Cluster-based Control Channel Allocation in Opportunistic Cognitive Radio Networks

Sisi Liu, *Student Member, IEEE,* Loukas Lazos, *Member, IEEE,* and Marwan Krunz, *Fellow, IEEE*
Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, 85721
E-mail:{sisimm, llazos, krunz}@ece.arizona.edu

Abstract—Cognitive radio networks (CRNs) involve extensive exchange of control messages, which are used to coordinate critical network functions such as distributed spectrum sensing, medium access, and routing, to name a few. Typically, control messages are broadcasted on a pre-assigned common control channel, which can be realized as a separate frequency band in multi-channel systems, a given time slot in TDMA systems, or a frequency hopping sequence (or CDMA code) in spread spectrum systems. However, a static control channel allocation is contrary to the opportunistic access paradigm. In this paper, we address the problem of *dynamically* assigning the control channel in CRNs based on time- and space-varying spectrum opportunities. We propose a cluster-based architecture that allocates different channels for control at various clusters in the network. The clustering problem is formulated as a bipartite graph problem, for which we develop a class of algorithms that provide different tradeoffs between two conflicting factors: number of common channels in a cluster and the cluster size. Clusters are guaranteed to have a desirable number of common channels for control, which facilitates for graceful channel migration when primary radio (PR) activity is detected, without the need for frequent reclustering. We perform extensive simulations that verify the agility of our algorithms in adapting to spatial-temporal variations in spectrum availability.

Index Terms—Dynamic spectrum networks, control channel assignment, cognitive radios, bipartite graphs, clustering.

1 Introduction

Fixed spectrum allocation policies address interference between different wireless technologies by isolating their operation in frequency. This fixed allocation policy has led to spectrum scarcity. As a consequence, new wireless services and technologies have strived to co-exist in overcrowded unlicensed bands with poor radio propagation characteristics. At the same time, it has been lately observed that portions of the licensed spectrum are highly underutilized [2], [14].

To address the emerging need for higher spectral efficiency, an alternative policy that allows unlicensed users to opportunistically access vacant portions of the licensed spectrum is currently being examined [2], [14]. In this so-called opportunistic spectrum access (OSA) paradigm, users are classified as primary if they are licensed to operate on a particular frequency band, and secondary otherwise. Secondary users can operate in licensed frequency bands only if the do not interfere with primary radio (PR) activity. Cognitive Radios (CRs) are intended as an enabling technology for OSA [2]. These devices are capable of sensing the spectrum for frequency holes and adapting their radio parameters to exploit spectrum opportunities without interfering with PRs.

Establishing a self-organizing cognitive radio network (CRN) requires extensive exchange of control messages, needed to coordinate various network functions such as cooperative sensing, channel access, topology management, and routing, to name a few. In many wireless networks architectures, control messages are broadcasted over a channel known to all nodes, commonly referred to as *the control channel*. This channel can be realized in a number of ways. For example, in multi-channel system, it can be a frequency band dedicated to control traffic (e.g., [28]). It can also be a designated time slot in a TDMA system, or a frequency hopping sequence (or CDMA code) in spread spectrum systems. In an opportunistic CRN, spectrum availability exhibits the temporal and spatial variations due to PR activity. Therefore, there is no guarantee that a given frequency band will be available for exchanging control information, either locally (one-hop neighborhood) or over multiple hops [31]. We refer to the problem of defining a channel for control purposes as the *control-channel assignment (CCA)* problem.

One possible solution to the CCA problem is to license a slice of the spectrum for control purposes [9]. However, such a design conflicts with the opportunistic nature of CRNs. Alternatively, the control channel can reside within an unlicensed band, such as the Industrial, Scientific and Medical (ISM) band, or the unlicensed ultra-wide band (UWB) [8], [9]. The use of unlicensed bands jeopardizes the reliability of the control channel, given that such bands are already overcrowded and can experience uncontrollable interference from other unlicensed users. In the absence of a dedicated frequency band, control traffic has to be carried in-band. In such a case, the control channel is subject to PR dynamics, and hence its allocation has to vary in frequency and time according to the locally perceived spectrum availability [4], [23], [31].

Our Contributions–We develop cluster-based methods for CCA in CRNs. This is an intuitive approach given the inherent partitioning of the network into clusters due to the location- and time-dependent spectrum availability. We formulate the clustering problem as a *bipartite graph problem*. In particular, we map the clustering process to the maximum edge biclique problem [13], [25] and the maximum one-sided edge cardinality problem [13]. Our mapping allows us to control the tradeoff between the set of common idle channels within each cluster and the cluster size.

Based on our graph theoretic formulations, we develop a distributed clustering algorithm called Spectrum Opportunity-based Clustering (SOC). SOC clusters neighboring CRs with similar channel availability. We show that such a criterion reduces the frequency of reclustering due to PR dynamics. Under SOC, nodes reach a mutual agreement with respect to cluster memberships

A preliminary version of this paper was presented at the IEEE SECON 2009 Conference*, Rome, June 2009.*

after the local exchange of a small number of messages. Finally, to account for PR dynamics in the time domain and to enable inter-cluster coordination, we propose a periodic channel rotation mechanism. By rotating the control channel among the list of common idle channels in the cluster, nodes in that cluster can communicate with a neighboring cluster as long as the two clusters have at least one common idle channel.

Paper Organization–The remainder of this paper is organized as follows: Section 2 discusses related work. In Section 3, we formalize the CCA problem and present our system model. Section 4 shows the mapping of the CCA problem to a bipartite graph problem. In Section 5, we describe the spectrum opportunity clustering (SOC) algorithm. A CCA mechanism that adapts to the spectrum dynamics is presented in Section 6. In Section 7, we develop a coordination protocol for CRNs for the initial exchange of information before a control channel is established. In Section 8, we evaluate the performance of SOC and compare it with other methods. In Section 9, we summarize our contributions.

2 Related Work

Previously proposed CCA schemes for CRNs can be classified into: (a) static assignment of a dedicated frequency band common to all CRs, and (b) dynamic assignment based on criteria such as spatial correlation, spectrum usage, connectivity degree, etc. We describe both categories in detail.

Static Control Channel Assignment Schemes–Several researchers have proposed the exchange of control information on an always available static frequency band, known to all nodes (e.g., [8], [9], [19], [27]). Čabrić et al. proposed the CORVUS system, in which control traffic is transmitted using UWB technology [9]. Brown proposed the use of ISM bands for control in CRNs [8]. Han et al. proposed an OFDM-based scheme to allow for long-range transmission of control messages with small bit error rates [19]. Several MAC protocol designs for CRNs assume the existence of a dedicated control channel, without specifying its allocation (e.g. [20], [27]).

Dynamic Control Channel Assignment Schemes–Zhao et al. proposed distributed coordination of CRs via a locally computed control channel that changes in response to PR activity [31]. The band available to the largest set of one-hop neighbors is selected for control in each neighborhood, implementing a partition of the CRN into clusters. This approach minimizes the number of distinct frequency bands needed for control, thus reducing the overhead of cluster management. However, this can lead to frequent reclustering due to variations in PR activity. Another cluster-based design was adopted in [11].

Chen et al. proposed a swarm-intelligence-based algorithm for adapting the control channel based on individual interference measurements [10]. Neighboring CRs engage in a negotiation process to decide on a control channel. This negotiation is carried out in licensed bands without consideration for PRs. Kondareddy and Agrawal proposed dynamic hopping of the control channel based on pseudo-random sequences [21]. Transmitter/receiver pairs randomly meet in different bands and decide on a common hopping sequence, called the *rendezvous channel*, until their data exchange is completed. One limitation of this design is that hopping coordination occurs over licensed channels without considering possible interference to PRs.

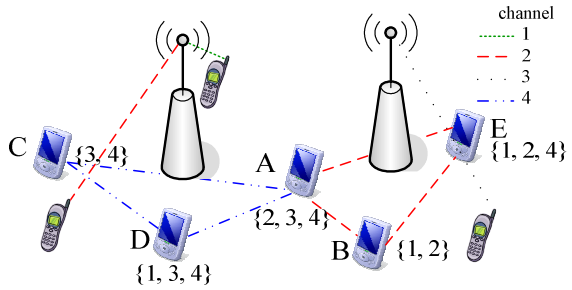


Fig. 1. Control channel assignment based on PR activity (idle frequency channels are indicated between braces).

Bahl et al. proposed WhiteFi, a system that provides Wi-Fi like connectivity over the UHF white spaces [4]. WhiteFi incorporates dynamic channel assignment algorithms for detecting and managing spectrum opportunities. Similar to our scheme, WhiteFi handles control traffic in-band, using one main and one backup control channel. The locations of the main and backup channels vary according to the dynamics of the spectrum. The main differences between how WhiteFi handles the CCA problem and our work are that: (a) WhiteFi is designed for an access point-client architecture, where a large set of clients is connected to a single access point. We consider an ad hoc network model where the control channel has to be maintained over multiple hops; (b) WhiteFi maintains only one backup channel for broadcasting control information. SOC organizes the CRN to clusters where several common idle channels are available for carrying control traffic and therefore, is more resilient to temporal variations of the spectrum.

3 Problem Statement and System Model

Problem Statement–The CCA problem addressed in this paper is illustrated in Figure 1, where a cellular network that acts as a PRN co-exists with an ad hoc network of CRs. CRs opportunistically use any of the four cellular channels that is sensed idle in its vicinity. For example, the idle channel list of CR node A, denoted as CR_A consists of channels $\{2, 3, 4\}$. Our goal is to allow CRs to agree on a CCA according to their spectrum vacancies. In the example in Figure 1, none of the idle channels is common to all CRs. Hence, different channels have to be assigned for control in different neighborhoods. This assignment leads to a natural partitioning of the CRN into clusters, each with at least one common idle channel. To assign the control channel, we investigate clustering algorithms that take into account the spatial and temporal PR activity dynamics.

System Model–We consider a CRN that co-exists with one or more PRNs in the same geographical area. PRs are licensed to operate on a fixed spectrum, which can be divided into a set of M *non-overlapping* frequency bands. Let $\mathcal{M} = \{1, 2, \ldots, M\}$ be such a set. For simplicity, it is assumed that these bands are of equal capacity, and that the CRN maintains the channelization structure of the PRNs. We also assume that CRs have the same communication range over every channel in $\mathcal{M}$. Hence, the connectivity graph is not impacted by which channel is used for

control. This property can be achieved, for example, by adjusting the transmission power.

CRs sense spectrum opportunities using energy detectors, cyclostationary feature extraction, or pilot signals. They may exchange their sensed data and cooperate in identifying spectrum opportunities [2], [17], [18], [24]. Spectrum sensing is conducted at a sufficient rate such that the list of available channels at each CR is up-to-date. However, the sensing process is assumed to be imperfect due to multipath fading and/or severe shadowing [17], [18], [24]. Such phenomena are typical in the bands that are likely to be open for CRN use (e.g., digital TV bands [15], [24]).

For the ith CR denoted by CR_i, its list of idle channels is denoted by $C_i = \{f_1^{(i)}, f_2^{(i)}, \ldots, f_{K_i}^{(i)}\}$, where $K_i = |C_i|$. Here, $f_j^{(i)}$ refers to the index of the jth channel in C_i, i.e., $f_j^{(i)} \in \mathcal{M}$. Finally, we assume a time-slotted system for CRN communications. Time synchronization for the purpose of maintaining a single time reference can be achieved using any of the methods in [16], [22], [30], or by periodic synchronization to PR signals.

4 Cluster-based Channel Assignment

To account for space- and time-dependent spectrum availability, we propose a cluster-based CCA scheme. In this scheme, the CRN is partitioned into clusters. Nodes in a given cluster observe highly similar spectrum opportunities. The control channel within each cluster is selected from the set of common idle channels. Ensuring a large set of common idle channels in each cluster has several advantages. First, if the current control channel becomes occupied by a PR user, CRs can migrate to a new one. Second, grouping neighboring CRs with similar idle channels implicitly implements hard-decision cooperative sensing [17], [18], [24]. Third, multiple concurrent data transmissions can take place within each cluster. On the other hand, if spectrum opportunities are highly heterogeneous, requiring a large number of common idle channels per cluster may lead to small cluster sizes and a large number of clusters. In this case, cluster management and inter-cluster communications involve significant overhead.

To provide a graceful tradeoff between cluster size and the number of cluster-wide idle channels, we formulate the clustering problem as a *bipartite graph problem*. Specifically, our clustering algorithms, called Spectrum-Opportunity Clustering (SOC) and Constrained-SOC (C-SOC), utilize two instances of a biclique construction problem: the *maximum edge biclique graph problem* [13], [25], and the *maximum one-sided edge biclique graph problem* [12]. We first present the mapping from the clustering problem to a biclique construction. Then, we show how bicliques can be utilized for distributed clustering.

4.1 Mapping to Biclique Graphs

Figure 2(a) shows the connectivity graph of an example CRN. Following neighborhood discovery and the exchange of idle channels, each CR_i becomes aware of its one-hop neighbors CR_j and the channel list $C_j, \forall\ \text{CR}_j \in \text{NB}_i$. This information can be represented as a bipartite graph. A graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is called bipartite if the set of vertices $\mathcal{V}$ can be partitioned into two disjoint sets $\mathcal{A}$ and $\mathcal{B}$ with $\mathcal{A} \cup \mathcal{B} = \mathcal{V}$, such that every edge in $\mathcal{E}$ connects a vertex in $\mathcal{A}$ to a vertex in $\mathcal{B}$. For CR_i, the set $\mathcal{A}$ corresponds to its neighborhood set NB_i plus CR_i itself, while $\mathcal{B}$ corresponds to the set of idle channels C_i. An edge (x, y) exists between vertices $x \in \mathcal{A}_i$ and $y \in \mathcal{B}_i$ if $y \in C_x$, i.e., channel y is in the channel list of CR_x. In Figure 2(b), we show the bipartite graph $\mathcal{G}_A(\mathcal{A}_A \cup \mathcal{B}_A, \mathcal{E}_A)$ constructed by CR_A. By construction, CR_A is connected to all vertices in $\mathcal{B}_A$.

A bipartite graph $Q(\mathcal{V} = X \cup Y, \mathcal{E})$ is called a *biclique* if for each $x \in X$ and $y \in Y$ there exists an edge between x and y, i.e., $\mathcal{E} = \{(x, y) \mid \forall\ x \in X \text{ and } \forall\ y \in Y\}$. The edge set $\mathcal{E}$ is completely determined by X and Y, and hence, is omitted from the biclique notation. For CR_i, a biclique graph $Q_i(X_i, Y_i)$ can be extracted from its bipartite graph $\mathcal{G}_i$. This biclique represents a cluster of nodes X_i that have channels $Y_i \subseteq C_i$ in common. In Figures 2(c), and 2(d) we show two possible bicliques for the bipartite graph $\mathcal{G}_A$. The first biclique (Figure 2(c)) represents the cluster $X_A = \{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_G\}$ with common channels $Y_A = \{1, 2, 3\}$. The second biclique (Figure 2(d)) represents the cluster $X_A = \{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_G, \text{CR}_H\}$ with common channels $Y_A = \{1, 2\}$. The algorithms for obtaining both bicliques will be given shortly.

To organize the CRN into clusters, we are interested in forming bicliques that satisfy certain performance criteria. We choose to: (a) maximize the set of edges of the biclique graph, or (b) maximize the cluster size under a constraint on the number of common idle channels in the cluster. We now describe both criteria in detail.

4.2 Maximum Edge Biclique Graphs

The first clustering criterion is to maximize the number of edges in the biclique graph. This corresponds to maximizing the product of the cluster size and the number of common channels. We show that this criterion gracefully adapts to spatial heterogeneity in spectrum availability. It also provides a tradeoff between cluster size and the number of idle channels in a cluster.

To illustrate, consider the biclique $Q_i(X_i, Y_i)$ associated with CR_i. Suppose that there is another biclique $Q_i^*(X_i^*, Y_i^*)$, where $|X_i^*| = |X_i| + \Delta|X_i|$ and $|Y_i^*| = |Y_i| + \Delta|Y_i|$. Note that an increase in the number of common channels by $\Delta|Y_i| > 0$ will result in $\Delta|X_i|$ change in the cluster size, where $\Delta|X_i| \leq 0$. According to the maximization criterion, Q_i^* should be selected over Q_i if

$$|X_i||Y_i| < (|X_i| + \Delta|X_i|)(|Y_i| + \Delta|Y_i|). \tag{1}$$

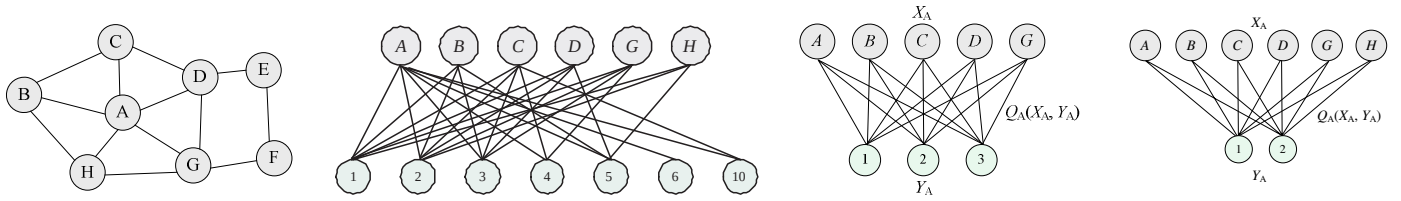
The above inequality translates into:

$$-\left(\frac{\Delta|X_i|}{|X_i|} + \frac{\Delta|Y_i|}{|Y_i|}\right) \quad < \quad \frac{\Delta|X_i|\Delta|Y_i|}{|X_i||Y_i|} \tag{2}$$

under the constraints

$$|Y_i| + \Delta|Y_i| \leq |C_i| \quad \text{and} \quad |X_i| + \Delta|X_i| \leq |N_i|.$$

Inequality (2) states that the fractional change in the number of edges has to be larger than the fractional change in the number of vertices of the biclique. The effect of the maximum-edge biclique construction on the clustering process can be explained as follows. If the CRs in a given neighborhood have similar channels lists, our clustering rule will be fairly inclusive, resulting in large



$C_A = \{1,2,3,4,5,6,10\}$, $C_B = \{1,2,3,5,7\}$, $C_C = \{1,2,3,4,10\}$, $C_D = \{1,2,3,5,7\}$, $C_E = \{2,3,5,7\}$, $C_F = \{2,4,5,6,7,10\}$, $C_G = \{1,2,3,4,8\}$, $C_H = \{1,2,5,8\}$.

Fig. 2. (a) Connectivity graph of an 8-node CRN, and the lists of idle channels sensed by various CRs, (b) bipartite graph constructed by CR_A.

Algorithm 1 Greedy Heuristic for Computing the Maximum Edge Biclique Graph

INPUT $\mathcal{G}_i(\mathcal{A}_i \cup \mathcal{B}_i, \mathcal{E}_i)$ // bipartite graph of CR_i
$Y_i \leftarrow \mathcal{B}_i$
for $j = 1$ to $|\mathcal{A}_i|$ **do**
  Find $\text{CR}_k \in \mathcal{A}_i$ that maximizes $|Y_i \bigcap C_k|$ over all nodes in $\mathcal{A}_i$
  if $Y_i \bigcap C_k = \emptyset$ **then**
    break
  else
    $S_i[j] = k$
    $\mathcal{A}_i \leftarrow \mathcal{A}_i - \text{CR}_k$, $X_i \leftarrow X_i \bigcup \text{CR}_k$, $Y_i \leftarrow Y_i \bigcap C_k$
    $P_i[j] = |X_i| \times |Y_i|$
  end if
end for
Find $j^* = \arg\max_j P_i[j]$
return $Q^*(X_i^*, Y_i^*)$; $X_i^* = \{\text{CR}_{S_i[1]}, \ldots, \text{CR}_{S_i[j^*]}\}$; $Y_i^* = \bigcap_{k=1}^{k=j^*} C_{S_i[k]}$

clusters. On the other hand, if the channel lists of neighboring CRs vary significantly, the clustering rule will reduce the cluster size in favor of the common channels within each cluster. Note that in general, the maximum edge biclique may contain any subset of vertices of the original bipartite graph. Our construction guarantees that: (a) any channel common to all neighbors of CR_i will be part of the maximum-edge biclique Q_i^*, and (b) CR_i will also be part of Q_i^*. This is shown in the following lemma.

Lemma 1: Let a vertex $x \in \mathcal{A}$ of a bipartite graph $\mathcal{G}(\mathcal{A} \cup \mathcal{B}, \mathcal{E})$ be connected to all vertices in the set $\mathcal{B}$. Then, x belongs to the maximum-edge biclique $Q^*(X^*, Y^*)$.

Proof: The proof is provided in Appendix 1. □

Finding the maximum-edge biclique of a bipartite graph is an NP-complete problem [25]. For small bipartite graphs, an exhaustive search is possible. However, the search space grows exponentially with the cardinality of the vertex set. Accordingly, we develop a greedy heuristic (Algorithm 1) that produces a biclique with a large number of edges. In each iteration, Algorithm 1 examines one CR node. The vector S_i holds the indices of CRs that have already been examined, whereas Y_i holds the list of common channels for the CRs in S_i. Initially, S_i is empty and $Y_i = \mathcal{B}_i$. In each iteration, we find a node CR_k whose channel list C_k has the highest overlap with Y_i. We then remove CR_k from $\mathcal{A}_i$, add k to S_i, and repeat the process until either $\mathcal{A}_i$ is empty or until the intersection of the common channel list with the remaining CRs is null. Note that in each iteration, $Q(X_i, Y_i)$ is a biclique. We record the number of edges of each constructed biclique in the vector P_i, and then find the biclique with the maximum number of edges. Algorithm 1 guarantees that a CR_x with $C_i \subseteq C_x$ will be included in the biclique Q_i^*. This is formalized in the following lemma.

Lemma 2: Any $x \in A_i$ with $C_i \subseteq C_x$ will be included in the biclique $Q_i^*(X_i^*, Y_i^*)$ computed by Algorithm 1.

Proof: The proof is provided in Appendix 2. □

We illustrate the steps of Algorithm 1 when executed at node A of Figure 2(a). The bipartite graph $\mathcal{G}_A$ for CR_A is shown in Figure 2(b). In the first iteration, node A is selected, as it has the highest overlap with $Y_A = C_A$. The number of edges of the biclique is $P_A[1] = 7$. In the second iteration, CR_C is selected, resulting in cluster $X_A = \{\text{CR}_A, \text{CR}_C\}$, $Y_A = \{1,2,3,4,10\}$, and $P_A[2] = 10$. Subsequent iterations result in the addition of $\text{CR}_D, \text{CR}_B, \text{CR}_G$, and CR_H, in this order, and corresponding numbers of edges $P_A[3] = 9$, $P_A[4] = 12$, $P_A[5] = 15$, and $P_A[6] = 12$. The biclique Q_A^* with the maximum number of edges is eventually obtained. The final outcome is the cluster $\{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_G\}$ with common channels $\{1,2,3\}$, depicted in Figure 2(c).

4.3 Maximum One-Sided Edge Biclique Graphs

When maximizing the number of edges in the biclique, no requirement is imposed on the cluster size $|X_i|$ or the set of common channels $|Y_i|$. If there are large differences between the channel lists of neighboring CRs, this approach may result in clusters of very small sizes. To avoid this outcome, we examine a constrained version of the maximum-edge biclique problem, which aims at maximizing the cluster size while satisfying a lower bound on the number of common channels. Such a formulation is related to the maximum one-sided edge biclique problem [12], which can be stated as follows. Given a bipartite graph $\mathcal{G}(\mathcal{A} \cup \mathcal{B}, \mathcal{E})$ and a positive integer k, we wish to find a maximum-edge biclique with at least k nodes on one side of the bipartition. In our problem, this corresponds to imposing a lower bound on $|Y_i|$ and maximizing $|X_i|$.

The maximum one-sided edge biclique problem is known to be NP-complete [12]. We provide a greedy algorithm (Algorithm 2) that yields clusters with $|Y_i| \geq \gamma_0$ idle channels in common, where γ_0 is a desired threshold. Algorithm 2 examines one channel in each iteration. The set X_i is initialized to all one-hop neighbors of CR_i. The set Y_i is initially empty. At the

Algorithm 2 Greedy Heuristic for Computing the Maximum One-sided Edge Biclique Graph

INPUT $\mathcal{G}_i(\mathcal{A}_i \cup \mathcal{B}_i, \mathcal{E}_i)$; t_0
$Y_i = \emptyset, \quad X_i = \mathcal{A}_i, \quad k = 1$
while $|Y_i| < \gamma_0$ and $|X_i| > 0$ **do**
  Find $y_k^* = \arg\max_{y \in \mathcal{B}_i \setminus Y_i} deg(y)$
  if y_k^* connects to no CR **then**
    break
  else
    $X_i \leftarrow X_i \bigcap S_i$; $S_i = \{\text{CR}_j \in \mathcal{A}_i \mid y_k^* \in C_j\}$
    $Y_i \leftarrow Y_i \bigcup y_k^*$
  end if
  $k = k + 1$
end while
return $Q^*(X_i^*, Y_i^*)$

kth iteration, CR_i finds channel $y_k^* \in B_i \backslash Y_i$ that is common to the largest number of one-hop neighbors, i.e., the channel with the highest connectivity degree $deg(y)$ in the bipartite graph $\mathcal{G}_i(\mathcal{A}_i \cup \mathcal{B}_i, \mathcal{E}_i)$. Any neighbor that has not sensed y_k^* as idle is removed from X_i. Then, y_k^* is added to Y_i. If several channels have the same degree, the decision of selecting y_k^* is deferred to the next iteration. All values of y_k^* are stored and for each one we find channel $y_{k+1}^* \in B_i \backslash Y_i + y_k^*$ with the highest degree. Assuming y_{k+1}^* is unique, then y_k^* and y_{k+1}^* are added to Y_i; else we proceed to the next iteration[1]. Note that at each step, the graph $Q(X_i, Y_i)$ is a biclique, because y_k^* is connected to all CRs in X_i. The number of required iterations in Algorithm 2 is equal to or less than γ_0.

We illustrate the application of Algorithm 2 to the CRN of Figure 2(a). Let $\gamma_0 = 2$. For node A, we initially have Y_A=$\emptyset$ and X_A=$\{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_H, \text{CR}_G\}$. Channels 1 and 2 have the highest degree of six, so both of them are included in Y_A. The cluster membership X_A remains intact. In the next round, we start with Y_A=$\{1\}$ and X_A=$\{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_H, \text{CR}_G\}$ and add channel 2 to Y_A. X_A remains the same. For Y_A=$\{2\}$ and X_A=$\{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_H, \text{CR}_G\}$, channel 1 is selected. So the two (X_A, Y_A) pairs form the same biclique, given by Y_A=$\{1, 2\}$ and X_A=$\{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D, \text{CR}_H, \text{CR}_G\}$. Since $|Y_A|$ satisfies $\gamma_0 \geq 2$, the algorithm terminates and returns $Q_A(X_A, Y_A)$. This biclique is depicted in Figure 2(d).

5 Spectrum-Opportunity Clustering

In this section, we develop the *spectrum-opportunity clustering* (SOC) algorithm which utilizes the clustering criteria based on the biclique mapping, as presented in Section 4. The SOC algorithm follows four steps:

Step 1: *Biclique computation*–In step 1, each CR_i is aware of its one-hop neighbors along with the channel availability at each $\text{CR}_j \in \text{NB}_i$. Using this information, CR_i constructs a bipartite graph and computes the "best" biclique $Q_i^1(X_i^1, Y_i^1)$. Here the superscript is used to denote the obtained biclique after a given iteration. The "best" biclique is computed using Algorithm 1 or Algorithm 2, as described in Section 4. We refer to the clustering method that uses Algorithm 1 as SOC, and to the clustering method that uses Algorithm 2 as Constrained SOC (C-SOC). Once the optimal bicliques are computed, CR_i broadcasts its biclique info $Q_i^1(X_i^1, Y_i^1)$ to its neighbors.

Step 2: *Updating cluster memberships*–In step 2, each CR_i checks if there is a biclique Q_j^1 with $\text{CR}_i \in X_j^1$ that provides *better* clustering than Q_i^1. That is, it checks if $Q_j^1 > Q_i^1$ with $\text{CR}_i \in X_j^1$. The inequality operator for two bicliques is defined as follows.

Definition 1: For two bicliques $Q_i(X_i, Y_i)$ and $Q_j(X_j, Y_j)$ constructed using Algorithm 1, we say $Q_i < Q_j$ if:

(a) $|X_i| \times |Y_i| < |X_j| \times |Y_j|$, or
(b) $|X_i| \times |Y_i| = |X_j| \times |Y_j|$ and $|X_i| < |X_j|$, or
(c) $|X_i| = |X_j|, \quad |Y_i| = |Y_j|$, and $i < j$.

In Definition 1, we first compare the number of edges in the two bicliques. If two bicliques have the same number of edges, we then compare their cluster sizes. If the cluster sizes are also equal, we break the tie by selecting the biclique of the CR with the highest id. Definition 1 imposes a total ordering between two bicliques (i.e., two bicliques can never be equal). Now for the C-SOC case, the inequality operator is defined as follows.

Definition 2: For two bicliques $Q_i(X_i, Y_i)$ and $Q_j(X_j, Y_j)$ constructed using Algorithm 2, we say $Q_i < Q_j$ if:

(a) $|X_i| < |X_j|$, or
(b) $|X_i| = |X_j|$ and $|Y_i| < |Y_j|$, or
(c) $|X_i| = |X_j|, \quad |Y_i| = |Y_j|$, and $i < j$.

In Definition 2, given that both Q_i and Q_j satisfy the constraint on the number of idle channels, we select the biclique that leads to a larger cluster size. If cluster sizes are equal, we compare the number of idle channels per cluster. If those are equal as well, we break the tie by selecting the biclique of the CR with the higher id. CR_i selects biclique $Q_j^1(X_j^1, Y_j^1)$ with $\text{CR}_i \in X_j^1$, that is best according to the relation operator given in Definitions 1 or 2, and updates its maximum edge biclique to $Q_i^2 = Q_j^1$. After computing Q_i^2, CR_i informs its neighbors of the updated cluster membership X_i^2 and the common channel list Y_i^2.

We illustrate the execution of step 2 for the CRN in Figure 2(a). CR_A receives the following updates from its neighbors: (a) Q_B^1 with $X_B = \{\text{CR}_A, \text{CR}_B, \text{CR}_H\}$ and $Y_B = \{1, 2, 5\}$, (b) Q_C^1 with $X_C = \{\text{CR}_A, \text{CR}_B, \text{CR}_C, \text{CR}_D\}$ and $Y_C = \{1, 2, 3\}$, (c) Q_D^1 with $X_D = \{\text{CR}_A, \text{CR}_C, \text{CR}_D, \text{CR}_E, \text{CR}_G\}$ and $Y_D = \{2, 3\}$, (d) Q_G^1 with $X_G = \{\text{CR}_A, \text{CR}_D, \text{CR}_G, \text{CR}_H\}$ and $Y_G = \{1, 2\}$, and (e) Q_H^1 with $X_H = \{\text{CR}_A, \text{CR}_B, \text{CR}_G, \text{CR}_H\}$ and $Y_H = \{1, 2\}$. Ordering the bicliques according to Definition 1 yields $Q_G^1 < Q_H^1 < Q_B^1 < Q_D^1 < Q_C^1 < Q_A^1$. Then A sets $Q_A^2 = Q_A^1$ since Q_A^1 has the maximum number of edges. Similarly, $Q_B^2 = Q_A^1$, $Q_C^2 = Q_A^1$, $Q_D^2 = Q_A^1$, $Q_G^2 = Q_A^1$, and $Q_H^2 = Q_A^1$.

Step 3: *Finalizing cluster membership*–In step 3, each CR_i examines the cluster membership X_i^2. For each $\text{CR}_j \in X_i^2$, if $\text{CR}_i \notin X_j^2$ then CR_i removes CR_j from its biclique Q_i^2. At the completion of this step, the final bicliques Q_i^3 are obtained for all nodes that were not removed from the biclique of their choice

1. To simplify the exposition, the details of breaking the tie are not shown in the pseudo-code of Algorithm 2.

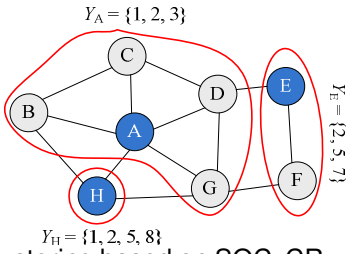


Fig. 3. Final clustering based on SOC. CR_A, CR_E, and CR_H are the CHs.

in step 2. If a CR_i has adopted the clustering of CR_j during step 2, it may be the case that X_j^2 contains CRs not within the range of CR_i. For those CRs, CR_i will not have biclique information. To provide this information, every CR_j must replay all received biclique information if its biclique is selected by any of its neighbors in step 2.

For illustration, consider the CRN in Figure 2(a). CR_A checks if it is included in the bicliques of all CRs in X_A^2. Given that all neighboring CRs have adopted Q_A^1, CR_A concludes that $X_A^3 = X_A^2$ and $Y_A^3 = Y_A^2$. Similarly, B goes through its list $X_B^2 = \{CR_A, CR_B, CR_C, CR_D, CR_G\}$. Because CR_D and CR_G are not neighbors of CR_B, the only way that CR_B can know about Q_D^2 and Q_G^2 is if CR_A relays their updates. According to our algorithm, CR_A relays Q_D^2 and Q_G^2, because both CR_D and CR_G have adopted Q_A^1. CR_B can now see that it is included in the bicliques of CR_D and CR_G, so it sets $X_B^3 = \{CR_A, CR_B, CR_C, CR_D, CR_G\}$. Note that CR_H is excluded from the biclique updates of CR_A, CR_B, and CR_G, and therefore continues to step 4.

Step 4: *Unclustered CRs*–CRs that did not join any clusters because they were removed from the biclique they chose in step 3 repeat steps 1-3, but exclude already clustered neighbors. For example, CR_H in Figure 2(a) was excluded by the biclique of CR_A and hence, has to join another cluster. CR_H deletes CR_A, CR_B and CR_G from its neighbor list, and exchanges information with the remaining neighbors to construct a cluster. If there are no remaining one-hop neighbors, as in the case of CR_H, then a single CR cluster is formed. The final clustering and the list of common channels are shown in Figure 3.

5.1 Correctness of the SOC Algorithm

We now prove that the SOC algorithm leads to consistent cluster memberships, i.e. all CRs distributively reach the same clustering outcome. The correctness proof follows the logic of the clique clustering method in [29]. Several modifications are made to the use of bicliques with cluster memberships that possibly do not form cliques. To show the correctness of SOC, we prove that at the end of step 3, $Q_i^3 = Q_j^3$ for any CR_j that belongs to cluster X_i^3. Because step 4 is a repetition of steps 1-3, it follows that SOC converges to the same cluster memberships. We first prove a series of lemmas leading to our main proof.

Lemma 3: If $CR_i \in X_j^2$ and $CR_j \in X_i^2$, then $Q_i^2 = Q_j^2$.

Proof: The proof is provided in Appendix 3. □

According to Lemma 3, two CRs that include each other in their respective bicliques after step 2 must have agreed on the same bicliques. We utilize this result in Lemma 4.

Lemma 4: Suppose that for three nodes CR_i, CR_j, and CR_k, we have $CR_k \in X_i^2$ and $CR_k \in X_j^2$ with $Q_i^2 = Q_j^2$. Then if $CR_i \notin X_k^2$, it must also hold that $CR_j \notin X_k^2$.

Proof: The proof is provided in Appendix 4. □

Based on Lemmas 3 and 4, we now show that SOC guarantees that CRs will have consistent cluster membership information. It also follows that CRs will agree on the set of common channels.

Theorem 1: For any $CR_j \in X_i^3$, $Q_i^3 = Q_j^3$.

Proof: The proof is provided in Appendix 5. □

Based on Theorem 1, at the end of step 3, CRs that have not been excluded from their cluster choice in step 2 agree on the same clusters. For any CRs that are removed (CR_H in our example), steps 1-3 are repeated with the exclusion of any already clustered members. Hence, the new clusters formed at step 4 lead to consistent cluster formation.

5.2 Clusterhead Election

SOC is a cluster-first algorithm, so clusterheads (CHs) are elected after clusters are formed. CHs are used to facilitate operations such as cooperative sensing, routing, and topology management. A typical requirement for a CH is that it must be connected to all members of its cluster. In SOC, though CRs of a cluster are not guaranteed to form a clique (for example in Figure 2(c), CR_B, CR_D are not within each other's communication range even though they belong to the same cluster), in the following lemma we prove that at least one CR in the cluster is guaranteed to be within range of every member of its cluster. This CR can be identified in the last step of cluster formation.

Lemma 5: In every cluster produced by SOC, at least one CR is one-hop away from all other CRs of that cluster.

Proof: The proof is provided in Appendix 6. □

For the CRN in Figure 3, CR_A, CR_E, and CR_H can be selected as CHs.

6 Dynamic Control Channel Assignment

Once clusters are formed, control channels must be selected from the common idle channel list within each cluster. This assignment can be facilitated by CHs. From an architectural standpoint, the assignment of different control channels to various clusters poses two major challenges.

Inter-cluster coordination problem–Consider the clustering in Figure 3. Suppose that CR_A, CR_E, and CR_H serve as CHs. For the three formed clusters, supposed that channels 1, 7 and 8 are selected for control respectively. Assume now that CR_G wants to send a control message to CR_F. Since channel 7 is not in the idle list of CR_G, the two CRs cannot exchange control messages despite the fact that channels $\{2, 4\}$ are common to both of them.

Control channel migration problem–For the CRN in Figure 3, suppose that a PR starts transmitting over channel 1, and only CR_B senses this PR activity (other CRs may be out of range of the transmitting PR or may not be sensing the channel). CR_B needs to notify the other CRs in its cluster that channel 1 is no longer idle. Since channel 1 is used for control, a notification sent on this channel will interfere with the PR transmission. Even if this interference is considered negligible due to its short duration, CR communication on channel 1 may not be possible due to the

PR activity. To migrate the control channel, the CH node CR_A has to correctly receive the notification from CR_B and determine a new control channel for the nodes in its cluster.

6.1 Periodic Control-Channel Rotation

To allow for inter-cluster communication and to coordinate control-channel migration, we propose the following periodic channel-rotation mechanism. Rather than selecting one channel for control until PR activity appears on it, the control channel is rotated among the common idle channels within each cluster. Let $W_i = \{f_1^{(i)}, f_2^{(i)}, \ldots, f_{|W_i|}^{(i)}\}$ denote the set of common channels in cluster i. For time slots $t = 1, 2, \ldots$, CRs within cluster i use channel $f_j^{(i)}$ where $j = [(t-1) \pmod{|W_i|}] + 1$. The channel hopping mechanism is similar to the hopping used in the neighbor discovery mechanism. However, CRs hop only through the list of common channels within their cluster and channel schedules between clusters may be different. To illustrate, consider the CRN of Figure 3. For the cluster $\{CR_A, CR_B, CR_C, CR_D, CR_G\}$, the set of common channels is $W_A = \{1, 2, 3\}$. The (slot, control channel) pairs for the first few slots are (1,1), (2,2), (3,3), (4,1)... Similarly, for cluster $\{CR_E, CR_F\}$, the (slot, control channel) pairs are (1,2), (2,5), (3,7), (4,2)...

When a CR senses PR activity on the current control channel, it waits until the control channel is migrated to an idle one before notifying other CRs within the same cluster. For example, in Figure 3, suppose that CR_B senses PR activity on channel 1. When the control channel hops to channel 2, CR_B notifies its CH (CR_A) of its new idle channel list. Then, CR_A updates the list of common channels to $W'_A = \{2, 3\}$ and broadcasts W'_A, using either channel 2 or 3. The control channel now rotates only between channels 2, and 3.

The rotation of the control channel also addresses the problem of inter-cluster coordination. Two neighboring CRs that belong to two different clusters can communicate with each other as long as the two clusters have at least one idle channel in common. For example, if CR_G is aware of the common channel list $W_E = \{2, 5, 7\}$ of the cluster $\{E, F\}$, it can use time slots t, where $(t-1) \equiv 1 \pmod 3 + 1$, to communicate control information on channel 2. To enable inter-cluster coordination, CRs use the broadcast of Q_j^3 from their neighbors to obtain the common channel list of adjacent clusters and derive their channel schedule. The above rotation mechanism implements an always-on virtual channel for control, located at different frequency bands in various time slots and clusters. The location of the control channel is known to all CRs within each cluster.

6.2 Reclustering

Although SOC converges after only a few messages are exchanged, it is desirable to limit frequent recomputation of clusters in order to reduce the communication overhead for forming new clusters, the traffic relay, and temporary disconnections. In SOC, the availability of multiple idle channels reduces the need for reclustering. The set of common idle channels is updated in accordance with PR activity. However, it may happen that a cluster is left without any common channel for some time, due to low idle channel availability. If this time is small, cluster members may temporarily switch to the coordination protocol until sufficient channels are freed again. A reclustering operation can be triggered periodically to account for the long-term dynamics of PR activity and changes in the CRN topology. In Section 8.3.3, we show that in SOC, only a small fraction of clusters are left without any common idle channel.

7 Coordination Without a Control Channel

During the execution of the SOC algorithm, neighboring CRs need to exchange their lists of idle channels. This exchange has to occur in the absence of a common control channel, because such a channel is not yet established. In this section, we propose a coordination protocol for CRNs that facilitates the exchange of broadcast information. Our mechanism relies on a combination of well established principles of multiple access such as time division and random access. Note that several coordination mechanisms that do not require the existence of a control channel are known for fixed spectrum networks (e.g., [3], [28]), but their adaptation to CRNs is not straightforward. The work most relevant to ours is the quorum channel hopping (QCH) system proposed in [7]. We compare the performance of our protocol with the scheme in [7], in Section 8.6.

7.1 Protocol Overview

Consider an arbitrary node CR_i. The steps of our coordination protocol are as follows:

1) CR_i determines C_i using spectrum sensing.
2) CR_i broadcasts its list C_i on channel $f_j^{(i)} \in C_i$ during slots $t = 1, 2, ...$, if the following relation is satisfied: $f_j^{(i)} = [(t-1) \pmod M] + 1$. Broadcasting is done according to a random access protocol. Any CR_ℓ that hears CR_i's transmission places CR_i in its neighbor list, denoted as NB_ℓ.
3) CR_i exchanges clustering information (explain in Section 5) with every neighbor $CR_\ell \in NB_i$ using the channel schedule derived from C_ℓ (e.g. on channel $f_j^{(i)}$ at time slot $kM + f_j^{(i)}, k = 0, 1, ...$, if they both see channel $f_j^{(i)}$ as available), until a common control channel is set up.

In Step 2, a universal time schedule for channel access is followed, regardless of nodes' individual views of channel availability. Each time slot t is mapped to a channel $j \in \mathcal{M}$ by a modulo-M operation. For example, for the CRN in Figure 1, the (slot, channel) pairs during which the broadcast in Step 2 is allowed to take place are (1,1), (2,2) (3,3), (4,4), (5,1),... CR_A can communicate with CR_C and CR_D on channel 3 at time slots $t = 3, 7, 11$,... Using this universal schedule, CRs can discover their neighbors and exchange channel information.

Our neighbor discovery protocol requires all CRs to be time-synchronized. For the purpose of neighbor discovery, a time slot corresponds to the time that CRs operate on one idle channel. The length of a time slot can be made appropriately large, to allow for the discovery of all CRs that are tuned to the same channel. Access to each frequency band can occur in a random fashion following a CSMA model [1]. Though random access protocols

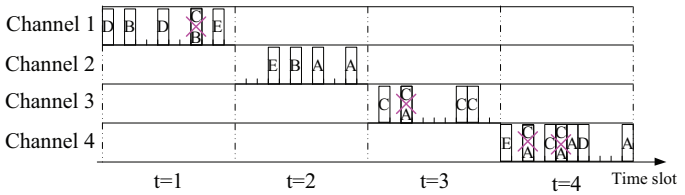


Fig. 4. A realization of the coordination protocol for the CRN of Figure 1.

have relatively low throughput, they are preferred here because they do not require any node coordination.

Each time slot is divided into mini-slots. Each mini-slot is long enough to allow CR_i to broadcast its list of idle channels. CRs tuned to the same channel broadcast their channel lists at each mini-slot with probability P_{access}. If a CR chooses to stay silent in a given mini-slot (with probability $1-P_{access}$), it will listen to other transmissions and record their announced channel lists. If more than one CR chooses the same mini-slot for transmission, a collision occurs. Note that in typical wireless communications, broadcast messages are not acknowledged in order to avoid the ACK implosion problem. Because of the absence of feedback regarding the reception of a broadcast message, contending nodes continue to access mini-slots with probability P_{access}, regardless of the success of their transmission.

One realization of the above coordination process for the CRN of Figure 1 is shown in Figure 4. There are four available channels. Each time slot is divided into 12 mini-slots. For time slot 1, we have $[(1-1) \pmod 4] + 1 = 1$, so nodes whose channel lists include channel 1 (CR_D, CR_B and CR_E) tune to this channel. These nodes contend at various mini-slots in slot 1. Even though a collision occurs during the 9th mini-slot, CR_D, CR_B and CR_E are still able to successfully broadcast their available channels in mini-slots 1,3,6,11 of slot 1. During time slot 2, CR_A, CR_B and CR_E tune to channel two, and announce their available channels without any collision. All of them identify each other as neighbors. This channel access mechanism is maintained until a control channel is established.

8 Performance Evaluation

In this section, we demonstrate the agility of our clustering algorithms in adapting to PR dynamics. We first investigate the performance of C-SOC for different threshold values. Then, we demonstrate the advantages of SOC and C-SOC over clustering methods that do not take into account PR activity. Moreover, we evaluate the rate of reclustering due to PR activity for various clustering methods. We then evaluate the performance of the periodic control-channel rotation mechanism, and study the effectiveness of Algorithms 1 and 2 in finding bicliques that are close to the optimal ones. Finally, we evaluate the performance of the coordination protocol, and compare it with the QCH system proposed in [7].

8.1 Evaluation Setup

In our evaluation, we consider a CRN that co-exists with a cellular PRN. The parameters for each of the two networks as well as the evaluation metrics are described below.

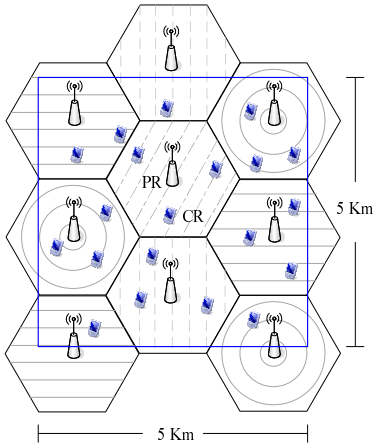


Fig. 5. Evaluation setup consisting of a cellular PRN and a CRN. Ten channels are assigned per cell. Adjacent cells do not share any channels.

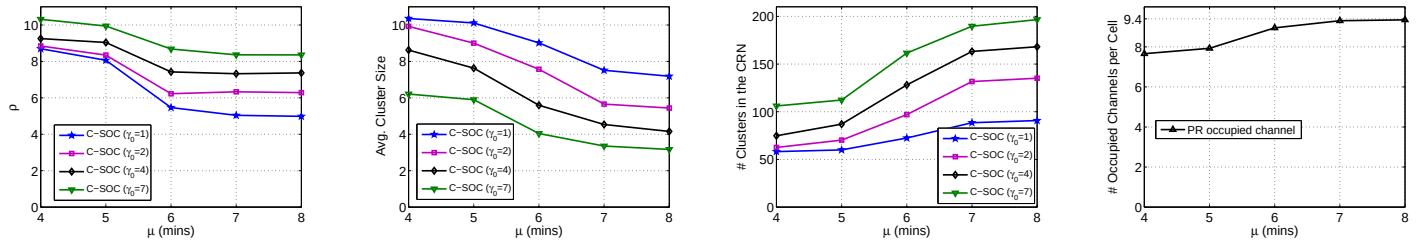
Cellular primary network setup–The cellular network consists of nine cell towers covering an area of 5Km×5Km, as shown in Figure 5. 40 frequency channels are assigned to the PRN, according to the four-color theorem [26]. Accordingly, each cell is assigned 10 channels, with adjacent cells operating over non-overlapping channels. This is illustrated in Figure 5 by the different shading on the various cells. The communication range for each cell tower is set to 1.25 Km. For each cell, calls arrive at each channel according to a Poisson process of rate λ calls/min. We assume an exponentially distributed call holding time with parameter μ minutes.

CRN Setup–CRs are randomly deployed in the area covered by the cellular network. They are assumed to be fixed. The CR communication range is set to 500 m. Each CR_i senses the set of idle channels C_i based on the cell it is located in. A CR is not allowed to access channels occupied by the cell it resides in or by adjacent cells. An imperfect sensing process is assumed, whereby the status of a channel at each CR is misdetected with probability p_f. The lists of idle channels are updated at each CR every time a new event (call arrival or call termination) occurs in a cell.

Clustering Schemes–We compare SOC and C-SOC against three clustering schemes: (a) the distributed clustering algorithm (DCA) [6], (b) the lowest-id clustering algorithm (LCA) [5], and (c) the distributed coordination scheme proposed in [31]. We will refer to the latter as DCRN. In DCA, a node is elected as a CH if it has the highest weight among its neighbors. Each CR associates itself with a neighboring CH that has the highest weight. We set the weight of a CR to its degree on the connectivity graph. In LCA, a node becomes a CH if it has the lowest id within its neighborhood. Finally, in DCRN, CRs select the channel common to the largest number of neighboring CRs for control. CRs may belong to multiple clusters at the same time, if they utilize more than one control channel to connect to their neighbors.

Evaluation Metrics:

- *Average number of common idle channels per cluster*–This metric, denoted by ρ, captures channel availability in a cluster. A larger value of ρ enables control-channel migration



(a) (b) (c) (d)

Fig. 6. Performance of C-SOC as a function of the call duration μ for different values of γ_0: (a) average number of common channels per cluster (ρ), (b) average cluster size, (c) average number of clusters in the CRN, and (d) number of occupied channels by PR per cell.

(in case of PR activity) with less likelihood of reclustering. It also implies that higher bandwidth is available for intra-cluster communications.

- *Coefficient of variation (CV) for the number of common idle channels*–The CV is defined as the ratio of the standard deviation over the mean value. This metric captures the uniformity on the availability of common idle channels per cluster. A low CV implies more uniformity among clusters.
- *Percentage of clusters with no common idle channels*–Poor clustering decisions can lead to clusters with no common idle channels. This metric captures the percentage of clusters whose members do not share any channels.
- *Number of clusters in the network*–The partitioning of the CRN into a large number of clusters increases the overhead for inter-cluster coordination.
- *Average cluster size*–This metric represents the average number of CRs that belong to a cluster.
- *CV of the average cluster size*–This metric captures the uniformity of the constructed clusters in terms of number of nodes per cluster.

8.2 Evaluation of the C-SOC Algorithm

In this set of experiments, 600 CRs are deployed. We fix λ at 1.5 calls/min, and vary μ from $\mu = 4$ mins to $\mu = 8$ mins. We first investigate the effect of the threshold value γ_0 on the number of common idle channels per cluster. As shown in Figure 6(a), ρ is larger than γ_0 for every value of μ. This effect can be explained by two factors. For small μ's, nodes within the same neighborhood share many more channels than γ_0. The threshold γ_0 does not have a significant effect on the clustering process. Thus, the neighborhood size is the limiting factor in cluster size. Indeed, in Figure 6(b), we observe very small differences in the cluster sizes for different γ_0. The impact of γ_0 becomes apparent when γ_0 is large (e.g., $\gamma_0 = 7$) or when μ is large. As μ increases, fewer channels become available for CR user, and hence, the common idle channel availability within each cluster drops. Nonetheless, ρ remains above γ_0 for all values of μ.

The threshold requirement in C-SOC has adverse impact on the cluster size and the number of clusters. To maintain ρ above γ_0, C-SOC creates smaller size clusters, leading to a partitioning of the CRN into a large number of clusters. This is depicted in Figure 6(c). In fact, for $\gamma_0 = 7$, the number of clusters is almost twice as large when $\mu = 8$ mins compared to $\mu = 4$ mins. This increase can be explained in conjunction with Figure 6(d), which shows the PR channel occupancy as a function of μ. We observe that for large μ, there are not enough idle channels to satisfy a high γ_0. This leads to the creation of many single-node clusters.

8.3 Comparison of SOC/C-SOC with Other Schemes

8.3.1 Variation in PR activity

In this set of experiments, we vary the PR activity by varying the average call duration μ. In Figure 7(a), we depict ρ as a function of μ. The threshold for C-SOC is set to $\gamma_0 = 2$. We observe that SOC and C-SOC maintain a larger value of ρ for all μ's. This advantage is demonstrated in the CV for the number of idle channels, and the fraction of clusters with no common channels. Figure 7(b) shows this CV as a function of μ. We observe a much smaller variation in channel availability for SOC and C-SOC compared with the other schemes. This behavior is essential to guarantee sufficient idle channels for migration when a PR appears on the current control channel. Hence, frequent reclustering is avoided.

When PR activity is high, clustering methods that do not take into account channel availability create many clusters with no common idle channels. This is illustrated in Figure 7(c), which shows the fraction of clusters with no common idle channels as a function of μ. We observe that for $\mu = 8$ mins, 35% of the clusters created using LCA and DCA do not share any common idle channels. SOC, C-SOC, and DCRN mitigate this problem, due to their spectrum-aware nature.

To increase ρ, SOC and C-SOC adjust the cluster size. This is verified in Figures 7(d) and 7(e), which depict the average cluster size and the number of clusters in the CRN as a function of μ, respectively. We observe that the average cluster size decreases with μ to compensate for the reduction in idle channel availability. In turn, this leads to the creation of more clusters, as shown in Figure 7(e). Figure 7(f) shows the CV for the cluster size under various schemes. A higher variation is observed for SOC, C-SOC and DCRN. This is due to the spatial adaptation of the cluster size to PR activity. One critical observation is that C-SOC behaves like DCA for low μ, yielding large cluster sizes and low CV values. when PR activity is low.

8.3.2 Variation in Node Density

In this set of experiments, we vary the node density by varying the number of deployed CRs. The call arrival rate is set to $\lambda = 1.5$ calls/min per cell, with the mean call duration fixed to 6

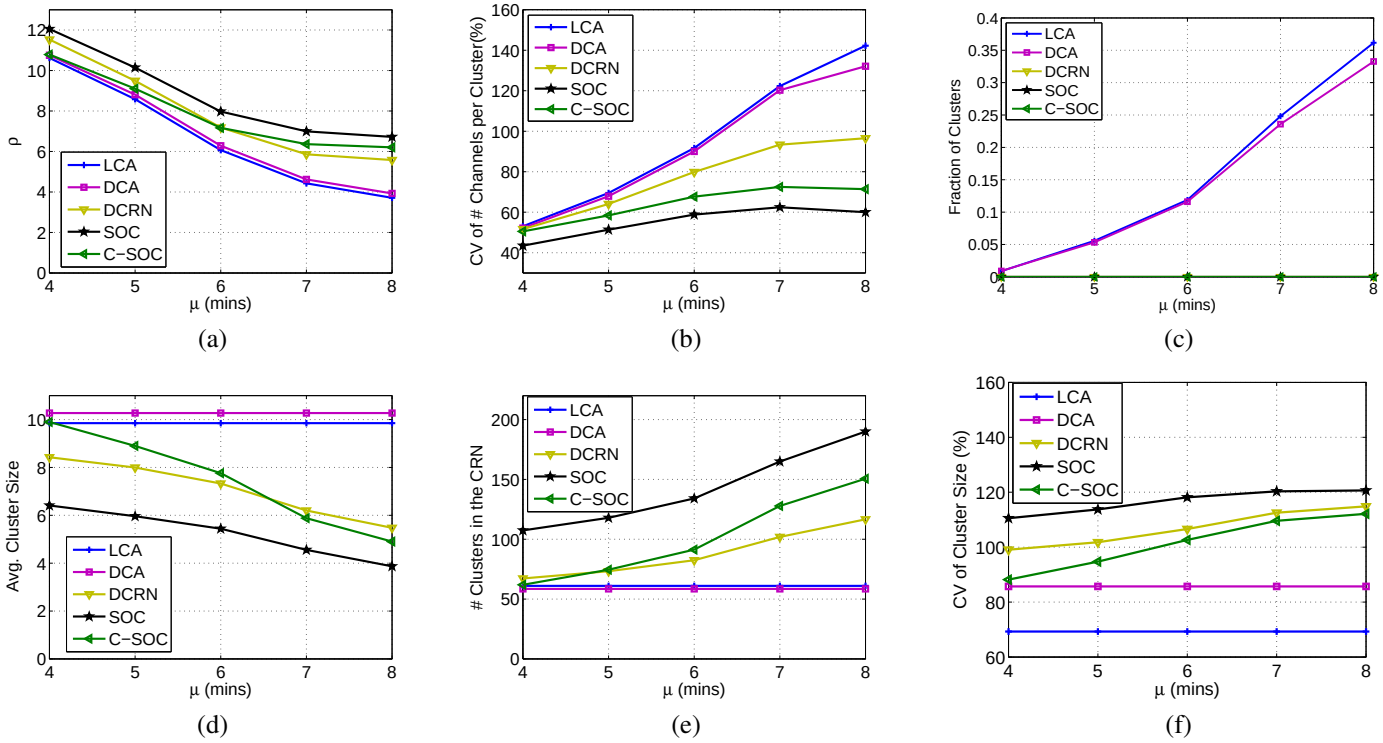


Fig. 7. Performance of various clustering schemes vs. call duration μ: (a) average number of common idle channels per cluster, (b) CV of the number of common channels, (c) fraction of clusters with no common idle channels, (d) average cluster size, (e) average number of clusters in the CRN, (f) CV of the cluster size.

mins. Higher node density leads to more flexibility in clustering, which can be potentially exploited to maintain a large number of common idle channels per cluster. In Figure 8(a), we depict ρ as a function of the number of CRs. Observe that for spectrum-aware solutions, ρ is almost insensitive to changes in node density, while decreasing for LCA and DCA. As shown in Figure 8(b), the CV for the common channels under SOC and C-SOC remains constant with variations in node density. Moreover, Figure 8(c) shows that for the same PR activity, the fraction of clusters with no common idle channels increases with the number of deployed CRs for LCA and DCA, up to about 18%. SOC, C-SOC, and DCRN do not produce any cluster with no common idle channels.

In Figures 8(d),(e),(f), we respectively show the average cluster size, the number of clusters in the CRN, and the CV for the cluster size, all as functions of the number of deployed CRs. Under SOC and C-SOC, the number of clusters increases almost linearly with the number of deployed CRs, in order to maintain a stable value of ρ. Moreover, our algorithms exhibit a larger variability in cluster size, as they adapt to variations in spectrum opportunities.

8.3.3 Frequency of Reclustering

In this set of experiments, we investigate the rate of reclustering due to PR activity. We first partition the CRN into clusters based on a snapshot of the channel availability. We then fix λ to 1.5 calls/min and vary the value of μ.

In Figure 9(a), we show the average fraction of time $\mathrm{E}(f_t)$ where *at least* one cluster with no common idle channels exists. We observe that as the PR activity increases, several clusters created based on a snapshot of PR activity stay without a control channel. SOC and C-SOC provide the best performance out of all clustering algorithms. Even when $\mu = 8$ mins, every cluster created using SOC has at least one common idle channel 86% of the time, in contrast to a 63% for LCA and DCA, and 75% for DCRN. C-SOC with $\gamma_0 = 1$ behaves almost as DCRN since only one idle channel is required per cluster. C-SOC yields a significant improvement for $\gamma_0 = 3$, for which 83% of the time clusters have at least one common idle channel.

Reclustering is greatly impacted by the outage duration of the control channel, defined as the period of time that cluster stays without a common idle channel. Figure 9(b) shows the maximum outage duration as a function of μ. We observe that SOC and C-SOC have much shorter outage durations compared with other schemes, thus avoiding reclustering. Note that for $\mu = 8$ almost all channels within each cell are occupied by the PR, as shown in Figure 6(d). Hence, the outage is mainly caused by the lack of available channels, rather than poor clustering decisions.

8.4 Periodic Control-Channel Rotation

In this set of experiments, we evaluate the periodic control-channel rotation mechanism which is employed for inter-cluster communications. In Figure 9(c), we show the average fraction of time $\mathrm{E}(f_t)$ that a cluster is reachable by CRs in adjacent clusters, as a function of μ when $\lambda = 2$ calls/min. We observe that when the PR activity is low (small values of μ), our rotation mechanism can maintain inter-cluster communication more than 90% of the time. This is because under low PR activity, the majority of the channels used for channel rotation within a cluster are also idle in adjacent clusters. On the other hand, under high PR activity (large values of μ), the heterogeneity between the sets of idle channels

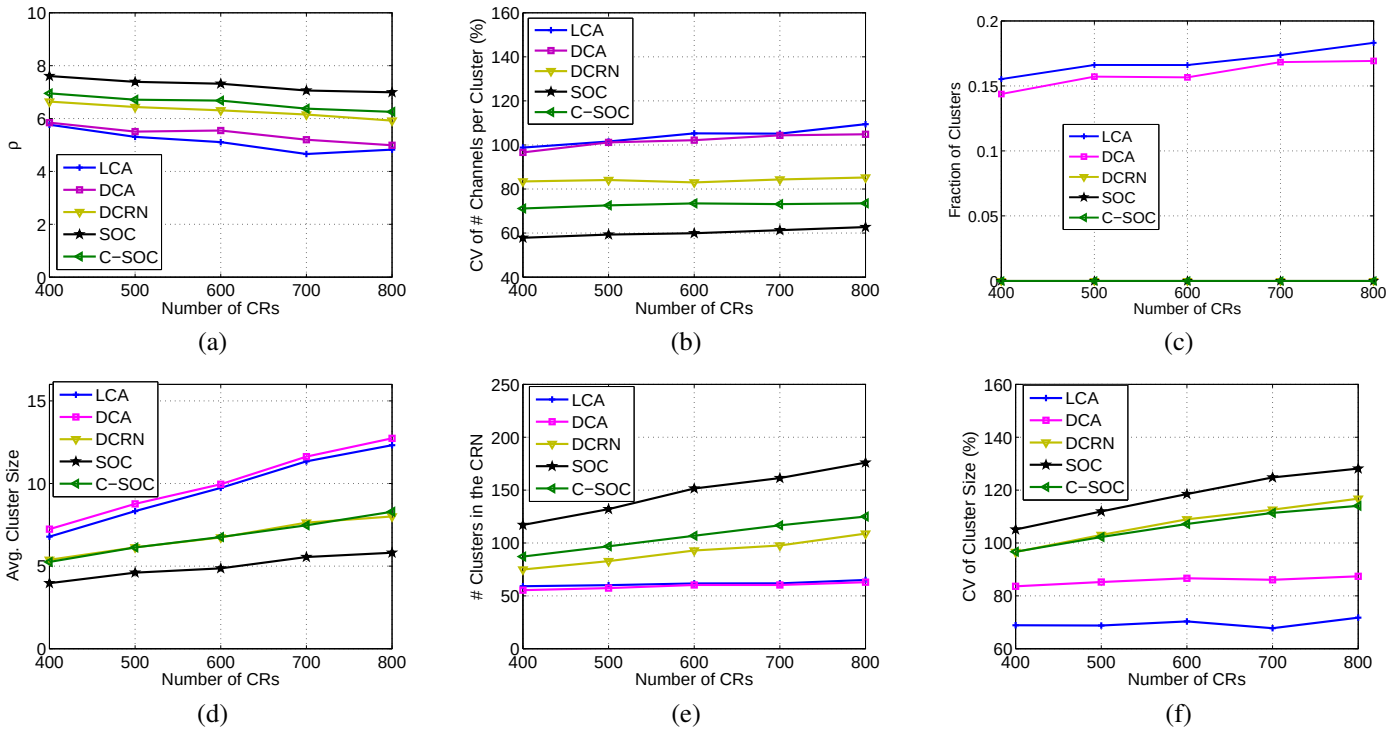


Fig. 8. Performance of various clustering schemes vs. node density: (a) average number of common idle channels per cluster, (b) CV of the number of common channels, (c) fraction of clusters with no common idle channels, (d) average cluster size, (e) average number of clusters in the CRN, (f) CV of the cluster size.

of neighboring CRs increases. To adjust to this change, SOC creates clusters of smaller size in order to maintain a larger set of common idle channels within each cluster. In this case, the overlap between the set of common idle channels of a cluster and the set of idle channels of CRs in adjacent clusters tends to be smaller. Nevertheless, over 78% of the time, inter-cluster communication is still feasible. Note that when μ is large, there are periods of time where inter-cluster communication is not possible simply because all available channels are occupied by PRs. For such periods of high activity, inter-cluster communication is limited by the opportunistic nature of the CRNs and is not related to the channel rotation mechanism.

From Figure 9(c), we also observe that C-SOC with $\gamma_0 = 1$ yields a higher value of $\mathrm{E}(f_t)$ compared to the case of $\gamma_0 = 3$ and to SOC. This difference can be explained as follows. For small values of γ_0, the network partitioning is primarily decided by its physical topology. To maximize the cluster size, C-SOC produces clusters with a small number of common idle channels. These channels are, therefore, likely to be seen idle by a large number of CRs, including CRs in adjacent clusters. Restricting channel rotation to those widely available channels increases the fraction of time that inter-cluster communication is possible. On the other hand, in SOC, it is more likely that during channel rotation, a channel is common among the cluster members but is not available to CRs in adjacent clusters (this is the reason why these members were excluded from that cluster in the first place). In Figure 9(d), we show $\mathrm{E}(f_t)$ as a function of the number of CRs in the network We observe that $\mathrm{E}(f_t)$ almost remains constant with the increase of the number of CRs. This is due to the fact that the spatial variation in PR activity is not affected by the number of deployed CRs.

Figures 9(e) and 9(f) show the maximum outage duration in inter-cluster communications, measured as the maximum time (in minutes) that *any* two adjacent clusters are unable to communicate during the course of the simulation. Outages in inter-cluster communications can occur when: (a) a cluster is left without any common idle channels due to PR activity, or (b) the set of common idle channels used for channel rotation does not overlap with the sets of idle channels for CRs in an adjacent cluster. In our simulations, we only measured outages due to scenario (b), since outages due to (a) are already evaluated in Section 8.3. From Figure 9(e), we observe that the maximum outage time is limited to small values ($<$1.5 minutes) even when $\mu = 8$ min. Moreover, Figure 9(f) illustrates a slightly increasing maximum outage time when the CR density is increased. This is because the probability of finding a CR that does not share common channels with an adjacent cluster increases with the increase of the number of border nodes.

8.5 Performance of Algorithms 1 and 2

In this section, we evaluate the performance of our greedy heuristics. Because the problems of finding the maximum edge biclique and the maximum one-sided node cardinality are NP-complete, we obtain optimal solutions via exhaustive search for small bipartite graphs. We randomly generate bipartite graphs of sizes 5×5 and 10×10. In each bipartite graph, an edge between a pair of vertices exists independently with probability P_{edge}, which is varied from 0.2 to 0.8. Though our experiments are limited in the size of the bicliques, they are are nonetheless useful for typical neighborhood sizes and number of channels. In Figure 10(a), we

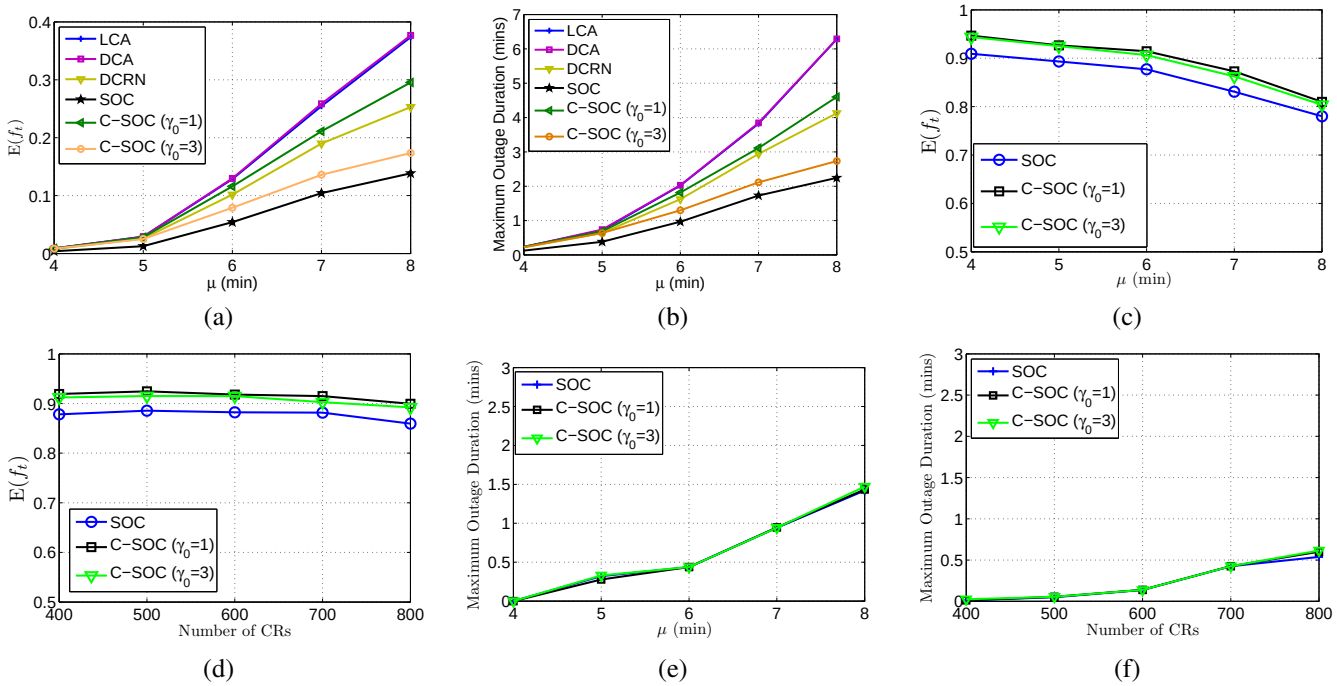


Fig. 9. (a) Fraction of time that at least one cluster exists without a common idle channel as a function of μ, (b) maximum duration of the control channel outage as a function of μ, (c) $\mathrm{E}(f_t)$ as a function of μ, (d) $\mathrm{E}(f_t)$ as a function of the number of CRs, (e) maximum outage duration for inter-cluster communication as a function of μ, and (f) maximum outage duration for inter-cluster communication as a function of the number of CRs.

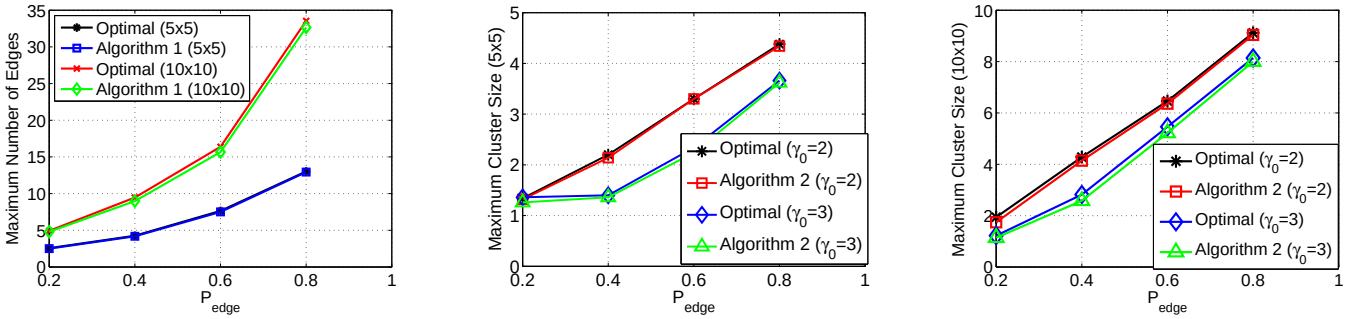


Fig. 10. Comparison of Algorithms 1 and 2 with the optimal solution as a function of the probability of edge existence P_{edge}: (a) Algorithm 1 for 5×5 and 10×10 bipartite graphs, (b) Algorithm 2 for 5×5 bipartite graphs and for $\gamma_0 = 2, 3$, (c) Algorithm 2 for 10×10 bipartite graphs and for $\gamma_0 = 2, 3$.

show the number of edges in bicliques obtained by Algorithm 1 when performing an exhaustive search, averaged over 50 bipartite graphs, as a function of P_{edge}. We observe that for the considered bipartite graphs, our greedy heuristic is near-optimal. In Figure 10(b),(c) we show the cluster size obtained by Algorithm 2 and through exhaustive search, averaged over 50 bipartite graphs, as a function of P_{edge}, and for threshold values $\gamma_0 = 2, 3$. Once again, our greedy heuristic is near-optimal.

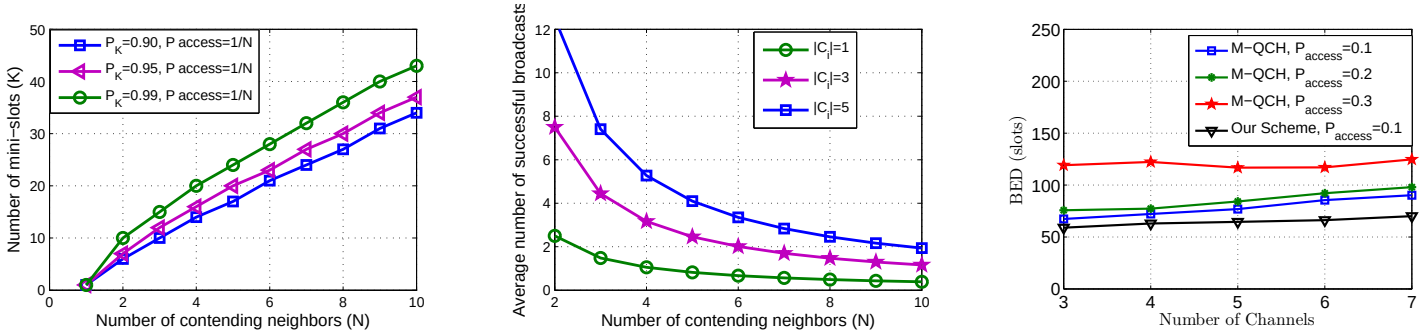
8.6 Evaluation of The Coordination Protocol

8.6.1 Analytical evaluation

We first compute the number of mini-slots needed, so that on average, every contending CR is able to perform one successful broadcast per time slot. Let N be the number of contending neighbors and let K be the number of mini-slots per time slot. Without loss of generality, we take $N < K$. Let P_{mini} denote the probability of a successful transmission in a given mini-slot. Because each contending node independently accesses a mini-slot with probability P_{access}, $P_{mini} = \binom{N}{1} P_{access} (1 - P_{access})^{N-1}$. Simple calculations show that P_{mini} is maximized when $P_{access} = 1/N$, i.e., when on average, exactly one CR attempts a transmission in a mini-slot. The probability of at least N successes in K mini-slots is:

$$P_K = \sum_{i=N}^{K} \binom{K}{i} P_{mini}^i (1 - P_{mini})^{K-i}. \quad (3)$$

One possible way to select K is to impose a lower bound on P_K. In Figure 11(a), we plot K as a function of N for different values of P_K, with P_{access} set to $1/N$. We observe that the minimum number of mini-slots that is necessary to



(a) (b) (c)

Fig. 11. (a) Number of mini-slots versus N for a given P_K with $P_{access} = 1/N$, (b) expected number of successful broadcasts after M time slots versus N ($K = 10$), (c) the BED as a function of the number of available channels under a dynamic spectrum scenario with $\lambda = 2$ calls/min and $\mu = 0.5$ mins.

guarantee a certain P_K increases approximately linearly with N. Approximately 15 mini-slots are enough when $N = 5$ with $P_K = 0.9$, while 35 mini-slots are needed when $N = 10$.

The neighbor discovery phase ends after M time slots, at which point all possible channels will have been scanned. The number of successful broadcasts for a single node CR_i in M time slots is binomially distributed with mean of $K|C_i|P_{access}\,(1 - P_{access})^{N-1}$.

Figure 11(b) shows this mean for different $|C_i|$, when $K = 10$. When CR_i has more than three available channels, it can make at least one successful broadcast even if there are nine other contending CRs. When on average only one channel is available, CR_i can still broadcast successfully if less than four CRs are contending. In this case, a large value of K is required.

Although in general channel availability is not expected to change during the neighbor discovery phase, a CR may still detect new PR activity on channel j during time slot t. In this case, the CR vacates this channel, updates its channel list, and uses other idle channels to continue exchanging channel information. After one-hop neighbors are discovered, CRs use the time schedules of their neighbors to coordinate the clustering and CCA process.

8.6.2 Comparison to the QCH system

Similar to our scheme, the QCH system proposed in [7] can be used for the coordination of CRs in the absence of a control channel. To compare the performance between the two schemes, we have measured the number of slots needed until every node within the same collision domain communicates one message to all its neighbors (our control channel assignment scheme requires that all neighbors exchange their set of idle channels during the coordination phase). We refer to this delay as the *broadcast exchange delay* (BED).

To provide a fair comparison with the QCH scheme, we have considered the quorum-based design that is most suitable for broadcast communications. According to [7], the channel access delay is minimized when the channel hopping system is constructed based on a majority cyclic quorum. Therefore, the authors suggest using the M-QCH system for the implementation of broadcast control channels. M-QCH has a minimum frame length equal to three and is $k = 3$ and is constructed over $U = Z_3$. We have selected the quorum system for constructing the hopping sequences as $S = \{\{0, 1\}, \{0, 2\}, \{1, 2\}\}$ for our simulations. In this setup, each node is randomly assigned a quorum from S in order to construct its hopping sequence.

To compare the performance of the two schemes under a dynamic spectrum scenario, we considered a PR network in which calls arrive at each channel according to a Poisson process of rate $\lambda = 2$ calls/min. We assumed an exponentially distributed call holding time with parameter $\mu = 0.5$ mins. Figure 11(d) shows the BED as a function of the number of available channels. We observe that our scheme is more efficient than the M-QCH scheme (M-QCH needs around 30% more time-slots to finish broadcast when the number of available channels is large). This difference in performance is attributed to the fact that in our scheme, CRs converge on the same channel in every time slot, thus facilitating the broadcast operation. In contrast, in M-QCH only a subset of nodes converge to the same channel at any time slot.

9 Conclusions

We addressed the problem of CCA in CRNs. We adopted a dynamic allocation policy in which the control channel is dynamically assigned according to PR activity. We mapped the clustering problem into instances of a bipartite graph problem, and showed that this mapping allows for a graceful tradeoff between the cluster size and the set of common channels in each cluster. In particular, we mapped the clustering process to the maximum edge biclique problem, and the maximum one-sided edge cardinality problem. Since both problems are known to be NP-complete, we proposed two greedy heuristics for finding bicliques that satisfy our requirements. We proposed two distributed clustering algorithms called SOC and C-SOC that takes into account the channel availability in deciding cluster memberships. We further proposed a control channel rotation mechanism that enables control channel migration in case of PR activity, inter-cluster communication, and adaptation to the temporal variations of spectrum availability.

Acknowledgments

This research was supported in part by NSF (under grants CNS-1016943, CNS-0844111, CNS-0721935, CNS-0904681, IIP-0832238), Raytheon, and the "Connection One" center. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] N. Abramson. The aloha system: another alternative for computer communications. In *Proc. of American Federation of Information Processing Societies (AFIPS'07) Conference*, volume 70, pages 281–285, November 1970.

[2] I. F. Akyildiz, W. Y. Lee, M. C. Vuran, and S. Mohanty. NeXt generation/dynamic spectrum access/cognitive radio wireless networks: A survey. *Computer Networks Journal (Elsevier)*, 50(13):2127–2159, September 2006.

[3] P. Bahl, R. Chandra, and J. Dunagan. SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *Proc. of the 10th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 216–230, 2004.

[4] P. Bahl, R. Chandra, T. Moscibroda, R. Murty, and M. Welsh. White space networking with Wi-Fi like connectivity. In *Proc. of the ACM SIGCOMM 2009 Conference*, pages 27–38, 2009.

[5] D. Baker, A. Ephremides, and J. Flynn. The design and simulation of a mobile radio network with distributed control. *IEEE Journal on Selected Areas in Communications*, 2(1):226–237, 1984.

[6] S. Basagni. Distributed clustering for ad hoc networks. In *Proc. of I-SPAN*, pages 310–315, June 1999.

[7] K. Bian, J. Park, and R. Chen. A quorum-based framework for establishing control channels in dynamic spectrum access networks. In *Proc. of MobiCom*, pages 25–36, 2009.

[8] T. Brown. An analysis of unlicensed device operation in licensed broadcast service bands. In *Proc. of DySPAN*, pages 11–29, Nov 2005.

[9] D. Čabrić, S. Mishra, D. Willkomm, R. Brodersen, and A. Wolisz. A cognitive radio approach for usage of virtual unlicensed spectrum. *Proc. of the 14th IST Mobile and Wireless Communications Summit*, 2005.

[10] T. Chen, H. Zhang, M. Katz, and Z. Zhou. Swarm intelligence based dynamic control channel assignment in CogMesh. In *Proc. of ICC*, pages 123–128, 2008.

[11] T. Chen, H. Zhang, G. Maggio, and I. Chlamtac. CogMesh: A cluster-based cognitive radio network. In *Proc. of DySPAN*, pages 168–178, 2007.

[12] M. Dawande, P. Keskinocak, J. Swaminathan, and S. Tayur. On bipartite and multipartite clique problems. *Journal of Algorithms*, 41(2):388–403, 2001.

[13] M. Dawande, P. Keskinocak, and S. Tayur. On the biclique problem in bipartite graphs. Technical report, GSIA Working Paper, 1996-04. Carnegie Mellon University, 1996.

[14] FCC. Spectrum Policy Task Force Report, 2002.

[15] FCC. OET Bulletin No.69–Longley-Rice Methodology for evaluating TV Coverage and Interference. *Federal Communications Commission*, 2004.

[16] S. Feng, H. Zheng, H. Wang, J. Liu, and P. Zhang. Preamble design for non-contiguous spectrum usage in cognitive radio networks. In *Proc. of WCNC*, 2009.

[17] G. Ganesan and Y. Li. Cooperative spectrum sensing in cognitive radio networks. In *Proc. of DySpan*, pages 137–143, 2005.

[18] A. Ghasemi and E. Sousa. Collaborative spectrum sensing for opportunistic access in fading environments. In *Proc. of DySpan*, pages 131–136, November 2005.

[19] C. Han, J. Wang, Y. Yang, and S. Li. Addressing the control channel design problem: OFDM-based transform domain communication system in cognitive radio. *Computer Networks Journal*, 52(4):795–815, 2007.

[20] J. Jia, Q. Zhang, and X. Shen. HC-MAC: a hardware-constrained cognitive MAC for efficient spectrum management. *IEEE Journal on Selected Areas in Communications*, 26(1):106–117, 2008.

[21] P. Kondareddy, Y.R. Agrawal. Synchronized MAC protocol for multi-hop cognitive radio networks. In *Proc. of ICC*, pages 3198–3202, 2008.

[22] M. Krondorf, T. Liang, and G. Fettweis. On synchronization of opportunistic radio OFDM systems. In *Proc. of the IEEE Vehicular Technology Conference (VTC'08)*, pages 1686–1690, 2008.

[23] L. Lazos, S. Liu, and M. Krunz. Mitigating control-channel jamming attacks in multi-channel ad hoc networks. In *Proc. of WiSec*, pages 169–180, 2009.

[24] S. Mishra, A. Sahai, and R. Brodersen. Cooperative sensing among cognitive radios. In *Proc. of ICC*, June 2006.

[25] R. Peeters. The maximum edge biclique problem is NP-complete. *Discrete Applied Mathematics Journal*, 131(3):651–654, 2003.

[26] N. Robertson, D. Sanders, P. Seymour, and R. Thomas. The four colour theorem. *Journal of Combinatorial Theory, Series B*, 70(1):2–44, 1997.

[27] H. A. B. Salameh, M. M. Krunz, and O. Younis. MAC protocol for opportunistic cognitive radio networks with soft guarantees. *IEEE Transactions on Mobile Computing*, 8(10):1339–1352, 2009.

[28] J. So and N. Vaidya. Multi-channel MAC for ad hoc networks: handling multi-channel hidden terminals using a single transceiver. In *Proc. of MobiHoc*, pages 222–233, 2004.

[29] K. Sun, P. Peng, P. Ning, and C. Wang. Secure distributed cluster formation in wireless sensor networks. In *Proc. of the Annual Computer Security Applications Conf. (ACSAC'06)*, 2006.

[30] T. Weiss, A. Krohn, F. Capar, I. Martoyo, and F. Jondral. Synchronization algorithms and preamble concepts for spectrum pooling systems. In *Proc. of the IST Mobile and Wireless Telecommunications Summit*, 2003.

[31] J. Zhao, H. Zheng, and G.-H. Yang. Spectrum sharing through distributed coordination in dynamic spectrum access networks. *Wireless Communications and Mobile Computing Journal*, 7(9):1061–1075, 2007.

Sisi Liu received the B.S. and M.S. degree in electrical engineering from University of Electronic Science and Technology of China,Chengdu, China, in 2004 and 2007, respectively. She is currently a research assistant working toward a Ph.D degree at the Advanced networking lab, Department of Electrical and Computer Engineering, University of Arizona, Tucson. Her research interests lie in the areas of cognitive radio-based, ad hoc, and wireless sensor networks with emphasis on network security, physical layer attacks, and misbehavior detection.

Loukas Lazos is an Assistant Professor of Electrical and Computer Engineering at the University of Arizona. He received his Ph.D. in Electrical Engineering from the University of Washington in 2006. In 2007, he was the co-director of the Network Security Lab at the University of Washington. Dr. Lazos joined the the University of Arizona in August 2007. His main research interests are in the areas of networking, security, and wireless communications, focusing on the identification, modeling, and mitigation of security vulnerabilities, visualization of network threats, and analysis of network performance. He is a recipient of the NSF Faculty Early CAREER Development Award (2009), for his research in security of multi-channel wireless networks. He has served and continues to serve on technical program committees of many international conferences and on the panels of several NSF directorates.

Marwan Krunz is a professor of ECE at the University of Arizona. He also holds a joint appointment at the same rank in the CS department. He is the UA site director for Connection One, a joint NSF/state/industry IUCRC cooperative center that focuses on wireless communication systems and networks. Dr. Krunz received his Ph.D. degree in electrical engineering from Michigan State University in 1995. He joined the University of Arizona in January 1997, after a brief postdoctoral stint at the University of Maryland, College Park. He previously held visiting research positions at INRIA, HP Labs, University of Paris VI, and US West (now Qwest) Advanced Technologies. In 2010, he was a visiting researcher at Institute IMDEA and a Chair of Excellence ("Cátedra de Excelencia") at the University of Carlos III, Madrid, Spain. Dr. Krunzs research interests lie in the fields of computer networking and wireless communications, with focus on distributed radio resource management in wireless and sensor networks; protocol design; and secure communications. He has published more than 170 journal articles and refereed conference papers, and is a co-inventor on three US patents. He is a recipient of the NSF CAREER Award (1998). He currently serves on the editorial board for the IEEE Transactions on Network and Service Management. Previously, he served on the editorial boards for the IEEE/ACM Transactions on Networking, the IEEE Transactions on Mobile Computing, and the Computer Communications Journal. He served as a TPC chair for various international conferences, including INFOCOM04, SECON05, and WoWMoM06. He is an IEEE Fellow.