling and channel selection should be treated in a cross-layer approach with the network layer. In this paper, we present and evaluate a novel route aware dynamic channel scheduling and selection approach (RADCSS) for MH-CRNs. Our proposed approach provides a conflict-free time slot transmission scheduling between neighbor to neighbor CR pairs while estimating the probability of primary user (PU) disruption for every mutual channel. Furthermore, in a cross-layer fashion, our proposed method maximizes neighbor to neighbor transmission time slot reuse in accordance with network layer route request. Through simulations, we demonstrate that our proposed method provides significant improvement on the average number of channel switching, effective channel utilization and average throughput.**

I. Introduction

In order for Cognitive Radio Networks (CRN) to reach their potential, it is necessary to investigate and understand *multi-hop* CRNs (MH-CRNs). Unlike infrastructure based networking, multi-hop point-to-point architecture can create wide-area CR back-haul networks where traffic can flow among peers directly using relay/forwarding via multiple hops resulting in higher capacity, ubiquitous connectivity and increased coverage. Currently however, there is little understanding on how such distributed architecture will operate in order to make the system feasible under DSA [1]. Accordingly, the issues in the design of *MH-CRNs* must be better studied for the concept of CRNs to reach its full potential.

MH-CRNs are flat routed networks where each CR node may act as a source or destination. Unlike a tiered network, there is no notion of specialized "gateway" or "router" nodes. Each CR node may have different streams of data passing through them. In a MH-CRN, a potential hop between two nodes is not guaranteed for the duration of the route, and is subject to breakage due to PU arrival probabilities and patterns. Thus, the notion of spectrum availability and spectrum mobility is tightly coupled with the PU arrival probabilities and patterns. It is imperative that PU arrival probability and pattern be a considerate driving factor towards the formulation of a robust channel selection scheme for the MH-CRNs. Consequently, in the lack of a centralized fusion center, CR nodes must learn to classify neighboring node(s) common channels according to PU disruption probability in order to elevate the link reliability.

A disruptive PU transmission necessitates either a mandatory channel evacuation at the link layer, or bypassing of the affected region via re-routing at the network layer. One of the most time consuming tasks for a CR node is coordination during channel evacuation and hand-off. Channel evacuation and switching has an adverse effect on the data rate passing over the affected link. For the duration that the new channel negotiation is taking place through the channel evacuation procedures, any data streams being routed at the link layer would be severely affected. If no channels could be negotiated during channel evacuation procedure, a longer route might be negotiated by the network layer which could introduce extra transit delay. Therefore, the problem of robust channel allocation via a thorough and comprehensive channel scheduling and selection is very important.

Channel selection for MH-CRN approaches have been proposed recently in [2] [3] [4]. [2] and [4] propose a Multi-Agent Reinforcement Learning approach which uses value functions to evaluate the desirability of choosing different transmission parameters. The proposed approaches work in isolation from the network layer and do not address the cross-layer functionality that MH-CRN require. [3] proposes a network architecture for cognitive wireless mesh networks which offers cross-layer functionality and is based on reinforcement learning. Through autonomous reconfiguration, the management system of [3] is shown to perform better than the original AODV protocol, however comparison with other proposed work has not been discussed. Though it is a very important requirement at the link layer, a robust channel selection approach is of low importance if is considered in isolation from the network layer.

With regard to the previous proposed research and the desire to design a comprehensive cross-channel approach, the contribution of our paper is threefold. We first provide a distributed channel scheduling method which arranges the pairs of neighboring CR nodes in a conflict free configuration. Unlike the channel allocation method devised on CR node contention [5], our distributed channel scheduling method is based on notion of dynamic time division and reuse such that the link layer can accommodate different priorities of network layer traffic. We use the term *fairness* to represent the minimum possible time slot(s) that a pair of CR neighbors should be assigned in order to allow for a) a resilient network and b) assist upper level network layer to route low priority traffic. Secondly, extending on the concept of dynamic time division, we assist upper level network layer via a route-centric method of dynamically *maximizing* the time slot reuse factor. Unlike other proposed research, where the maximization of the assigned number of channels is handled in a node-centric manner through distributed graph coloring solutions [6], our time slot reuse *maximization* is dynamically enhanced by the network layer. As a result, the proposed approach provides better dynamic utilization of spectrum opportunity (SOP) and offers high effective channel utilization in various PU arrival patterns. Thirdly, we propose a channel selection approach which is based on each individual CR node's learned probability of channel success rate. The probability of channel success rate evolves through interactions with the environment. The probabilistic channel selection approach evolves and improves during each neighbor to neighbor communication.

The rest of the paper is organized as follows. We provide the RADCSS framework and system model subsequently. Performance and result analysis through simulations are presented in Section III. We conclude the paper with future research directions in Section IV.

II. RADCSS Framework

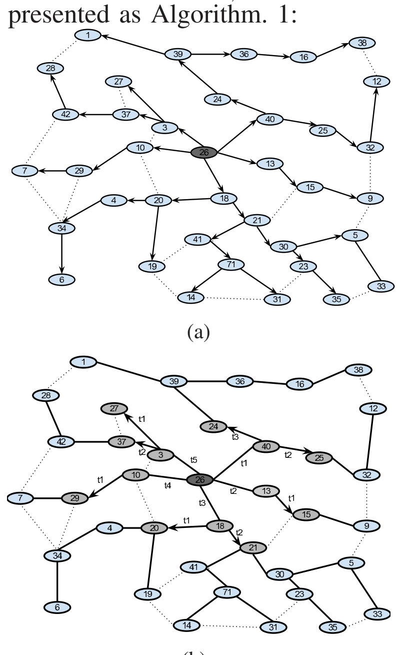
Our approach is envisioned to be deployed on a stationary MH-CRN where SUs are static nodes and where each CR holds a unique identifier. SOP fluctuation is affected only by PU transmissions. We assume that CRs use directional antenna for transmission purpose and omni-directional antenna for receiving purpose. Conclusions from previous research [7] recommend the use of directional antenna model, which can largely reduce radio interference, and thereby improve the utilization of the wireless medium. Furthermore, the channel reuse rate is improved when utilizing the directional antenna model. In our model we assume that CR nodes are able to communicate with neighboring nodes through an out of band Common Control Channel (CCC) by incurring a minimal overhead penalty.

A. Distributed Channel Scheduling Component

In our system model, PUs affect the network unevenly and cause certain channels to be unavailable. The number of available channels is assumed to be non uniformly available to the nodes in the network. In other words, clusters of nodes which operate on different space (location), time and frequency are subject to different SOPs.

The fairness criterion of the scheduling component is met by transmission time division and transmission time slot reuse. We aim to divide time in as little time slots as possible. During each time slot, a set of CR pairs are assigned transmission rights. An equal and fair amount of transmission time is allowed to each CR pair in order to meet the fairness criterion. Transmission can happen only during the alloted time slot, following certain conflict-free requirement (we will elaborate on this more later).

During network initialization nodes discover neighbors and run a distributed minimum spanning tree (DMST) algorithm. Note that, this needs to be run only once, during initialization time[1]. We build and maintain the DMST through the Nearest Neighbor Tree (NNT) [8] which has a time complexity of $O(D(G) + L(G, w))$, where $D(G)$ is the diameter of the graph, and $L(G, w)$ is the local shortest path diameter. It has a message overhead complexity of the order of $O(|E| + nlogn)$ where E is the number of edges, and n is the number of CR nodes. DMST produces a rooted tree where the root node has the highest degree d. Time is divided into d + 1 time slots. During DMST formation nodes learn which edges create loops. Let G = (V,E) be a connected undirected graph. As shown in Fig. 1(a), DMST partitions the set E into E' and L $|$ E' is the set of all direct parent to child edges and L is the set of all looping edges. We define $\mathbb{N}_v$ the set of vertices $| \forall v \in V \; \exists v' \wedge (v, v') \in E' \vee (v, v') \in L$. We define $\mathbb{T}_v^F$ as the set of free time slots of vertex v and $\mathbb{T}_v^R$ as the reserved time slot set of vertex v. $\mathbb{T}_v^F \cup \mathbb{T}_v^R = \emptyset$. The set of mutual free time slots for (v, v') is denoted as $\mathbb{T}_{v,v'}^F$. The *fairness* component algorithm is presented as Algorithm. 1:



(a)

(b)

Fig. 1. a)Sample tree structure produced by the DMST. Arrows also denote the propagation of time slot assignment rights. b) Distributed time slot assignment by nodes 40, 13, 18, 10, 3.

B. Routing-aware enhanced time slot reuse maximization

While the fairness component allows for high network resiliency and guaranteed performance, complete time slot assignment is not fully done. Some available time slots are not assigned onto links which could use them without conflict. One approach we could take to utilize the rest of the available time slots could be from a local CR node perspective. However, that would lead to an inefficient arrangement of the potential time slots. We aim to optimize active route throughput. Accordingly, we opt for the route-aware enhanced method instead, in order to maximize assigned time slots

[1]Note that, any efficient DMST algorithm could be chosen, which is beyond the scope of the paper.

Algorithm 1 Channel Scheduling Algorithm

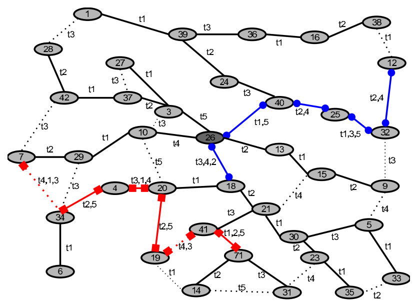
```
 1: procedure FAIRNESS(r)          ▷ r is root, follow solid edges
 2:     for all  $v \in \mathbb{N} \mid (r, v) \in E'$  do
 3:         if  $\mathbb{T}_r^F \cap \mathbb{T}_v^F \neq \emptyset$ then
 4:             pick any t from $\mathbb{T}_r^F \cap \mathbb{T}_v^F$
 5:             $\mathbb{T}_r^R = t \cup \mathbb{T}_r^R$
 6:             $\mathbb{T}_v^R = t \cup \mathbb{T}_v^R$
 7:             $\mathbb{T}_r^F = \mathbb{T}_r^F \setminus t$
 8:             $\mathbb{T}_v^F = \mathbb{T}_v^F \setminus t$
 9:         end if
10:     end for
11:     for each child of r run in parallel FAIRNESS (child)
12: end procedure
13: procedure FAIRNESS(r)          ▷ r is root, follow loops
14:     for all  $v \in \mathbb{N} \mid (r, v) \in L$  do
15:         if  $\mathbb{T}_r^F \cap \mathbb{T}_v^F \neq \emptyset$ then
16:             pick any t from $\mathbb{T}_r^F \cap \mathbb{T}_v^F$
17:             $\mathbb{T}_r^R = t \cup \mathbb{T}_r^R$
18:             $\mathbb{T}_v^R = t \cup \mathbb{T}_v^R$
19:             $\mathbb{T}_r^F = \mathbb{T}_r^F \setminus t$
20:             $\mathbb{T}_v^F = \mathbb{T}_v^F \setminus t$
21:         end if
22:     end for
23:     for each child of r run in parallel FAIRNESS (child)
24: end procedure
```

only on active routes, while guaranteeing a minimum system performance.

Maximization of time slot re-usage is attempted as follows. During the DMST formation, the root node is chosen to be the highest degree node d, and such knowledge is propagated to each node. CR nodes attempt to maximize time slot re-usage on active network hops through a "borrowing" process. Time slots assigned to neighboring CR nodes will be "borrowed" so long as there is no active routes passing along any of the lending nodes. Borrower nodes retain and use the extra time slots, but relinquish them immediately to the lenders if the lenders need to communicate with neighboring nodes for which the time slot was originally reserved for. We denote as $(v, v')^t$ as the edge (v, v') active on time slot t and define the set $\mathbb{T}_v^A \subseteq \mathbb{T}_v^R \mid t$ is an active time slot. We define the $\mathbb{T}_v^B \subseteq \mathbb{T}_v^R \mid t$ can be borrowed. Fig. 2 shows *maximization* component time assignment. The algorithm for it is shown as Algorithm. 2.

C. Channel selection component

In Section II-A and 2, transmission scheduling is performed through time partitioning and dynamic routing-aware time slot assignments and reuse. However, CR node pairs must communicate over a mutually available channel. In this subsection we explain how a pair of CR nodes choose the best channel out of a possible N mutually available channels sensed by the PHY layer. We utilize Q-learning, which is a Reinforcement Learning(RL) based algorithm [9], as the basis of our channel selection scheme. In RL, an agent acquires the target knowledge through trial and error interactive actions with the environment. Each action is associated with a positive



(a)

Fig. 2. Two sample routes, after the maximization component is run

Algorithm 2 Channel Scheduling Algorithm

```
 1: procedure MAXIMIZATION(s)          ▷ s is the source route
 2:     as per a routing policy choose next hop p  | $(s, p)^t$
 3:     while  P $\neq$ Destination do
 4:         for any t $\in (T_s^R \setminus T_s^A) \cap (T_p^R \setminus T_p^A)$ do
 5:             $T_s^A = t' \cup T_s^A$
 6:             $T_p^A = t' \cup T_p^A$
 7:             $T_s^B = t' \cup T_s^B$
 8:             $T_s^B = t' \cup T_s^B$
 9:         end for
10:         MAXIMIZATION(p)
11:     end while
12:     reverse source s and destination node d
13:     GOTO 1
14: end procedure
```

or negative reward, and over time, the agent builds up an optimal action taking policy which maximizes the overall reward. Our RL-enhanced channel selection is defined by a Markov Decision Process (MDP). Node x has N channels in common with receiving node y. In our model, there exist a set of states S, a set of actions A and set of rewards R. Time is assumed to be discretely divided into transmission slots (not to be confused with the time slot reuse notion of channel scheduling component). A state $s \in S$ denotes the channel at the t-1 transmission slot. $|S| = |N|$ since any of the possible channels could have been used during the t-1 transmission slot. Actions denote the set of channels that are possible to use at the current transmission slot. $|A| = |N|$ since we can choose to use any of the possible N channels by either staying on the current channel or by switching into a new one. $|R|$ is defined as the set of rewards. In our model, a transition from a state to another is performed according to the transition function which we explain shortly.

Formally we denote the environment via the tuple $< s_t, a_t, r_t, s'_{t+1} >$ meaning that at time slot t, a CR which is on state $s \in S$ might perform action $a \in A$ and will receive a reward or penalty leading to a new state s' according to some transition function. Link layer confirmation of successful transmission gain a +1 reward whereas failed transmissions are associated with a -5 penalty. Q-values are represented as a 2 dimensional array.

Transition for a tuple $< s, a, r, s' >$ is dictated by a state-distribution function. Actions vary between *explore* (i.e choose an action randomly) or *exploit* (i.e choose the action that will lead to the next state that yields the maximum cumulative reward). Which action is taken depends on an ε-greedy strategy with ε=[0-1] where we explore with $\mathbb{P} = 1-\varepsilon$ and we exploit with $\mathbb{P} = \varepsilon$. An action to exploit is defined as the action which yields the maximum Q-value in the next state: $a_t = \max_{a_t \in A_t} Q_{t+1}(s_t, a_t)$. Q-values are updated as follows: $Q(s_{t+1}, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma \max_{a_t \in A_t} Q_{t+1}(s_t, a_t) - Q(s_t, a_t)]$ such that $0 \leq \alpha \leq 1$ is the speed of learning, and $0 \leq \gamma \leq 1$ is the discount factor for future rewards. Q-learning will adjust its policy π of Q-value updates through interactions with the environment and will converge to the optimal policy π^*.

III. Performance Analysis

A. Simulation Setup

For performance analysis of the proposed mechanisms, we conducted extensive simulation experiments with various networks. We ran each simulation for 10.000 time units. Each simulation was averaged over a large number of runs. Each link (pair of neighboring nodes) is capable of scanning up to 20 channels and can use one channel for transmission. At network initialization, each channel is injected with randomized PU arrival probability. The goal of the performance analysis is to identify the link layer and network layer benefits of our proposed RADCSS method as a function of varying PU arrival probability and varying network load.

B. Simulation Results

As a measurement of the joint dynamic channel scheduling and RL based channel selection performance, we compared it against a) random channel selection scheme b) greedy channel selection opting for the channel with highest SINR. Research in [10] [11] shows that channel switching is a very costly procedure, the minimization of which should be the goal of a channel scheduling and selection method. SUs will switch channels if the SINR deteriorates and causes packet drop. Most importantly, in a DSA environment, upon PU arrival, SUs must evacuate the active transmission channel and must switch to a new one.

Fig. 3 represents the amount of channel switches per node as a function of PU arrival probability. A point on the X-axis represents the highest PU arrival probability for a channel for a particular simulation run. In the X-axis, a PU arrival probability of 0.4 means that we inject all channels with PU arrival probability randomly between 0 and 0.4. Y-axis reveals that as the PU arrival probability increases, the amount of channel switching per node increases in all of the compared methods. However, our proposed method, provides significant improvement by reducing the amount of channel switches per link. We can see that greedy channel selection method performs well when PU arrival probability is very low but deteriorates with the increase in PU arrival probability. The reason behind such behavior is that the greedy on SINR approach does not take into account the PU arrival probability. Channels with highest SINR might happen to be associated with very high PU arrival rates, and the greedy on SINR does not take it into account.

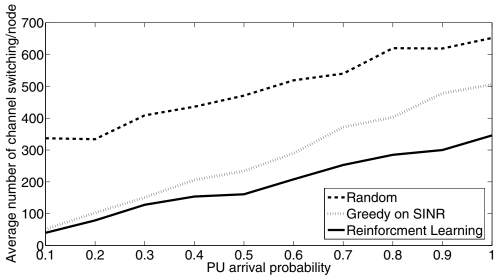


Fig. 3. Average number of channel switching per node

Fig. 4 represents the effective resource utilization. We calculate it from each node's perspective, as the ratio of the actual successful packets sent vs total attempted packets to be sent. We can see that with the increase of PU arrival probability, resource utilization deteriorates in all of the three compared methods. However, our proposed method of joint channel scheduling and selection is more resilient towards PU arrival disruptions. The reason behind it is due to the cognition gained by our RL based approach. Through interaction with the environment, channels which might have high SINR but high PU probability are avoided accordingly by our approach. We can further see that, as the PU arrival probability increases, the advantage of using our method becomes clearer.

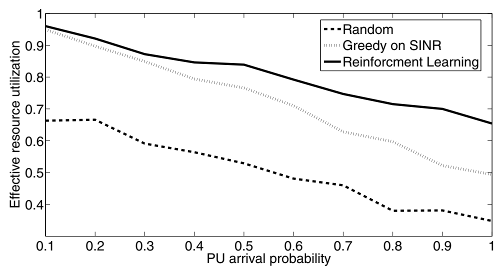


Fig. 4. Effective resource utilization rate

The next performance index we investigate is how fast nodes learn about the environment to adapt with it. To evaluate this performance we run our simulation for a long time and after every 100 time units we observe the number of channel switches in that period. PU arrival probability was randomly varied between 0 and 1 in this part of the simulation. The results from this simulation were averaged over 1000 runs. Greedy and random approach offer no possibility to interact with the environment and improve their channel quality estimates. Our proposed RL method improves channel quality estimation and which helps to reduce the average number of channel switches per node. Our RL based approach converges to the optimal number of channel switches at the mark of 300 time units.

In our simulation, balance between exploration vs exploitation is controlled from the ε value. We investigate what value of ε provides the best average number of packets received (throughput) as a function of varying PU probability. The value of ε as 0.9 means that we choose to exploit 90% of the time(i.e by choosing the channel with the highest state value) and with a 10% chance of exploring (i.e choosing a channel randomly). In this part of the simulation, we run every simulation a high amount of times with varying ε values. We ran it for 2 different

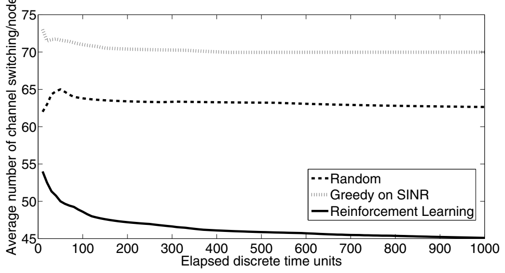


Fig. 5. Convergence to optimal channel switching rate

set of PU arrival probability, namely low (0-0.33), and high (0.67-1). We can observe that for low PU arrival probability ε=0.6 gives highest average packets received per node. For high PU arrival probability, ε=0.8 yields the highest average packets received per node. The intuition on such findings is that at higher PU arrival probabilities, nodes behave more *pessimistically* and opt for safekeeping of the best channel. Conversely at lower PU arrival probabilities, nodes act more *optimistically* and attempt to explore newer actions.

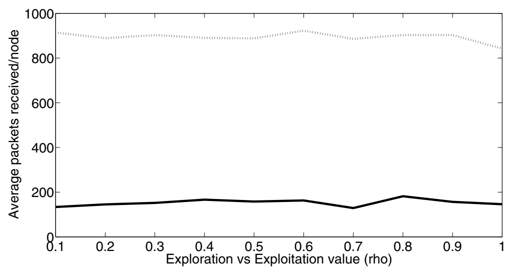


Fig. 6.

Lastly we compare the performance of network with fairness constraint time slot assignment vs the maximization of time slots by "borrowing" . For validating this experiment, we did not use static routes as in previous parts of the simulation. Instead, we generated routes randomly. The X-axis represents the percentage of active links vs total links in the network. For every generated set, we ran the experiment 1000 times and we generated paths 100 times and then we took the average in order to gain good observation. PU arrival probability for all channels was initialized randomly between 0-1. We measured the number of successfully transmitted packets per each node. Two things can be observed. First, irrespective of the number of paths in the network the fairness constraint time slot assignment is giving a constant, guaranteed performance. Second, when network load is low (i.e low number of active paths) links can borrow more time slots and maximize throughput better. As the network load increases (i.e higher number of active routes) nodes cannot borrow as many time slots. As a result effective throughput degrades. In the given network, total possible time slot equals 5. We can see that even at full network load, some nodes can use 2 slots/edge if their incoming/outgoing degree is 2. That is why on average we are getting better performance than the fairness constraint time slot assignment, even at full network load.

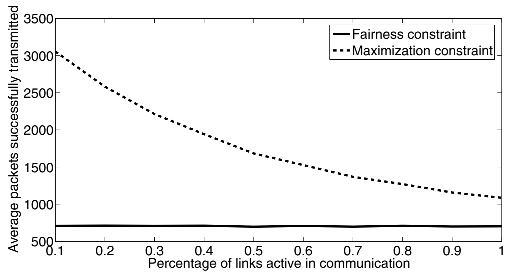


Fig. 7. Average packets successfully transmitted as a function of the 2 proposed time slot assignment components

IV. Conclusion

We have extensively evaluated the performance of our proposed model. From the results, it is very clear that our proposed method provides improvement of network performance by learning the environment and efficiently scheduling time slots and selecting channels. In our simulation we experienced less channel switching, higher resource utilization, faster convergence with the dynamic environment. We conclude that RADCSS can be suitable for a MH-CRN in providing reliable service quality. It can be extended to include the high dynamic behavior of the physical layer.

References

[1] I. Akyildiz, W. Lee, M. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks*, vol. 50, no. 13, pp. 2127–2159, 2006.

[2] C. Wu, K. Chowdhury, M. Di Felice, and W. Meleis, "Spectrum management of cognitive radio using multi-agent reinforcement learning," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Industry track*. International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 1705–1712.

[3] M. Lee, D. Marconett, X. Ye, and S. Yoo, "Cognitive network management with reinforcement learning for wireless mesh networks," *IP Operations and Management*, pp. 168–179, 2007.

[4] K. Yau, P. Komisarczuk, and P. Teal, "Applications of reinforcement learning to cognitive radio networks," in *Communications Workshops (ICC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1–6.

[5] C. Cordeiro and K. Challapali, "C-mac: A cognitive mac protocol for multi-channel wireless networks," in *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, april 2007, pp. 147 –157.

[6] M. Steenstrup, "Opportunistic use of radio-frequency spectrum: a network perspective," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*. IEEE, 2005, pp. 638–641.

[7] S. Yi, Y. Pei, and S. Kalyanaraman, "On the capacity improvement of ad hoc wireless networks using directional antennas," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, ser. MobiHoc '03. New York, NY, USA: ACM, 2003, pp. 108–116. [Online]. Available: http://doi.acm.org/10.1145/778415.778429

[8] M. Khan and G. Pandurangan, "A fast distributed approximation algorithm for minimum spanning trees," *Distributed Computing*, pp. 355–369, 2006.

[9] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[10] S. Krishnamurthy, M. Thoppian, S. Venkatesan, and R. Prakash, "Control channel based mac-layer configuration, routing and situation awareness for cognitive radio networks," in *Military Communications Conference, 2005. MILCOM 2005. IEEE*. IEEE, 2005, pp. 455–460.

[11] P. Pawelczak, R. Venkatesha Prasad, L. Xia, and I. Niemegeers, "Cognitive radio emergency networks-requirements and design," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*. IEEE, 2005, pp. 601–606.