An Adaptive Multiple Rendezvous Control Channel for Cognitive Radio Wireless Ad Hoc Networks

Claudia Cormio
Telematics Lab, Dipartimento di Elettronica ed Elettrotecnica
Politecnico di Bari, Italy
c.cormio@poliba.it

Kaushik. R. Chowdhury
Electrical and Computer Engineering Department
Northeastern University, Boston MA 02115, US
krc@ece.neu.edu

***Abstract*—Cognitive Radio (CR) technology enables the opportunistic use of the portions of the licensed spectrum by the CR users, while ensuring that the performance of the primary users (PUs) of the licensed bands is not affected. The spectrum is sensed locally by the CR users, and a specific channel that is acceptable to both the end nodes of the communication link is chosen. However, this necessitates a common control channel (CCC) for exchanging the sensing information and reserving the channel before actual data transfer. The design of an adaptive frequency hopping CCC, called as adaptive multiple rendezvous control channel (AMRCC), is proposed for CR ad hoc networks in this work, that is scalable, and allows continuous connectivity between the CR users under dynamic PU activity. The contribution made in this paper is twofold: (i) A frequency hopping scheme is proposed that allows altering the hopping sequence based on the PU activity in the channels, and (ii) A simple and low-overhead procedure is developed to aid new node-join and leave events. Simulation results reveal that our solution achieves better performance than the other classic CCC solutions in terms of the time to coordinate a feasible channel for communication and the resulting throughput specifically in CR ad hoc networks.**

***Keywords*-Cognitive Radio; Opportunistic Spectrum Access; Common Control Channel; Media Access Control;**

I. Introduction

Cognitive radio networks allow the CR users to share the wireless channel with licensed or primary users (PU) of the spectrum in an opportunistic manner. Such a radio is equipped with dynamic spectrum access capability, thereby allowing it to identify portions of the spectrum that are currently available for transmission. This technology is envisaged to solve the problem of spectrum scarcity in the unlicensed 2.4 GHz ISM bands, and the inefficient spectrum utilization that exists in some licensed frequency bands.

One of the key considerations in using CR technology is ensuring that the PU transmission is always protected. For this, it is essential to exchange the information pertaining to the spectrum availability between a given node pair before starting data transmission. This functionality is generally provided at the link layer, and several works have addressed the problems of medium access control (MAC) for CR networks [5]. The MAC protocols assume either a dedicated control channel with reserved frequencies within the licensed band, or the explicit use of the unlicensed band to exchange the control information [1]. Message exchanges on such a common control channel may be related to (i) channel access and contention, (ii) neighbor discovery or (iii) spectrum management. While the first two approaches are common to the classical wireless ad hoc networks, there are several spectrum related functions that are unique to CR networks. Specifically, the spectrum sensing information has to be exchanged over a channel that is always available before data can be transmitted. Moreover, once a PU reclaims the spectrum, the new channel must be quickly determined for the affected link, and this link recovery information cannot be transmitted over the previously used spectrum. In either case, a CCC is used to facilitate the continuous operation of the CR users without any disruption. In this work, we propose a frequency shifting design for a CCC, AMRCC, that can adaptively change the sequence of the channels based on the sensing results. Our design is motivated by the fact that strict synchronization is difficult to achieve in an ad hoc network, and the CCC design must be scalable.

The research community has proposed several schemes for setting up and maintenance of a reliable CCC in cognitive radio wireless networks, addressing the challenges posed by (i) CCC saturation [8], (ii) robustness to PU activity [1], (iii) jamming attack [9], and (iv) control channel coverage [1]. We describe some of the related works in the following.

The SYN-MAC is a slotted protocol that integrates the control channel access with the regular available data channel [7]. Each time slot is dedicated to a channel which can be used both for control and data exchange. This also addresses the problems of control channel saturation and jamming. The main drawbacks are that it does not guarantee protection from PU activity, and provides disrupted control channel coverage whenever a PU occupies the control slot. The SRAC protocol addresses the control channel issue by exploiting the techniques of dynamic channelization and cross-channel communication [9]. However, this work is limited from the viewpoint of network-wide CCC coverage, and cannot easily recover from a sudden PU appearance.

The C-MAC protocol is characterized by in-band signaling as it integrates the CCC in the superframe structure [4].

978-1-4244-5328-3/10/$26.00 ©2010 IEEE

It guarantees flexible control channel coverage and it is robust to PU activity through the use of backup channels. The main drawbacks of this work are the high control overhead due to beacon exchange and requirement of strict synchronization. The necessity for synchronization can be addressed in networks with special topologies, such as a cluster. Such a solution is presented in [2], where the clusterhead assists in the choice of the channel. Moreover, several different clusters representing different CCCs may be integrated over time. This work may be limited in application as it assumes special topology formation, and is also affected in dynamically changing topologies.

A multiple rendezvous control channel (MRCC) is proposed in [6]. Here, the nodes hop over multiple channels till a common channel to the pair (rendezvous channel) is reached. The main problem of this work is that the hopping sequences between a node pair is static once the rendezvous condition is reached. If the PU activity is detected on a channel, it is simply removed from the schedule. For dynamically changing PU activity, this may lead to inefficient hopping schedules. Moreover, there is no bound on the time taken to achieve the rendezvous (TTR).

The adaptive multiple rendezvous control channel (AMRCC) scheme proposed in this paper maximally spreads control signalling and data transmission among channels compared to [6]. In fact, the adaptive MRCC scheme builds the channel hopping sequences based on sensing information in order to hop across the different channels by minimizing the interference to licensed users. The main differences of our work with [6] is that the sequences are chosen adaptively to combat the problem of PU interference. Similar to Bluetooth [3], the hopping sequences are built in such a way that the channels with minimum interference to other devices occur a higher number of times than the others. Additionally, the start and stop times of each slot do not have to be rigidly synchronized, as seen in [6].

In Section II the motivations of our work are given and our proposed AMRCC control channel design is explained. Section III gives a detailed performance evaluation of the AMRCC scheme and, finally, we conclude in Section IV.

II. Adaptive Multiple Rendezvous Control Channel

In this Section we propose our Adaptive MRCC (AMRCC) scheme for CR ad hoc networks, which achieves high performance by dynamically adapting the hopping sequences to the detected licensed activity. Our proposed approach has the advantages of being completely distributed, not requiring strict synchronization, and using a single radio interface.

A. Motivation for an Adaptive MRCC

We chose to extend the MRCC solution for our approach because it may be used to maximally spread the control and data packets among the channels of the licensed spectrum, thereby making the CCC robust to the unpredictable PU activity. Furthermore, it guarantees high throughput by exploiting spectrum holes in an efficient way with minimum delay.

Classic multiple rendezvous control channel schemes have several drawbacks which make them infeasible in cognitive radio networks. They are (i) strict synchronization, (ii) low scalability and (iii) low robustness to PU activity changes. We explain these factors as follows:

- *S*trict Synchronization: Most hopping sequences require the users to hop together, in which the precise start times of the hops must be synchronized. This is difficult to achieve and maintain in distributed environments. In AMRCC, the channel switching times may not be the same for all the users during the usual network operation. The synchronization is only enforced after the rendezvous, i.e. after both the sender and the receiver meet on a common channel during the hopping process. This is when the SYNC packet is exchanged between the users. As opposed to this, the classical MRCC approach requires a strict synchronization, starting from the network initialization phase.
- *L*ow Scalability: In the classical network, every time a new node appears, it should exchange the sequence seed with all its neighbors. This action is necessary in order to be able to follow the hopping pattern of any intended receiver in the neighborhood. Our adaptive scheme, on the contrary, results in a significantly reduced overhead as the seed exchange only occurs at the rendezvous channel. The sender and its intended receiver hop in an independent and asynchronous manner over their own sequences, which are completely unknown to each other before the rendezvous. Thus, this requires fewer timer settings and less schedule maintenance than in the classical MRCC.
- *L*ow Robustness to PU Activity Changes: The classical MRCC solutions are not suitable for cognitive radio networks as they are not able to adapt to the dynamic behavior of the primary network and, consequently, avoid interference to the PUs. Previously proposed solutions assume that after the rendezvous of a node pair over a given channel, they perform the entire data packet exchange over that channel. There is no mechanism to vacate the spectrum band due to the appearance of a PU. In our approach, the hopping sequence is modified on the basis of the latest sensing results by favoring the data exchange over the channels with the lowest PU activity.

B. AMRCC Scheme Description

In this Section, we introduce our Adaptive MRCC (AMRCC) scheme, which aims at improving the time to rendezvous (TTR) and the overall network performance by

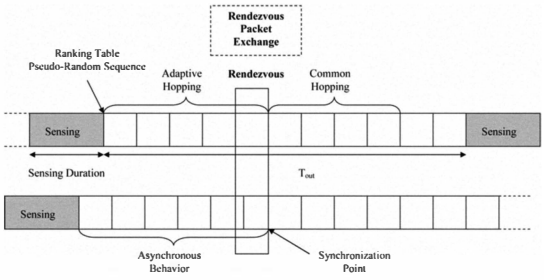

Figure 1. Overview of the protocol behavior.

overcoming the performance issues of the classical MRCCs discussed in Section II. The goal of our approach is to achieve a lower time to rendezvous and higher throughput, which means better exploitation of the spectrum holes in the PU network, compared to the classical solutions.

1) Overview: In Figure 1 the entire protocol behavior within a period of the sensing cycle is shown. A node performs spectrum sensing periodically after a time out T_{out} and the period of the sensing cycle is assumed to be equal to the sum of the sensing duration and the time out period. The sensing results are used to build a ranking table of the available channels based on the PU activity detected on each channel. The node then generates a pseudo-random hopping sequence, which is mapped to the ranking table in order to build an adaptive hopping sequence. The frequency hopping performed over the adaptive hopping sequence increases the probability to achieve rendezvous on a channel with low PU activity and to decrease the time to attain the rendezvous. When the rendezvous is achieved, sender and receiver synchronize by exchanging SYNC packets. The SYNC packet includes the information for building the common hopping sequence. The two nodes then start hopping on the common hopping sequence in order to exchange data packets until the expiration of the time out period, when a new round of sensing is performed.

The different phases within a period of the sensing cycle can be summarized in the following routines: (i) Sensing, and (ii) Handshaking. We describe these phases as follows:

2) Sensing Routine: it is performed periodically with an interval equal to the sum of the sensing duration and a time out T_{out}. The sensing routine gives as an output the adaptive hopping sequence based on the sensing results. At the start up, i.e. when they are switched on, nodes may be asynchronous, and start their activity by performing spectrum sensing. As no network-wide silence duration is enforced, the CR user may not be able to distinguish between a signal originating from a PU, from another one from a CR user if simple energy detection techniques are used. For this reason, we assume that a feature detection scheme is in place [1]. After the sensing is completed, the node builds a channel *ranking table* based on the sensing results. In particular, a channel ranking table is a table where channels are ordered based on the PU activity, starting from the channels where the lowest PU activity is detected. It is worth noticing that neighboring nodes explore with high probability a similar PU activity across the channels, implying high correlation among their channel ranking tables and, consequently, between their adaptive hopping sequences.

After building the channel ranking table, the node generates a *pseudo-random sequence* containing integer values in the interval $[1, c]$, where c is the number of channels. The pseudo-random sequence is biased towards the lowest values, i.e. the lowest is the value the more often it occurs in the sequence. We propose two possible approaches for building these sequences in the (i) basic AMRCC scheme and the (ii) enhanced AMRCC scheme by using a decreasing linear function, or a parabolic decreasing function, respectively. In the basic AMRCC, the number of occurrences y of each value x in the sequence is given by the function:

$$y = -b \cdot x + c + 1. \tag{1}$$

where b is the slope of the line and c is the number of channels. For simplicity we assume b to be 1. In the enhanced AMRCC scheme, the number of occurrences y of each value x is given by the following function:

$$y = (c - x + 1)^2. \tag{2}$$

where c is the number of channels. The total length l of the sequence for the basic AMRCC is given by the following Gaussian formula:

$$l = c(c+1)/2. \tag{3}$$

where c is the number of channels. In the enhanced scheme, the length l of the sequence is given by:

$$l = \sum_{i=1}^{c} (c - i + 1)^2. \tag{4}$$

where c is the number of channels. In both cases the sequence is built by sampling without replacement the values from a set I, where the set I is defined as the set of the elements $j \epsilon N$, where the cardinality of each element j equal to $i \epsilon N$, is equal to y_i. In both two cases the sequence is repeated in a periodic manner until a change is forced by the protocol, for example a sensing is performed or the rendezvous is achieved. The following step is the *mapping* between the pseudo-random sequence and the channel ranking table. In particular, value i in the sequence is substituted with the i_{th} channel in the ranking table. In this way, the channels with higher ranking occur more often in the resulting sequence, which makes it extremely adaptive to the detected PU activity. A similar concept is exploited by Bluetooth devices as explained in [10], where the hopping sequences are altered in order to avoid channels with the highest interference.

3) Handshaking Routine: This phase begins when a node, which is hopping over the adaptive hopping sequence, has a packet to transmit. The handshaking procedure gives as an output the common hopping sequence after the rendezvous between the node and the intended receiver is achieved. After the sequence has been computed, the node starts hopping over the adaptive hopping sequence. As soon as a new packet arrives from the upper layers, the node starts sends a request to send (RTS) packet on each channel it hops in. If the intended receiver is on the same channel that the RTS packet is sent, it replies with a clear to send (CTS) packet. Upon receiving the CTS from the intended receiver, the *rendezvous* procedure is completed. The time interval between the instant at which the node starts the handshaking, and that in which it arrives at the rendezvous channel is referred as time to rendezvous (TTR). Now, the sender and receiver exchange SYNC packets in order to synchronize the future hops, and then exchange the *rendezvous packet*, that contains their ranking tables, the seed of their pseudo-random sequences, and the time elapsed from the last sensing, namely T_{ls}.

The next step for the node and the intended receiver is to compute the *common hopping sequence* they both will hop to in order to exchange data packets. The adaptive hopping sequence related to the lowest value of T_{ls} between sender and receiver is assumed by both the nodes as the common hopping sequence. Following the above procedure, the common hopping sequence is the most up to date between the sequences of transmitter and receiver. The nodes can easily compute each other sequence starting from the information included in the rendezvous packet. After exchanging data while hopping on the common hopping sequence, transmitter and receiver keep hopping till a new sensing action is performed and a new adaptive hopping sequence is computed. It is possible to notice that, unlike the most classic MRCCs [6], our solution does not cause high overhead in order to configure nodes joining the network, as seeds and ranking tables are exchanged only after rendezvous. The above implies high scalability. On the opposite, in previous solutions, when a new node joins the network, it must be configured by receiving the seed of other nodes' pseudo random sequence in order to start hopping on the same sequence of its intended receiver.

In the following Section the performance evaluation of our scheme, considering both the basic and the enhanced approaches, is given together with a comparison to a previous MRCC solution [6]. We put in evidence that our solution outperforms it in terms of time to rendezvous (TTR) and throughput.

III. Performance Evaluation

A. Simulation Setup

A simulation tool in Matlab was built in order to evaluate the performance of the proposed scheme. We assumed a single hop scenario where all the n CR users are in the same radio range. We assumed both PU and CR user traffic to have a Poisson distribution with mean value for the packet waiting time respectively equal to λ_{PU} and λ_{CR}. An asynchronous behavior of CR users is assumed, i.e. a random delay offset is introduced as the network initialization. Upon startup, each node performs sensing which is carried out periodically after a timeout T_{out}. The time out is set in the beginning of the simulation and it is the same for all the nodes. We suppose to ignore the collisions of control packets sent by CR users for the rendezvous with the packets sent by the PUs. The reason is that the frequency of RTS/CTS sending is very high during a single slot and so the effects of collisions are negligible. In fact, the portion of collided control packets will be generally unimportant compared to the total number of control packets transmitted during a slot. Another assumption that we do is to drop a CR user packet if it collides with a packet sent by a PU instead of retransmitting it after a backoff. Also the collisions among CR users can be ignored because the goal of the paper is to show the CR user behavior towards the PU activity, putting in evidence how efficiently CR users are able to exploit the spectrum holes from the PU activity. Furthermore, as the context is a single hop environment, we assume that all the CR users which are close among them experience a very similar primary user activity through the channels. This implies high correlation among the ranking tables of the CR users and thus improves the time to rendezvous and the overall performance.

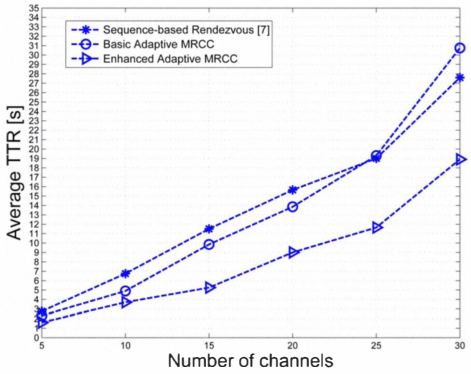

Figure 2. Average Time To Rendezvous.

B. Simulation Results

Here some results for the time to rendezvous (TTR) and the average throughput of CR users by using the AMRCC are given, together with a comparison with a previous sequence based MRCC solution [6].

In the following the main parameters set in the simulations are defined: the global observation time $\Delta_{ob} = 10000s$, the duration of one hop $\Delta_h = 5s$, the sensing duration

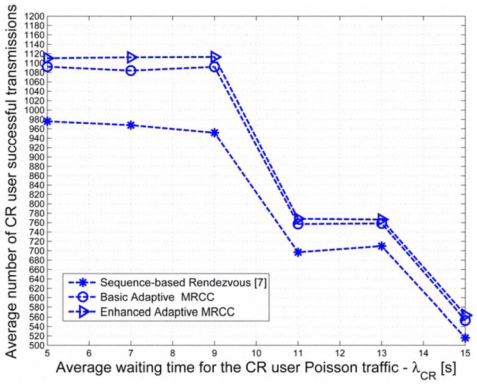

Figure 3. Average number of successful CR user transmissions vs the average waiting value for CR user traffic.

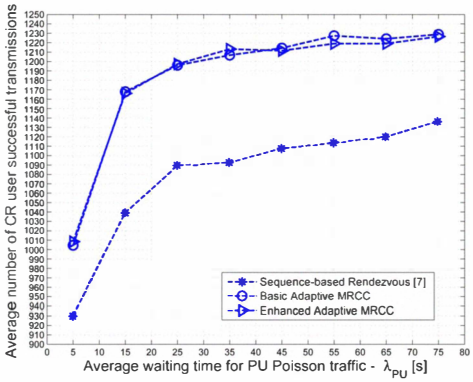

Figure 4. Average number of successful CR user transmissions vs the average waiting value for PU traffic.

$\Delta_s = 15s$, the time out for sensing $T_{out,s} = 500s$, the mean waiting time for PU traffic $\lambda_{PU} \in [5, 90]s$, the mean waiting time for CR traffic $\lambda_{CR} = 5s$, the number of CR users $n = 3$, the maximum offset for the CR user clock $off_{max} = 5s$, and the number of channels $c \in [5, 30]$. The results shown in this section are averaged over 20 iterations.

Figure 2 shows the average time to rendezvous. We observe that the average TTR increases with c because the sequences get longer and the probability to reach the rendezvous quickly is progressively lowered. The TTR for our AMRCC solution is always significantly lower than the one of the sequence-based rendezvous, with an average improvement of 32 %. On the other hand, the TTR for the basic AMRCC is lower compared to the sequence-based rendezvous only for c lower that 25, at which point the plots intersect. The reason is the following: When $c = 25$, the length of the sequences for the basic AMRCC is such that if we built a sequence of the same length, by concatenating periodically the sequence for the sequence-based rendezvous, the average probability of occurrence for each channel is on the average the same. The TTR depends on Δ_h, which is the duration of one hop, in this case equal to $5s$. The average TTR is proportional to the Δ_h.

Figure 3 shows the average number of successful CR user transmissions versus the mean waiting time for the CR user Poisson traffic λ_{CR}. The parameters have been set as follows: $\Delta_{ob} = 10000s$, $\Delta_h = 5s$, $\Delta_s = 15s$, $T_{out,s} = 500s$, λ_{PU} comprised in the interval $[5; 20]s$, λ_{CR} spanning in the interval $[5; 15]s$, $n = 3$, $off_{max} = 5s$ and $c = 5$. All the solutions show a decreasing curve with the increasing of the λ_{CR} as we expected and, especially for low values of λ_{CR}, i.e. for greedy traffic, both the basic and the enhanced AMRCCs show higher values compared to the sequence-based rendezvous. In particular, for $\lambda_{CR} = 9s$ our enhanced AMRCC outperforms the sequence-based rendezvous of almost 20 % achieving a number of successful transmissions which is the 56 % of the packets totally generated during Δ_{ob}.

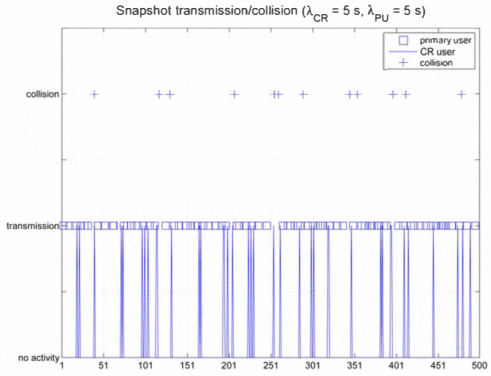

Figure 5. CR user transmissions/collisions in case of high PU activity.

Figure 4 shows the average number of CR user successful transmissions versus the average waiting time for PU Poisson traffic λ_{PU}, i.e. CR user activity versus PU activity. Obviously, when λ_{PU} increases, i.e. the PU traffic decreases, the number of CR user successful transmissions increases significantly. For all the values of c, both the basic and the enhanced AMRCC outperform the sequence-based rendezvous. Our solutions achieve an improvement of almost 13 % over the sequence-based rendezvous. Furthermore, for high values of λ_{PU}, i.e. for low PU activity, both the basic and the enhanced adaptive MRCCs achieve the 61 % of successful transmissions over the packets globally generated during Δ_{ob}.

We observe that the curves related to the basic and enhanced AMRCC do not show a significant difference, even though we expected the enhanced to perform better than the basic AMRCC. The reason is that, although the enhanced solution has lower average TTR and thus more CR user transmissions are attempted, the number of collisions to the PU packets leads to a similar performance in the results.

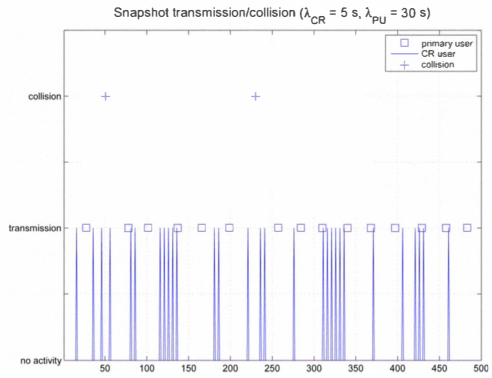

Figure 6. CR user transmissions/collisions in case of medium PU activity.

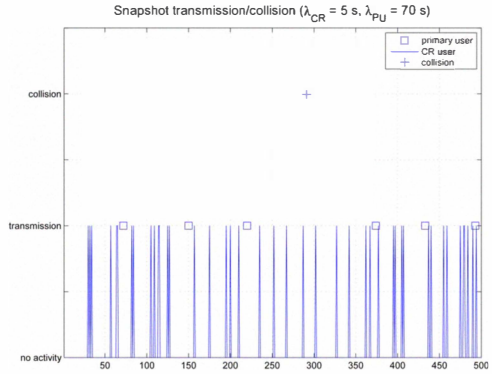

Figure 7. CR user transmissions/collisions in case of low PU activity.

Figures 5, 6 and 7 show snapshots of the protocol behavior in the interval of observation $[0; 500]s$. The above figures show how the CR user activity reacts to different PU traffic loads. In all the snapshots, the average waiting time for CR user Poisson traffic $\lambda_{CR} = 5s$, while different loads are considered for the PU traffic. In particular, Figure 5 refers to a high PU traffic load, where $\lambda_{PU} = 5s$, Figure 6 refers to a medium PU traffic load, with $\lambda_{PU} = 30s$, and Figure 7 refers to a low PU traffic load, with $\lambda_{PU} = 70s$. We observe that when the PU activity decreases, the number of successful CR user transmissions increases and the number of collisions between CR user and PU packets significantly decreases.

IV. Conclusions and Future Directions

In this paper we proposed AMRCC, a new adaptive multiple rendezvous control channel aiming at the reduction of the time to rendezvous and the increase of the overall network performance compared to the classical MRCCs. Our hopping sequence formation was adaptive based on the sensing results, where the channels with less PU activity occur more often. Two different approaches were considered: the basic adaptive MRCC and the enhanced adaptive MRCC, and both have the advantages not to require synchronization, are scalable and robust to dynamic PU activity. This work can be extended by making the slot time adaptive and by exchanging the ranking tables among the CR users. Overall, the design of *always available* CCC is an important prerequisitie for higher layer protocols in the area of CR networks.

References

[1] I. F. Akyildiz, W.-Y. Lee, and K. R. Chowdhury, "CRAHN: cognitive radio ad hoc networks", *Ad Hoc Networks (Elsevier) Journal*, pp. 810-836, Jul.2009.

[2] T. Chen, H. Zhang, G. M. Maggio, and I. Chlamtac, "CogMesh: A cluster-based cognitive radio network, in *Proc. of IEEE DySPAN*, pp. 168-178, Apr. 2007.

[3] M. C.-H. Chek, Y.-K. Kwok, "On adaptive frequency hopping to combat coexistence interference between Bluetooth and IEEE 802.11b with practical resource constraints", *Proc. of 7th International Symposium on Parallel Architectures, Algorithms and Networks*, pp. 391-396, May 2004.

[4] C. Cordeiro, K. Challapali, "C-MAC: A cognitive MAC protocol for multichannel wireless networks", in *Proc. of IEEE DySPAN*, pp. 147-157, Apr. 2007.

[5] C. Cormio, K. R. Chowdhury , "A survey on MAC protocols in cognitive radio wireless networks", in *Ad Hoc Networks (Elsevier) Journal*, Spring 2009.

[6] L. A. DaSilva, and I. Guerriero, "Sequence-based rendezvous for dynamic spectrum access", in *Proc. of the 3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN'08)*, pp. 1-7, Oct 2008.

[7] Y. R. Kondareddy, and P. Agrawal, "Synchronized MAC protocol for multi-hop cognitive radio networks", in *Proc. of IEEE International Conference on Communications (ICC)*, pp. 3198-3202, May 2008.

[8] L. Ma, X. Han, and C.-C. Shen, "Dynamic open spectrum sharing for wireless ad hoc networks", in *Proc. of IEEE DySPAN*, pp. 203-213, Nov. 2005.

[9] L. Ma, C.-C. Shen, and B. Ryu, "Single-radio adaptive channel algorithm for spectrum agile wireless ad hoc networks", in *Proc. of IEEE DySPAN*, pp. 547-558, Apr. 2007.

[10] B. Treister, , A. Batra, K. C. Chen, O. Eliezer, "Adaptive frequency hopping: A non-collaborative coexistence mechanism", in *IEEE P802.15 Working Group Contribution*, IEEE P802.15-01/252r0, Orlando, FL, May 2001.

[11] A. Tzamaloukas, and J. J. Garcia-Luna-Aceves, "Channel-hopping multiple access, in *Proc. IEEE International Conf. Comm. (ICC)*, vol. 1, pp. 415-419, Jun. 2000.