Adaptive Frequency Hopping Algorithms for Multicast Rendezvous in DSA Networks

Mohammad J. Abdel Rahman Hanif Rahbari Marwan Krunz
Department of Electrical and Computer Engineering
University of Arizona, Tucson, AZ 85719
{mjabdelrahman, rahbari, krunz}@email.arizona.edu

***Abstract*—Establishing communications in a dynamic spectrum access (DSA) network requires communicating nodes to "rendezvous" before transmitting their data packets. Frequency hopping (FH) provides an effective method for rendezvousing without relying on a predetermined control channel. FH rendezvous protocols have mainly targeted pairwise rendezvous, using fixed (non-adaptive) FH sequences and assuming a homogeneous spectrum environment, i.e., all nodes perceive the same spectrum opportunities. In this paper, we address these limitations by developing three multicast rendezvous algorithms: AMQFH, CMQFH, and nested-CMQFH. The three algorithms are intended for asynchronous spectrum-heterogeneous DSA networks. They provide different tradeoffs between speed and robustness to node compromise. We use the *uniform k-arbiter* and the *Chinese remainder theorem (CRT)* quorum systems to design our multicast rendezvous algorithms. We also design two "optimal" channel ordering mechanisms for channel sensing and assignment, one for AMQFH and the other for CMQFH and nested-CMQFH. Finally, we develop a proactive out-of-band sensing based dynamic FH (DFH) algorithm for online adaptation of the FH sequences used in the proposed rendezvous algorithms. Extensive simulations are used to evaluate our algorithms.**

***Index Terms*—Control channel, dynamic frequency hopping, dynamic spectrum access, multicast rendezvous, quorum systems.**

I. Introduction

Motivated by the need for more efficient utilization of the licensed spectrum and facilitated by recent regulatory policies [5], significant research has been conducted towards developing cognitive radio (CR) technologies for dynamic spectrum access (DSA) networks. CR devices utilize the available spectrum in a dynamic and opportunistic fashion without interfering with co-located *primary users* (PUs). The communicating entities of an opportunistic CR network are called *secondary users* (SUs).

Establishing a communication link in DSA networks requires nodes to rendezvous for the purpose of exchanging control information, such as common spectrum opportunities, transmission parameters, topology changes, etc. In the absence of centralized control, the rendezvous process needs to be carried out in a distributed manner. To address this problem, many existing MAC protocols for CR networks rely on a dedicated global or group control channel. Presuming a common control channel (CCC) surely simplifies the rendezvous process, but it comes with two main drawbacks. First, a CCC can become a network bottleneck, creating a single point of failure. Second, PU dynamics and spectrum heterogeneity make it very difficult to always maintain a single CCC (a.k.a. the coverage problem [13]).

Frequency hopping (FH) provides an alternative method for rendezvousing without relying on a dedicated CCC. Most existing work on FH designs (e.g., [1]) is based on ad hoc approaches that do not provide any performance guarantees. One way to construct FH sequences in a systematic manner is to use quorum systems [6]. Quorums have been widely used in distributed systems to solve the mutual exclusion problem, the agreement problem, and the replica control problem. Systematic quorum-based approaches for designing FH protocols for control channel establishment have been proposed in [2], [3]. One important advantage of quorum-based FH designs is their robustness to synchronization errors [7]. As will be explained later, some quorum systems, such as *uniform k-arbiter* and *Chinese remainder theorem (CRT)* quorum systems, enjoy certain properties that allow them to be used for asynchronous communications. The approaches in [2], [3] do not intrinsically support *multicast rendezvous*, where all the nodes in a multicast group are required to rendezvous in the same time slot. Furthermore, these protocols are intended for a homogeneous spectrum environment, i.e., SUs are assumed to perceive the same spectrum opportunities. Third, the FH sequences in these protocols are constructed by assigning channels to time slots without considering the temporal PU dynamics over these channels.

Group-based schemes have been proposed to facilitate multicast rendezvous [13]. These schemes are divided into two categories: (i) neighbor coordination schemes (e.g., [4]), where neighboring nodes broadcast their channel parameters to make a group-wide decision, and (ii) cluster-based schemes (e.g., [10]), where nodes group around a cluster head according to their channels availabilities. One drawback of these schemes is the need for the initial step of neighbor discovery prior to establishing a CCC. Also, these schemes incur considerable overhead for maintaining the group-based control channel. Even though these solutions establish a CCC for intra-group communications, the problem of inter-group communications is yet another challenge that remains to be addressed [13].

In [12], the authors proposed an FH-based jamming-resistant broadcast communication scheme, called TDBS. TDBS operates in one of two modes, TDBS-SU and TDBS-AB. In both modes, the broadcast operation is implemented as a series of unicast transmissions, distributed in time and

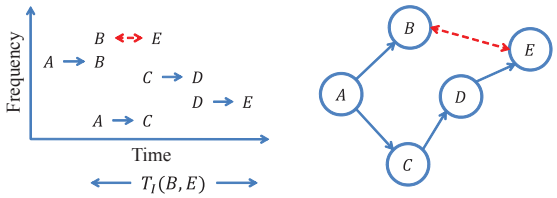


Fig. 1: Multicast as a series of unicasts. Node E receives A's message $T_I(B,E)$ seconds after node B receives it.

frequency. The protocol does not account for PU dynamics during the rendezvous process. Moreover, implementing the multicast operation as a series of unicasts can lead to multicast inconsistency. For example, a group of SUs may share a *group key* that is used to encode/decode common secure communication messages. For security purposes, this key may have to be updated periodically [15]. However, the change in the group key has to be consistent among all SUs in the multicast group. Such consistency cannot be guaranteed if changes in the group key are conveyed using a series of unicast transmissions. Figure 1 shows a network of 5 nodes, where node A needs to send an update message about the group key to nodes B, C, D, and E. If A's message is conveyed sequentially to B, then to C, then to D, and finally to E, using, for example, TDBS-AB, then, nodes B and E will have inconsistent information during the time duration $T_I(B,E)$. Thus, B will not be able to establish a secure communication link with E during this duration.

Instead of designing different FH sequences that overlap at common slots, the multicast rendezvous in [11] is established after a series of pairwise rendezvous operations that result in tuning all nodes in the multicast group to a common FH sequence. The effectiveness of this approach cannot be maintained under node compromise (if one node is compromised, then the FH sequences of all nodes are exposed).

Ignoring PU dynamics leads to excessively long time-to-rendezvous (TTR), as shown in the example in Figure 2. In this example, we consider two FH sequences of two SUs. The time axis is divided into fixed-length frames. The two SUs rendezvous on channel h_1 in the first frame and on channel h_2 in the second frame (h_x refers to a randomly assigned channel). Figure 2(a) shows the case when the rendezvous channels h_1 and h_2 become occupied by PUs during the rendezvous slots (indicated in the arrows) in the first and second frames, respectively. Figure 2(b) depicts the TTR for the FH sequences in [3] as a function of the average channel-time availability for SU communications. The TTR increases as less time is available for SU communications. In the absence of any channel prediction mechanism, Figure 2(c) shows the collision rate with PUs/SUs as a function of the average channel availability for the FH sequences in [3].

In this paper, we develop three multicast rendezvous algorithms, namely AMQFH, CMQFH, and nested-CMQFH, for asynchronous spectrum-heterogeneous DSA networks. We use the uniform k-arbiter and the CRT quorum systems to design our rendezvous algorithms. These two special quorum systems satisfy the *rotation k-closure property*, which enables them to function in the absence of node synchronization. To account for PU dynamics, we also design two "optimal" channel ordering mechanisms for channel sensing and assignment, one for AMQFH and the other for CMQFH and nested-CMQFH. These mechanisms improve the rendezvous process (i.e., reduce TTR) by selecting the top channels as rendezvous channels. The ordering mechanisms are then used in developing a proactive out-of-band sensing based dynamic FH (DFH) algorithm for online adaptation of the FH sequences used in our algorithms.

The remainder of this paper is organized as follows. In Section II, we present the system and channel activity models. Section III presents our proposed AMQFH and CMQFH multicast rendezvous algorithms. Section IV compares the performance of AMQFH with that of CMQFH. In Section IV, we show that AMQFH is much faster than CMQFH, but CMQFH has much larger Hamming distance (HD). In Section V, we introduce nested-CMQFH, which represents a compromise between AMQFH and CMQFH. Section VI introduces our optimal channel ordering mechanisms, followed by our DFH algorithm in Section VII. We evaluate the protocol in Section VIII. Finally, Section IX concludes the paper.

II. System and Channel Models

A. System Model

We consider an opportunistic DSA environment, with L licensed channels, $f_1, f_2, \ldots, f_L$. SUs can hop over these channels if they are not occupied by PUs. Without loss of generality, we assume that FH occurs on a per-slot basis, where the slot duration is T seconds. A packet can be exchanged between two or more nodes if they hop onto the same channel in the same time slot. We assume that one time slot is sufficient to exchange one message. If multiple SU groups happen to rendezvous on the same channel in the same time slot, they use a CSMA/CA-like contention resolution procedure.

For $j = 1, \ldots, k$, each SU j has its unique FH sequence $\boldsymbol{w}^{(j)}$. The channel used in the ith slot of FH sequence $\boldsymbol{w}^{(j)}$ is denoted by $w_i^{(j)}$, $w_i^{(j)} \in \{f_1, \ldots, f_L\}$. Channel f_j is called a *rendezvous frequency* for the FH sequences $\boldsymbol{w}^{(1)}$, $\boldsymbol{w}^{(2)}, \ldots, \boldsymbol{w}^{(k)}$ if there exists a *rendezvous slot* i such that $w_i^{(m)} = f_j, \forall m \in \{1, \ldots, k\}$. As in previous quorum-based FH designs, in our setup each FH sequence consists of several time frames (see Figure 2(a)). Each frame consists of a block of time-frequency hops.

B. Channel Activity Model

We assume that each channel $f_m, m = 1, \ldots, L$, can be in one of three states: idle (state 1), occupied by a PU (state 2), or occupied by an SU (state 3). Transitions between these states are assumed to follow a continuous-time Markov chain (CTMC) with state space $S = \{1, 2, 3\}$, as shown in Figure 3. For any i and $j \in S$, $i \neq j$, we assign a nonnegative number $\alpha_{ij}^{(m)}$ that represents the rate at which channel f_m transitions from state i to state j. Let $\rho_i^{(m)}$ denote the total rate at which

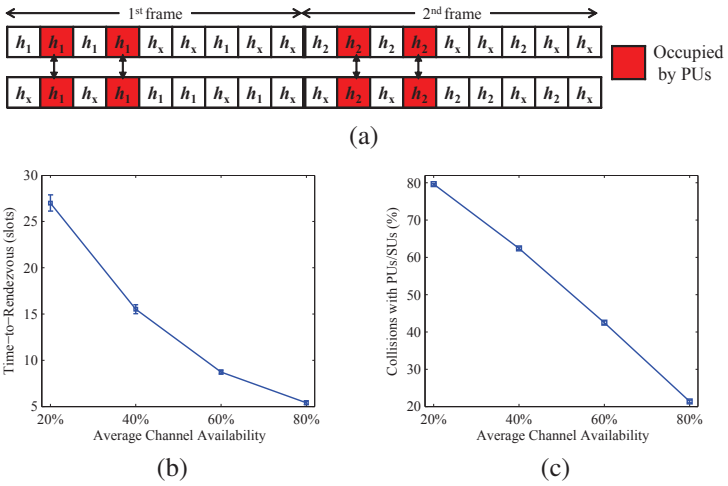


Fig. 2: (a) Two quorum-based FH sequences. Effect of channel dynamics on (b) the time-to-rendezvous and (c) collisions with PUs/SUs (95% confidence interval is shown in the figure).

channel f_m leaves state i, i.e., $\rho_i^{(m)} = \sum_{j \neq i} \alpha_{ij}^{(m)}$. Because an SU is not allowed to access channels occupied by PUs, a channel cannot directly transit from state 2 to state 3, i.e., $\alpha_{23}^{(m)} = 0, \forall m \in \{1, \ldots, L\}$. In contrast, when a PU becomes active on a channel occupied by an SU, the SU must leave that channel immediately, so $\alpha_{32}^{(m)} \neq 0$ in general.

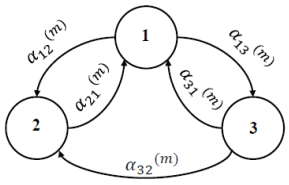


Fig. 3: State transition diagram for channel f_m.

Let $\boldsymbol{A}^{(m)} = \left[\alpha_{ij}^{(m)}\right]_{i,j}$ be the infinitesimal generator matrix for channel m, and let $\boldsymbol{P}_t^{(m)}$ be the matrix whose (i,j) entry, $p_t^{(m)}(i,j)$, is the probability that channel m goes from state i to state j in t seconds. It is known that [9]:

$$\boldsymbol{P}_t^{(m)} = e^{t\boldsymbol{A}^{(m)}}. \tag{1}$$

Without loss of generality, we assume that PUs become active on channel m with rate $\lambda_p^{(m)}$, and terminate their activity with rate $\mu_p^{(m)}$, both according to Poisson processes. Similarly, SUs arrive on channel m with rate $\lambda_s^{(m)}$ and depart with rate $\mu_s^{(m)}$, both according to Poisson processes. Let $\boldsymbol{\pi}^{(m)} = \left(\pi_1^{(m)}, \pi_2^{(m)}, \pi_3^{(m)}\right)$ be the steady-state distribution for channel m. Then, $\boldsymbol{\pi}^{(m)}$ can be written as:

$$\pi_1^{(m)} = \frac{\mu_p^{(m)}\left(\lambda_p^{(m)}+\mu_s^{(m)}\right)}{\left(\lambda_p^{(m)}+\mu_p^{(m)}\right)\left(\lambda_s^{(m)}+\lambda_p^{(m)}+\mu_s^{(m)}\right)}, \quad \pi_2^{(m)} = \frac{\lambda_p^{(m)}}{\left(\lambda_p^{(m)}+\mu_p^{(m)}\right)}$$

$$\pi_3^{(m)} = \frac{\mu_p^{(m)}\lambda_s^{(m)}}{\left(\lambda_p^{(m)}+\mu_p^{(m)}\right)\left(\lambda_s^{(m)}+\lambda_p^{(m)}+\mu_s^{(m)}\right)}.$$

III. Quorum-Based FH Algorithms for Multicast Rendezvous

In this section, we present two algorithms for constructing a set of FH sequences (≥ 2) for multicast rendezvous. The first algorithm, denoted by AMQFH, uses the uniform k-arbiter quorum system. The second algorithm, denoted by CMQFH, uses the CRT quorum system. These algorithms have two main attractive features. First, they allow a node to construct its sequence by only knowing the number of nodes in its multicast group. Hence, these algorithms can be executed in a fully distributed way. Second, these algorithms can still function in the absence of node synchronization.

A. Preliminaries

Before describing our quorum-based multicast rendezvous-ing algorithms, we give a few basic definitions that will facilitate further understanding of subsequent sections.

Definition 1. *Given a set $Z_n = \{0, 1, \ldots, n-1\}$, a quorum system Q under Z_n is a collection of non-empty subsets of Z_n, each called a quorum, such that*: $\forall G, H \in Q : G \cap H \neq \emptyset$.

Throughout the paper, Z_n is used to denote the set of nonnegative integers less than n.

Definition 2. *Given a non-negative integer i and a quorum G in a quorum system Q under Z_n, we define $rotate(G, i) = \{(x+i) \mod n, x \in G\}$ to denote a cyclic rotation of quorum G by i.*

Definition 3. *A quorum system Q under Z_n is said to satisfy the rotation k-closure property for some $k \geq 2$ if $\forall G_1, G_2, \ldots, G_k \in Q$ and $\forall i_1, i_2, \ldots, i_k \in Z_n$, $\bigcap_{j=1}^{k} rotate(G_j, i_j) \neq \emptyset$.*

The rotation k-closure property exhibited by the quorum systems used in our algorithms enables them to work in the absence of node synchronization.

B. Uniform k-Arbiter Multicast FH Algorithm (AMQFH)

Before explaining the AMQFH algorithm, we first define the k-arbiter and uniform k-arbiter quorum systems.

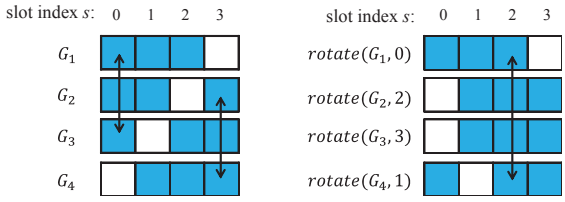


Fig. 4: Rotation 3-closure property of the uniform 2-arbiter quorum system.

Definition 4. *A k-arbiter quorum system Q under Z_n is a collection of quorums such that $\bigcap_{i=1}^{k+1} G_i \neq \emptyset, \forall G_1, G_2, \ldots, G_{k+1} \in Q$* [14].

For example, the quorum system $Q = \{\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2,3\}\}$ under Z_4 is a 2-arbiter quorum system. The intersection among any three quorums is not empty.

One specific type of k-arbiter quorum systems that is of interest to us is the so-called uniform k-arbiter quorum system. Such a system Q satisfies [8]:

$$Q = \left\{ G \subseteq Z_n \quad : \quad |G| = \left(\left\lfloor \frac{kn}{k+1} \right\rfloor + 1 \right) \right\}. \tag{2}$$

The above 2-arbiter quorum system is a uniform 2-arbiter because each quorum in Q contains $\lfloor 2 \times 4/(2+1) \rfloor + 1 = 3$ elements of Z_4. It is known [8] that the uniform k-arbiter quorum system has the rotation $(k+1)$-closure property. Figure 4 illustrates the rotation 3-closure property of the quorum system $Q = \{\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2,3\}\}$.

To create FH sequences that satisfy the rotation $(k+1)$-closure property using a uniform k-arbiter quorum system, n needs to be selected such that the number of different quorums of length $\lfloor kn/(k+1) \rfloor + 1$ that can be derived from Z_n is greater than or equal to $k+1$, i.e.,

$$\begin{pmatrix} n \\ \left\lfloor \frac{kn}{k+1} \right\rfloor + 1 \end{pmatrix} \geq k+1. \tag{3}$$

To satisfy (3), $\left\lfloor \frac{kn}{k+1} \right\rfloor$ should be less than $n-1$, which requires n to be greater than $k+1$ (k and n are positive integers, and $\frac{k}{k+1}$ is monotonically increasing in k).

We now explain AMQFH through an example. Consider a multicast group of 3 nodes. In AMQFH, each FH sequence consists of several time frames, each containing several slots. Because the uniform 2-arbiter quorum system satisfies the rotation 3-closure property (i.e., any three cyclically rotated quorums overlap in at least one slot), each frame is constructed using one quorum. Thus, the frame length will be n. We set n to the smallest value that satisfies (3), i.e., $n = k+2 = 4$. The following steps are used to obtain the various FH sequences:

1) Construct a universal set $Z_4 = \{0,1,2,3\}$.
2) Construct a uniform 2-arbiter system Q under Z_4.
3) Construct an FH sequence $\boldsymbol{w}$ as follows:
   - Select a quorum from Q and assign it to $G^{(1)}$ (e.g., $G^{(1)} = \{0,1,2\}$). The quorum selection procedure will be explained in Section VII.
   - Assign a frequency h_1 to the FH slots in the given frame that correspond to $G^{(1)}$, and assign a random frequency h_x to the other slots, where h_1 and $h_x \in \{f_1, f_2, \ldots, f_L\}$. The channel selection procedure will be discussed in Section VI.
   - Repeat the above procedure for the other frames using quorum $G^{(k)}$ and channel f_k for the kth frame.
4) Repeat step 3 to construct the other FH sequences.

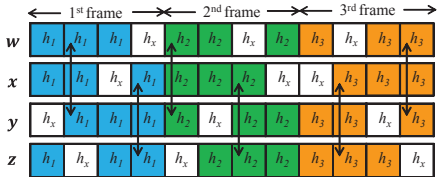


Fig. 5: AMQFH FH construction algorithm.

A pseudo-code of the AMQFH algorithm for constructing one frame of FH sequences, where any $k+1$ sequences overlap at least in one slot, is shown in Algorithm 1. Figure 5 shows three frames of FH sequences $\boldsymbol{w}$, $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, constructed according to the AMQFH algorithm.

Algorithm 1 AMQFH Algorithm

Input: multicast group size $(k+1)$, $\boldsymbol{f} = \{f_1, f_2, \ldots, f_L\}$, h
Output: $\boldsymbol{R}$
$\boldsymbol{R} = \emptyset$
Compute: $n = k+2$, Z_n, $|\boldsymbol{R}| = \begin{pmatrix} n \\ \left\lfloor \frac{kn}{k+1} \right\rfloor + 1 \end{pmatrix}$, and a uniform k-arbiter quorum system $Q = \{G_1, G_2, \ldots, G_{|R|}\}$ under Z_n
for $j = 1 : |\boldsymbol{R}|$ **do**
  Select a quorum H_j from Q
  for $i = 0 : n-1$ **do**
    if $i \in H_j$ **then**
      $w_i^{(j)} = h$
    else
      $w_i^{(j)} = h_x$, randomly chosen from $\boldsymbol{f}$
    end if
  end for
  $\boldsymbol{R} = \boldsymbol{R} \cup \boldsymbol{w}^{(j)}$
end for

C. CRT Multicast FH Algorithm (CMQFH)

The CMQFH algorithm uses the CRT quorum system, which also exhibits the rotation k-closure property. The CRT is formally described as follows [16].

Theorem 1. *Let $x_1, \ldots, x_k$ be k positive integers that are pairwise relatively prime, i.e., $gcd(x_i, x_j) = 1, \forall i, j \in \{1, \ldots, k\}$, where $gcd(x_i, x_j)$ is the greatest common divisor of x_i and x_j. Let $y = \prod_{l=1}^{k} x_l$ and let $z_1, \ldots, z_k$ be k integers, where $z_i < x_i, \forall i \in \{1, \ldots, k\}$. Then, there exists a solution I for the following system of simultaneous congruences:*

$$z_1 \pmod{x_1} \equiv z_2 \pmod{x_2} \equiv \ldots \equiv z_k \pmod{x_k}.$$

Furthermore, any two solutions I and I' to the above system are congruent modulo y, i.e., $I' \equiv I \pmod{y}$. That is, there exists exactly one solution I between 0 and $y-1$.

Using Theorem 1, we can construct quorum systems that satisfy the rotation k-closure property, as in Theorem 2.

Theorem 2. *Let $x_1, \ldots, x_k$ be k positive integers that are pairwise relatively prime, and let $y = \prod_{l=1}^{k} x_l$. The CRT*

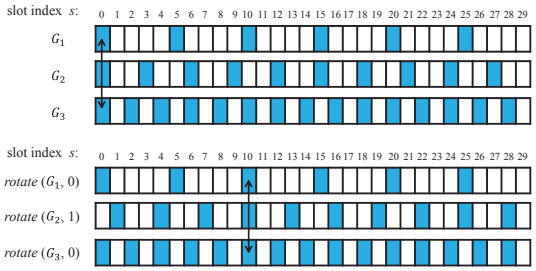


Fig. 6: Rotation 3-closure property of a CRT quorum system.

quorum system $Q = \{G_1, \ldots, G_k\}$, *where* $G_i = \{x_i c_i, c_i = 0, \ldots, y/x_i - 1\}$, *satisfies the rotation* k*-closure property.*

As an example of the CRT quorum system, consider three pairwise relatively prime numbers $x_1 = 2, x_2 = 3$, and $x_3 = 5$. Then, $y = x_1 x_2 x_3 = 30$. We can construct three quorums $G_1 = \{0, 2, 4, \ldots, 28\}$, $G_2 = \{0, 3, 6, \ldots, 27\}$, and $G_3 = \{0, 5, 10, \ldots, 25\}$ according to x_1, x_2, and x_3, respectively, under Z_{29}. When $z_1 = 0, z_2 = 1$, and $z_3 = 0, \bigcap_{j=1}^{3} rotate(G_j, z_j) = 10$, as shown in Figure 6. It is not difficult to verify that $\forall z_1, z_2, z_3 \in Z_{29}$, quorums G_1, G_2, and G_3 have an intersection. Thus, the CRT quorum system $Q = \{G_1, G_2, G_3\}$ satisfies the rotation 3-closure property.

The CMQFH algorithm for generating k asynchronous multicast FH sequences is similar to the AMQFH algorithm, with two main differences. First, The frame length is equal to $y = \prod_{i=1}^{k} x_i$. Second, CMQFH uses the CRT quorum system instead of the uniform $(k-1)$-arbiter quorum system.

D. Asynchronous Multicast Rendezvous

Result 1. *FH sequences constructed according to AMQFH and CMQFH algorithms can support asynchronous multicast rendezvous if each FH sequence continues to use the same frequency and the same quorum in all frames of the FH sequence (e.g., in Figure 5,* $h_1 = h_2 = h_3 = h^*$ *and* $G^{(1)} = G^{(2)} = G^{(3)} = G^*$*).*

Proof. Result 1 is a direct consequence of the $(k+1)$-intersection and the rotation $(k+1)$-closure properties of the uniform k-arbiter and CRT quorum systems, and the fact that each frame in an FH sequence is constructed using one quorum from the uniform k-arbiter or CRT quorum systems.

Throughout the paper, we will use G^* and h^* to denote the common quorum and the common rendezvous channel that applies to all frames of an FH sequence. The condition in Result 3 is sufficient but not necessary. Thus, FH sequences can still rendezvous even if the quorum changes from one frame to the next, provided that this change does not occur very frequently. This is illustrated in Figure 7, where sequence $\boldsymbol{w}$ uses two quorums H_1 and H_2, and sequence $\boldsymbol{x}$ uses quorums H_3 and H_4, as depicted in the figure. The left shaded part of sequence $\boldsymbol{x}$ in Figure 7 represents a cyclic rotation of H_3, and hence, by the rotation k-closure property of the uniform k-arbiter and CRT quorum systems, this part overlaps with quorum H_1 of sequence $\boldsymbol{w}$. The right shaded part in sequence $\boldsymbol{x}$ does not generally overlap with H_1 in $\boldsymbol{w}$.

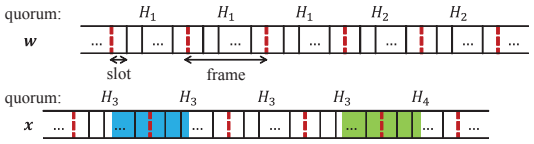


Fig. 7: Example of asynchronous rendezvous.

Because the condition in Result 3 is sufficient but not necessary, we require the rendezvous channel of a given FH sequence to be available for a certain number of slots in the current quorum in order to keep assigning this channel to this same quorum in the next frame. Otherwise, this channel is assigned to the quorum for which it is maximally available (i.e., the quorum that has the maximum number of available slots during which this channel is predicted to be idle).

E. Heterogeneous Multicast Rendezvous

In Figure 5, the four FH sequences are constructed using the same rendezvous channels h_1, h_2, and h_3, but with different $G^{(1)}, G^{(2)}$, and $G^{(3)}$ quorums. To allow nodes to construct their FH sequences in a fully distributed way, depending on their different views of spectrum opportunities, we also consider a variant of our multicast algorithms whereby each node assigns channels to quorum slots based on the forecasted availability of these channels. Note that even in a heterogeneous spectrum environment, neighboring nodes are still likely to partially overlap in their views of idle channels. Hence, when neighboring nodes construct their FH sequences independently based on the forecasted channel availability, they will likely end up having similar channel assignments. The algorithm for forecasting the channel state and accordingly assigning rendezvous channels will be discussed in Section VII.

IV. AMQFH vs. CMQFH

In this section, we compare the AMQFH and CMQFH algorithms. Before comparing these algorithms, we explain two different implementations of these algorithms:

- **Centralized:** In this case, the source node constructs the multicast FH sequences based on AMQFH or CMQFH. It then performs *pairwise rendezvous* with the nodes in the target multicast group. During each pairwise rendezvous, the source node communicates one multicast FH sequence. The receiving node tunes to the new sequence. Note that a receiving node does not wait until all nodes in the multicast group tune to their new FH sequences, as these sequences are constructed using AMQFH or CMQFH, which have the rotation k-closure property.
- **Distributed:** The source node uses a series of pairwise rendezvous to communicate only the number of nodes in the multicast group to the target multicast group. Then, each receiving node constructs its own multicast FH sequence. In this case, it is possible for two nodes to end up having the same multicast FH sequence. Note that for AMQFH and CMQFH, knowing the number of nodes in the multicast group is enough to construct the multicast FH sequences.

If resilience to node compromise is important, the centralized approach is more preferable to the distributed approach, because different nodes in the multicast group follow different FH sequences. Otherwise, the distributed approach is preferable, since it does not require the multicast initiator to send the FH sequences to the other nodes in the multicast group.

A. Expected TTR

Let $\mathcal{T}_{AH}$ and $\mathcal{T}_{CH}$ denote the expected TTR for AMQFH and CMQFH, respectively, where the expectation is taken over all possible random assignments. Considering the distributed implementation, $\mathcal{T}_{AH}$, can be expressed as:

$$\mathcal{T}_{AH} = \sum_{i=1}^{n-1} \left[i\beta(\alpha_{i+1}) \prod_{j=1}^{i} (1 - \beta(\alpha_j)) \right] \tag{4}$$

where $\beta(\alpha_i)$ and $\alpha_i, i = 1, \ldots, n-1$, are given by:

$$\beta(\alpha_j) = \sum_{i=0}^{k} \left[\binom{k+1}{i} \alpha_j^{k+1-i} \left(\frac{1-\alpha_j}{L} \right)^i \right] + (1-\alpha_j)^{k+1} \left(\frac{1}{L} \right)^k \tag{5}$$

$$\alpha_i = \frac{\left\lfloor \frac{kn}{k+1} \right\rfloor - i + 2}{n} + \frac{i-1}{n} \times \frac{\left\lfloor \frac{kn}{k+1} \right\rfloor - i + 3}{n-i+1}. \tag{6}$$

As for $\mathcal{T}_{CH}$, it is given by:

$$\mathcal{T}_{CH} = \sum_{i=1}^{n-1} i(1-\varphi)^{i-1}\varphi \tag{7}$$

where φ is given by:

$$\varphi = \sum_{i=0}^{k-1} \sum_{\substack{\forall \{s_1, s_2, \ldots, s_{k-i}\} \\ \in \{x_1, x_2, \ldots, x_k\}}} \left[\frac{1}{s_1 s_2 \ldots s_{k-i}} \left(\frac{1}{L} \right)^i \prod_{j=k-i+1}^{k} \left(1 - \frac{1}{s_j} \right) \right] + \left(\frac{1}{L} \right)^{k-1} \prod_{l=0}^{k-1} \left(1 - \frac{1}{s_l} \right). \tag{8}$$

k equals to the number of nodes minus one for $\mathcal{T}_{AH}$, and to the number of nodes for $\mathcal{T}_{CH}$. Figures 9 and 10 show the expected TTR for AMQFH and CMQFH, respectively. The upper bound on the TTR corresponds to the case when nodes cannot rendezvous during the randomly assigned slots with probability 1. For both AMQFH and CMQFH, the expected TTR increases with L and with the multicast group size.

B. Expected Hamming Distance (HD)

The expected HD for two FH sequences $\boldsymbol{x} = (x_1 x_2 \ldots x_n)$ and $\boldsymbol{y} = (y_1 y_2 \ldots y_n)$, denoted by $\mathcal{D}^{(\boldsymbol{x},\boldsymbol{y})}$, is defined as $\mathcal{D}^{(\boldsymbol{x},\boldsymbol{y})} \stackrel{\text{def}}{=} E[D^{(\boldsymbol{x},\boldsymbol{y})}] = E[\left(\sum_{i=1}^{n} \mathbf{1}_{\{x_i \neq y_i\}}\right)/n]$, where $\mathbf{1}_{\{\cdot\}}$ is the indicator function. In addition to robustness against node compromises, FH sequences with higher HD will have a lower collision probability. A collision occurs when two or more neighboring multicast groups rendezvous on the same slot using the same channel. We now derive $\mathcal{D}^{(\boldsymbol{x},\boldsymbol{y})}$ under AMQFH and CMQFH. For simplicity, we drop the superscript in $\mathcal{D}^{(\boldsymbol{x},\boldsymbol{y})}$ and replace it with $\mathcal{D}$. In AMQFH, $\mathcal{D}$ is the same for all pairs of FH sequences, whereas in CMQFH they are different for different pairs. Thus, for CMQFH, we will compute the expected value over all pairs of FH sequences.

Let $\mathcal{D}_{AH}$ and $\mathcal{D}_{CH}$ denote the value of $\mathcal{D}$ for the AMQFH and CMQFH algorithms, respectively. Let $\mathcal{D}_{AH,C}$ and $\mathcal{D}_{AH,D}$ denote the value of $\mathcal{D}$ for the centralized and distributed versions of AMQFH, respectively. $\mathcal{D}_{CH,C}$ and $\mathcal{D}_{CH,D}$ are defined similarly for CMQFH. Let $\mathcal{D}^*_{AH}$ and $\mathcal{D}^*_{CH}$ represent upper bounds on $\mathcal{D}_{AH}$ and $\mathcal{D}_{CH}$, respectively. Then, after some manipulations it can be shown that:

$$\mathcal{D}_{AH} = \mu_1 \frac{\left(n - \left\lfloor \frac{kn}{k+1} \right\rfloor\right)\mu_3}{n} + (1-\mu_1)\frac{\left(n - \left\lfloor \frac{kn}{k+1} \right\rfloor - 1\right)\mu_3}{n} \tag{9}$$

where $\mu_1 = \frac{\mu_2 - 1}{\mu_2}$, $\mu_2 = \binom{n}{\left\lfloor \frac{kn}{k+1} \right\rfloor + 1}$ for $\mathcal{D}_{AH,D}$, and $\mu_1 = 1$ for $\mathcal{D}_{AH,C}$ and $\mathcal{D}^*_{AH}$. μ_3 equals $\frac{L-1}{L}$ for $\mathcal{D}_{AH,C}$ and $\mathcal{D}_{AH,D}$, and equals 1 for $\mathcal{D}^*_{AH}$. Furthermore,

$$\mathcal{D}_{CH} = \frac{\sum_{i=1}^{k} \sum_{\substack{j=1 \\ \{\alpha_1\}}}^{k} \left[\left(n - \frac{n}{x_i x_j} \right) \alpha_2 \right]}{2\alpha_3 n} \tag{10}$$

where α_1 equals $j \neq i$ for $\mathcal{D}_{CH,C}$ and $\mathcal{D}^*_{CH}$ and equals $\emptyset$ for $\mathcal{D}_{CH,D}$, α_2 equals $\frac{L-1}{L}$ for $\mathcal{D}_{CH,C}$ and $\mathcal{D}_{CH,D}$ and equals 1 for $\mathcal{D}^*_{CH}$, and α_3 equals $\binom{k}{2}$ for $\mathcal{D}_{CH,C}$ and $\mathcal{D}^*_{CH}$ and equals k^2 for $\mathcal{D}_{CH,D}$. The frame length n equals to $\prod_{i=1}^{k} x_i$.

Figure 8 depicts $\mathcal{D}$ vs. the multicast group size for the AMQFH and CMQFH algorithms. As the multicast group size increases, $\mathcal{D}_{CH}$ increases whereas $\mathcal{D}_{AH}$ decreases. Hence, the improvement in $\mathcal{D}$ that is achieved by using CMQFH instead of AMQFH increases with the increase in the size of the multicast group. The centralized implementations of AMQFH and CMQFH have better $\mathcal{D}$ than their distributed versions.

V. Nested-CMQFH Algorithm

As shown in the previous section, the TTR of CMQFH is much larger than that of AMQFH, but its average HD is also much higher. Hence, AMQFH is preferred over CMQFH if the primary goal is to establish multicast communications as fast as possible, but it may not be very efficient under node comprise because of its relatively small HD. To provide a tradeoff between speed of rendezvous and robustness against node compromise, in this section we propose a third multicast rendezvous algorithm, called nested-CMQFH. Nested-CMQFH is faster than CMQFH, but not as fast as AMQFH. At the same time, the HD of nested-CMQFH is larger than that of AMQFH, but not as large as CMQFH. We explain the nested-CMQFH algorithm through an example.

Suppose that the number of nodes in the multicast group is 3. Then, according to the CMQFH algorithm, $x_1 = 2, x_2 = 3, x_3 = 5$, and the frame length $y = x_1 x_2 x_3 = 30$. The

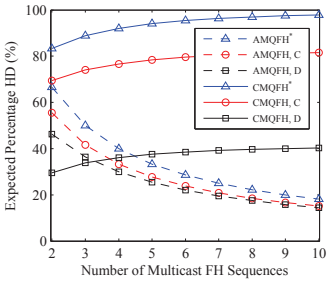


Fig. 8: $\mathcal{D}$ vs. multicast group size ($L = 6$).

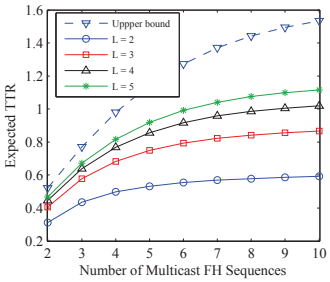


Fig. 9: Expected TTR for AMQFH.

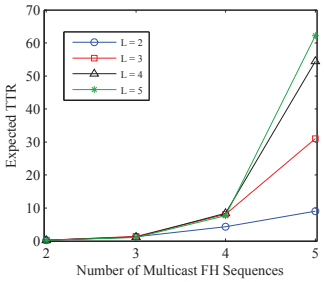


Fig. 10: Expected TTR for CMQFH.

difference between CMQFH and nested-CMQFH is that instead of having one quorum in each frame of an FH sequence, each FH sequence will have a certain number of quorums in each frame, depending on the prime number that is used in constructing this FH sequence. The number of quorums in a frame for a given FH sequence is called *the nesting degree* of this FH sequence. In contrast to AMQFH and CMQFH, knowing the number of nodes in the multicast group is not enough to construct the multicast FH sequences in nested-CMQFH. In addition to the multicast group size, a node needs to know its nesting degree. The nesting degree constitutes a tradeoff between TTR and HD. Large values of the nesting degree result in a small TTR, but also a small HD. In our design, the FH sequence that uses a prime number x_i will have a nesting degree of $\lceil \frac{x_i}{2} \rceil$. The quorums used in constructing this FH sequence will be denoted throughout the paper by $G_j^*, j \in \{1, \ldots, \lceil \frac{x_i}{2} \rceil\}$. The selection of the prime number x_i and the corresponding $\lceil \frac{x_i}{2} \rceil$ quorums, $G_j^*, j \in \{1, \ldots, \lceil \frac{x_i}{2} \rceil\}$, will be explained in Section VII. Figure 11 illustrates the nested-CMQFH design when the multicast group size is 3. The performance of nested-CMQFH will be examined and compared with AMQFH and CMQFH in Section VIII.

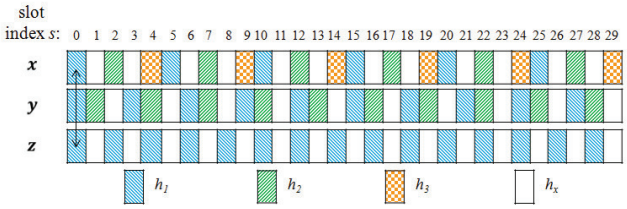


Fig. 11: Nested-CMQFH (group size = 3).

VI. Optimal Channel Ordering

In the previous sections, we introduced three multicast rendezvous algorithms without explaining the channel assignment process, i.e., the channel h^* to assign to quorum G^* for AMQFH and CMQFH, and the channels $h_i, i \in \{1, \ldots, \lceil \frac{x_i}{2} \rceil\}$, to assign to quorums $W_i, i \in \{1, \ldots, \lceil \frac{x_i}{2} \rceil\}$, respectively, for nested-CMQFH. To achieve efficient channel assignment, each node *independently* sorts available channels in some optimal sense (no message exchange is assumed between the nodes). In addition to channel assignment, the sorted list of channels is also used to *sequentially sense channels*, such that the best channel will be sensed first, followed by the next best channel, and so on. The best channel is selected according to several factors, as will be explained in this section.

Furthermore, in the previous sections, we did not specify the quorum selection procedure. One naïve approach to jointly address the channel sorting and quorum selection problems is to exhaustively examine all possible channel-quorum assignments and select the one that maximizes the number of available slots (i.e., slots during which the assigned channels are available). The time complexity of this exhaustive search is given by:

$$\begin{array}{ll} \mathcal{O}\left(\binom{n}{\lfloor \frac{kn}{k+1} \rfloor + 1}\left(\lfloor \frac{kn}{k+1} \rfloor + 1\right)L'\right), & \text{AMQFH} \\ \mathcal{O}(kyL'), & \text{CMQFH} \\ \mathcal{O}\left(\sum_{i=1}^{k}\binom{L'}{\lceil \frac{x_i}{2} \rceil}\frac{x_i! y \lceil \frac{x_i}{2} \rceil}{(x_i - \lceil \frac{x_i}{2} \rceil)! x_i}\right), & \text{nested-CMQFH} \end{array}$$

where k is the size of the multicast group minus one for AMQFH, and the size of the multicast group for CMQFH and nested-CMQFH, n is the frame length for AMQFH, x_i is the prime number used in constructing the ith FH sequence (as explained in Theorem 2), $y = \prod_{i=1}^{k} x_i$ is the frame length for CMQFH and nested-CMQFH, which represents the kth primorial and is given by $e^{(1+o(1))k \log k}$, and L' is the number of available channels. This expensive exhaustive search needs to be performed by each node in each frame.

To avoid performing an expensive exhaustive search for each frame and also delaying making the decision until all channels are sensed, we address the problems of quorum selection and channel assignment separately, and propose a one-time sorting algorithm that prioritizes channels. In this section, we present two channel ordering algorithms, one for AMQFH and one for CMQFH and nested-CMQFH. In Section VII, we address the quorum selection problem.

In our approach, channels are sorted based on their probabilistic availability, which is determined by the channel average availability time and its fluctuation level. The fluctuation level of a channel affects its prediction accuracy. Channel activity prediction is more conservative when a channel exhibits higher fluctuations for a given mean channel availability time. Conservative prediction of channel availability results in less exploitation of actually available slots. In addition

to probabilistic channel availability, our sorting mechanism also considers the probabilities of collision with PUs/SUs over these channels, which we require to be below a specific threshold. Moreover, for AMQFH, the sorting mechanism considers as a third metric the time that a channel spends in state 1 (idle state), which is known as the mean sojourn time of state 1. To sort channels based on these criteria, we propose an optimization problem for AMQFH that is different from the optimization problem proposed for CMQFH and nested-CMQFH. These ordering mechanisms start over when the estimate of at least one channel parameters changes.

A. Optimal Sorting for AMQFH

As mentioned earlier, channel sorting is followed by quorum selection. Consider a given channel. The best quorum for that channel is the one with the maximum number of probabilistically available slots. The AMQFH algorithm is based on the uniform k-arbiter quorum system. As shown in Figure 4, in a uniform k-arbiter quorum system, quorums consist of several consecutive slots. Hence, to be inline with the quorum selection procedure that will be performed after completing channels sorting, the sorting mechanism needs to consider the mean sojourn time of state 1, in addition to the average channel availability times, fluctuation levels, and probabilities of collision with PUs/SUs. The goal of our optimization problem is to sort channels based on their average availability times, while considering (i) the channel's fluctuation levels, (ii) the probabilities of collision with PUs/SUs for each channel, and (iii) the channel's mean sojourn time in state 1.

We formulate the channel sorting problem as a linear programming problem. The objective function is a convex combination of the average channel availability times, which needs to be maximized. The channel's fluctuation levels and collision probabilities are jointly captured by imposing constraints on them, as explained later. The channel's mean sojourn times of state 1 are incorporated through a secondary objective function, which also needs to be maximized. Hence, our problem is formulated as a multi-objective linear programming problem. Let q_m be a weight associated with channel $f_m, m \in \{1, \ldots, L\}$. The weights will be used for two different purposes. First, in the quorum-based assigned slots, the weights will be used to sort channels such that the channel with the largest weight will be considered as the best channel. Second, in the randomly assigned (non-quorum) slots, these weights will be interpreted as probabilities, such that channel f_m will be assigned to non-quorum slots with probability q_m. We obtain the optimal value of the vector $\boldsymbol{q} = (q_1, q_2, \ldots, q_L)$ that maximizes a convex combination of the average channel availabilities, as a primary objective function, and maximizes a convex combination of the channels average sojourn times of state 1 as a secondary objective, subject to the PUs/SUs collisions constraints, which also capture the channels fluctuation levels.

For $i \in \{1, 2, 3\}$ and $m \in \{1, \ldots, L\}$, let $T_i^{(m)}$ and $R_i^{(m)}$ be the sojourn time for channel m in state i and the first time that channel m returns to state i after leaving it, respectively. Let $\mathcal{T}_i^{(m)} \stackrel{\text{def}}{=} \mathbb{E}[T_i^{(m)}]$ and $\mathcal{R}_i^{(m)} \stackrel{\text{def}}{=} \mathbb{E}[R_i^{(m)}]$. Following standard Markov analysis, the fraction of time that channel m spends in state i (i.e., $\mathcal{T}_i^{(m)}/(\mathcal{T}_i^{(m)} + \mathcal{R}_i^{(m)})$) is $\pi_i^{(m)}$, which was given in Section II-B. $\mathcal{T}_1^{(m)}$ is given by $1/(\lambda_p^{(m)} + \lambda_s^{(m)})$.

Our multi-objective channel sorting problem is formulated as a two-stage sequential optimization problem. Problem 1 is the first stage and Problem 2 is the second stage.

Problem 1.

$$\underset{\boldsymbol{q}=(q_1,q_2,\ldots,q_L)}{\text{maximize}} \left\{ \mathcal{U}(\boldsymbol{q}) \stackrel{\text{def}}{=} \sum_{m=1}^{L} \pi_1^{(m)} q_m \right\}$$

s.t.

$$\left[1 - \prod_{\substack{u=0 \\ u \neq i}}^{n-1} (1 - p_{(n+u)T}^{(m)}(1, s))\right] q_m < \lambda_{PU,Col}^{(m)}(n), \quad \forall s \in \{2, 3\}, \forall m \in \{1, \ldots, L\}, \forall i \in \{1, \ldots, \mathcal{C}\} \tag{11}$$

$$\sum_{m=1}^{L} q_m = 1 \tag{12}$$

$$0 \leq q_m \leq 1, \forall m \in \{1, \ldots, L\} \tag{13}$$

where $\mathcal{C} \stackrel{\text{def}}{=} \begin{pmatrix} n \\ \left\lfloor \frac{kn}{k+1} \right\rfloor + 1 \end{pmatrix}$, and $\lambda_{PU,Col}^{(m)}(n)$ and $\lambda_{SU,Col}^{(m)}(n)$ are prespecified thresholds on the probabilities of collisions with PUs and other SUs, respectively. Note that both sets of thresholds are functions of the frame length n and channel m. The objective function in Problem 1 represents a convex combination of the average channel availabilities $\pi_1^{(m)}, m = 1, \ldots, L$. Constraint (11) restricts the collision probabilities with PUs and SUs, while considering the specific structure of uniform k-arbiter quorum systems. In addition to constraining the collision probabilities, constraint (11) is used to differentiate between channels based on their levels of fluctuations by imposing different upperbounds on q_m. Note that channels with higher fluctuation levels will have higher collision probabilities, and hence will be less preferable (i.e., assigned a smaller weight). Let $\boldsymbol{q}_I^*$ be an optimal solution to Problem 1, and let $\mathcal{U}_I^* = \mathcal{U}(\boldsymbol{q}_I^*)$. The goal of the second optimization stage is to give slightly higher priority to channels with larger mean sojourn times of state 1 (i.e., channels with more consecutive available slots).

Problem 2.

$$\underset{\boldsymbol{q}=(q_1,q_2,\ldots,q_L)}{\text{maximize}} \left\{ \mathcal{F}(\boldsymbol{q}) \stackrel{\text{def}}{=} \sum_{m=1}^{L} \mathcal{T}_1^{(m)} q_m \right\}$$

s.t.

$$\mathcal{U}_I^*(1 - \epsilon) < \mathcal{U}(\boldsymbol{q}). \tag{14}$$

Problem 2 aims at maximizing a convex combination of the average sojourn times of state 1 subject to constraints (11)– (13), in addition to the new constraint in (14). Let $\mathcal{F}^*$ be the optimal value of $\mathcal{F}(\boldsymbol{q})$ in Problem 2, and let $\mathcal{U}^*$ be the corresponding value of $\mathcal{U}(\boldsymbol{q})$. In (14), ϵ, $0 \leq \epsilon \leq 1$, restricts the reduction in the first objective function optimal

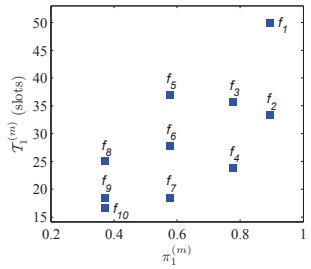


Fig. 12: Example of 10 channels with different $\pi_1^{(m)}$ and $\mathcal{T}_1^{(m)}$.

value (i.e., $\mathcal{U}_I^* - \mathcal{U}^*$). Increasing ϵ increases the effect of the second objective function on channel ordering.

Having explained the structure of the two-stage sequential optimization problem, we now study the behavior of this problem under different parameters using an example. Consider a network with ten channels. The values of $\pi_1^{(i)}$ and $\mathcal{T}_1^{(i)}$ for channel $f_i, i = 1, 2, \ldots, 10$, are shown in Figure 12. These values are selected such that they cover the cases where different channels have the same average availability times, but different mean sojourn times of state 1, fluctuation levels, and collision probabilities. To complete the description, Table I depicts the values of π_2/π_3 for different channels. Table II shows the order of channels for different values of $\lambda = \lambda_{PU,Col}^{(m)}(n) = \lambda_{SU,Col}^{(m)}(n)$ and different values of ϵ ($k = 4$). Only channels with non-zero weights are shown.

TABLE I: $\pi_2^{(m)}/\pi_3^{(m)}$ for different channels.

Ch. (m)	$\pi_2^{(m)}/\pi_3^{(m)}$	Ch. (m)	$\pi_2^{(m)}/\pi_3^{(m)}$
f_1	11.00427	f_6	188.3636
f_2	11.00427	f_7	188.3636
f_3	4.68	f_8	1.066667
f_4	4.68	f_9	1.066667
f_5	188.3636	f_{10}	1.066667

For our example, the minimum value of λ that makes the problem feasible, denoted by $\lambda_{\min}$, is 0.05. This value is small enough so that none of the channels can receive a very high weight, leaving the other channels with zero weights. As λ increases, channels with larger values of $\pi_1^{(m)}$ receive higher weights, leaving the less available channels with zero weights. Note that in this example, for some values of λ and ϵ, even though channels f_5, f_6, and f_7 have larger $\pi_1^{(m)}$ values than f_8, they receive smaller weights than f_8. The reason is that

TABLE II: Channel order for AMQFH (best is leftmost).

ϵ										
				λ	=	0.05				
0	f_1	f_2	f_3	f_4	f_8	f_5	f_9	f_6	f_7	f_{10}
0.05	f_1	f_2	f_3	f_4	f_8	f_5	f_9	f_6	f_7	f_{10}
0.1	f_1	f_2	f_3	f_4	f_8	f_5	f_9	f_6	f_7	f_{10}
ϵ				λ	=	0.07				
0	f_1	f_2	f_3	f_4	f_5	f_6				
0.05	f_1	f_2	f_3	f_5	f_6	f_8	f_4			
0.1	f_1	f_2	f_3	f_8	f_5	f_6	f_4			
ϵ				λ	=	0.08				
0	f_1	f_2	f_3	f_4	f_5	f_6				
0.05	f_1	f_2	f_3	f_5	f_6	f_8	f_4			
0.1	f_1	f_2	f_3	f_5	f_6	f_8				

these channels have different values of π_2/π_3, and as shown in constraint (11), Problem 1 differentiates between collisions with PUs and collisions with SUs. Channels f_5, f_6, and f_7 have relatively high collision probabilities with PUs but very low collision probabilities with other SUs, whereas channel f_8 have probabilities of collision with PUs and SUs that are neither high nor low. Because channels f_5, f_6, and f_7 have higher collision probabilities with PUs compared to channel f_8, q_5, q_6, and q_7 are smaller than q_8. Furthermore, for small values of λ, increasing ϵ does not change the channels order because the feasibility region is small. On the other hand, when λ is large enough (e.g., $\lambda = 0.08$), increasing ϵ can affect the order of channels.

B. Optimal Sorting for CMQFH and Nested-CMQFH

CMQFH and nested-CMQFH algorithms use the CRT quorum system. In this system, quorums do not necessarily have consecutive slots, and hence there is no need for the second objective function in Problem 2. Furthermore, since the structure of the CRT quorum system is different than that of the uniform k-arbiter quorum system, the constraints of the new optimal sorting mechanism are different from those in Problem 1. Let $\phi \stackrel{\text{def}}{=} \max_{1 \le j \le k} \{x_j\}$ and $\psi_i \stackrel{\text{def}}{=} \left\lfloor \frac{\phi}{x_i} \right\rfloor, i = 1, 2, \ldots, k$. Then, our channel sorting problem for CMQFH and nested-CMQFH can be formulated as follows:

Problem 3.

$$\underset{\boldsymbol{q}=(q_1,q_2,\ldots,q_L)}{\text{maximize}} \left\{ \mathcal{U}(\boldsymbol{q}) \stackrel{\text{def}}{=} \sum_{m=1}^{L} \pi_1^{(m)} q_m \right\}$$

s.t.

$$\frac{1}{\psi_i} \sum_{v=1}^{\psi_i} \left[1 - \prod_{u=\frac{(v-1)y}{\phi}}^{v\frac{y}{\phi}-1} (1 - p_{(y+ux_i)T}^{(m)}(1,s)) \right] q_m < \lambda_{PU,Col}^{(m)}(y),$$
$$\forall s \in \{2,3\}, \forall m \in \{1,\ldots,L\}, \forall i \in \{1,\ldots,k\} \quad (15)$$

$$\sum_{m=1}^{L} q_m = 1 \quad (16)$$

$$0 \le q_m \le 1, \forall m \in \{1,\ldots,L\} \quad (17)$$

where k is the number of nodes and $y = \prod_{i=1}^{k} x_i$.

In contrast to uniform k-arbiter quorum systems, different quorums in CRT quorum systems have different sizes. Because of this, following the same approach in constraining the collision probabilities as in (11) will lead to two undesirable effects. First, quorums that use small prime numbers (i.e., have a large number of slots) will have high collision probabilities (i.e., the probability that one of the slots of a quorum is in collision), and hence cannot satisfy the collision probabilities constraints. Second, quorums with smaller numbers of slots will end up with lower collision probabilities, and hence receive larger weights, even though they may have fewer available slots. To remedy these undesirable outcomes, we divide each frame into sub-frames, each with length equals

TABLE III: Channel order for CMQFH and nested-CMQFH.

λ	best									worst
0.067	f_1	f_2	f_3	f_4	f_8	f_9	f_{10}	f_5	f_6	f_7
0.068	f_1	f_2	f_3	f_4	f_8	f_9	f_{10}	f_5	f_6	f_7
0.069	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
0.07	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}

to y/ϕ, and consider the collision probability for each sub-frame. Then, we compute the average over the sub-frames of a given frame, as shown in (15). Table III depicts the order of channels in Figure 12, obtained by solving Problem 3 for different values of $\lambda = \lambda_{PU,Col}^{(m)}(n) = \lambda_{SU,Col}^{(m)}(n)$ and for $k = 5$. Setting λ to the minimum value that makes the problem feasible (i.e., $\lambda = \lambda_{\min} = 0.067$), we observe that the problem favors channels f_8, f_9, and f_{10} over channels f_5, f_6, and f_7 because the former three channels have higher collision probabilities with PUs. Similar to AMQFH, increasing λ relaxes the constraints in (15), which makes $\pi_1^{(m)}$ the dominant factor in ordering channels.

VII. DFH Algorithm

We now explain how quorums are selected in the AMQFH, CMQFH, and nested-CMQFH algorithms. As mentioned before, our quorum selection procedure relies on forecasting the states of various channels in the next frame, driven by proactive out-of-band sensing of their states in the current frame. Because this procedure results in online adaptation of the quorum G^* (for AMQFH and CMQFH) and the quorums $G_j^*, j \in \left\{1, \ldots, \left\lceil \frac{x_i}{2} \right\rceil\right\}, i \in \{1, 2, \ldots, k\}$ (for nested-CMQFH), and hence online adaptation of the FH sequences, it is effectively a DFH algorithm. We use the following example to explain this DFH algorithm. In this example, we use the nested-CMQFH algorithm. CMQFH is a special case of nested-CMQFH, where all nodes have a nesting degree of one. For AMQFH, the same approach applies, but using a different quorum structure.

Consider an FH sequence that uses the prime number 5. Then, there will be $\left\lceil \frac{5}{2} \right\rceil = 3$ nested quorums in each frame. Consider a given frame in this sequence. The node that follows this FH sequence starts sensing channels according to the order obtained in Section VI. Let $\boldsymbol{h} = \{h_1, h_2, h_3\}$ be the best three available channels, ordered decreasingly according to their quality. Then, quorums G_1^*, G_2^*, and G_3^* that will be assigned to channels h_1, h_2, and h_3, respectively, will be selected so as to maximize the number of quorum slots for which the assigned channel is idle with probability greater than a threshold γ. If more than one quorum assignment results in the same maximum number of slots, we break the tie based on the average idle probabilities of h_1, h_2, and h_3, averaged over all slots that belong to G_1^*, G_2^*, and G_3^*, respectively. Formally, the problem of selecting quorums $G_1^*, \ldots, G_{\left\lceil \frac{x_i}{2} \right\rceil}^*$ for the FH sequence that uses prime number x_i is formulated as follows:

$$\begin{aligned} \underset{\left(G_1^*, G_1^*, \ldots, G_{\left\lceil \frac{x_i}{2} \right\rceil}^*\right)}{\text{maximize}} & \left\{ \sum_{j=1}^{\left\lceil \frac{x_i}{2} \right\rceil} \sum_{l=0}^{n-1} \mathbf{1}_{\{p_{(n-j\frac{\tau_s}{T}+l)T}^{(j')}(1,1) \geq \gamma\}} \right. \\ & \left. + \frac{1}{\left\lceil \frac{x_i}{2} \right\rceil \left(\frac{y}{x_i}\right)} \sum_{j=1}^{\left\lceil \frac{x_i}{2} \right\rceil} \sum_{l=0}^{n-1} p_{(n-j\frac{\tau_s}{T}+l)T}^{(j')}(1,1) \right\} \end{aligned} \tag{18}$$

where τ_s is the sensing time for one channel and y is the frame length as defined in Section III-C. Given that $h_j = f_{j'}$ where $j' \in \{1, \ldots, L\}$, $p_{(n-j\frac{\tau_s}{T}+l)T}^{(j')}(1,1)$ is the probability that h_j will remain available in the lth slot of the next frame, given that it is currently available. The computation of $p_t(x, y)$ was explained in Section II-B. The second term in (18) is always < 1. Hence, for two different sets of quorums $H_i, i \in \left\{1, 2, \ldots, \left\lceil \frac{x_i}{2} \right\rceil\right\}$ and $H_i^{'}, i \in \left\{1, 2, \ldots, \left\lceil \frac{x_i}{2} \right\rceil\right\}$, if the set $\{H_i\}$ has more probabilistically available slots than the set $\{H_i^{'}\}$, then $\{G_i^*\}$ is taken as $\{H_i\}$.

The above maximization problem is subject to three constraints: (i) $p_{(n-j\frac{\tau_s}{T}+l)T}^{(j')}(1,1) = 0, \forall l \notin H_j$ if $G_j^* = H_j$, (ii) each channel in $\left\{h_1, h_2, \ldots, h_{\left\lceil \frac{x_i}{2} \right\rceil}\right\}$ is assigned to only one of the x_i quorums, and (iii) there is no more than one channel assigned to the same quorum.

We solve the above maximization problem by considering all combinations of $\left\lceil \frac{x_i}{2} \right\rceil$ channels and x_i quorums (in our example, we have $3 \times 5 = 15$ different channels-quorums assignments) and selecting the channels-quorums assignment that results in the maximum number of available slots.

Among all prime numbers, we select the one that results in the maximum *fractional* number of available slots. The reason for considering the fractional number is that different prime numbers will result in different numbers of quorum slots, i.e., different values of $\left\lceil \frac{x_i}{2} \right\rceil \frac{y}{x_i}$.

VIII. Performance Evaluation

We now present simulation results for AMQFH, CMQFH, and nested-CMQFH. The proposed algorithms are studied under different multicast group sizes, values of γ in (18), and multicast spectrum heterogeneity levels, denoted by κ_m, $\kappa_m \in [0, 1]$. κ_m is defined as the fraction of channels whose active/idle states are perceived differently by every pair of nodes in a multicast group. Channel sorting and quorum selection procedures are integrated into AMQFH, CMQFH, and nested-CMQFH, and evaluated based on (i) the prediction accuracy, indicated by the collision rates with PUs/SUs and by missed opportunities (i.e., number of actually available slots that were considered unavailable), (ii) TTR, and (iii) average percentage HD, averaged over different pairs of FH sequences. Our algorithms are simulated under a realistic setting of no synchronization. Specifically, the misalignment between FH sequences is randomly selected in each experiment.

In our simulations, we use ten licensed channels, with the profiles given in Figure 12. Each channel is characterized by specific values of $\lambda_p^{(m)}, \lambda_s^{(m)}, \mu_p^{(m)}$, and $\mu_s^{(m)}$. To avoid having

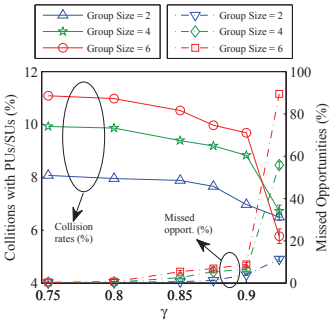
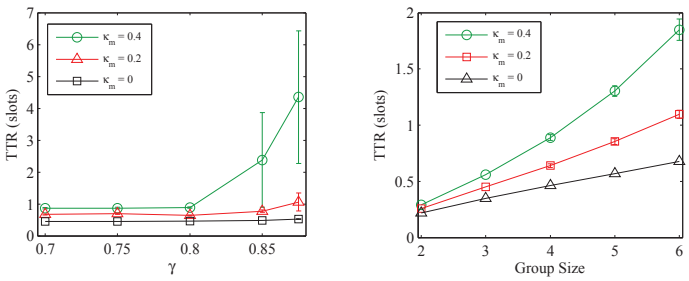
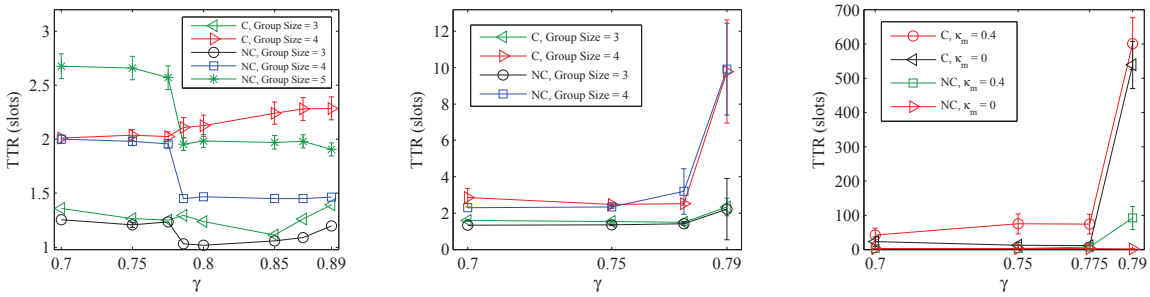


Fig. 13: Prediction accuracy for AMQFH.



(a) Group size $= 4$ (b) $\gamma = 0.8$

Fig. 14: TTR for AMQFH.



(a) $\kappa_m = 0$ (b) $\kappa_m = 0.4$ (c) Group size $= 5$

Fig. 15: TTR for CMQFH and nested-CMQFH.

the same order of channels for different runs, we slightly perturb the nominal values for the above four channel parameters within small ranges, so that the efficiency of our channel sorting and quorum selection mechanisms can be examined as well. $\lambda^{(m)}_{PU,Col}(n)$ and $\lambda^{(m)}_{SU,Col}(n)$ in problems 1 and 2 are selected to be $\lambda^{(m)}_{PU,Col}(n) = \lambda^{(m)}_{SU,Col}(n) = \lambda_{\min} + 0.02$, and in Problem 3 are selected as $\lambda^{(m)}_{PU,Col}(n) = \lambda^{(m)}_{SU,Col}(n) = \lambda_{\min} + 0.0005$, where $\lambda_{\min}$ was defined in Section VI. The reason for exceeding $\lambda_{\min}$ is to increase the feasibility region. For the AMQFH algorithm, the value of ϵ is set to 0.02. The 95% confidence intervals are indicated. When they are very tight, they are not drawn to prevent cluttering the graph.

A. TTR

1) AMQFH: The TTR for a multicast group is affected by the number of useful slots in the FH sequences that are used by the members of this group. On the one hand, as can be seen in Figure 13, relatively small values of γ bring about maximum collision rates. Decreasing γ below 0.75 does not increase the collision rate, because our ordering mechanism causes the collision rate to be upper bounded ($\sim 11\%$ if the channels are obtained from Figures 12). On the other hand, adopting a more conservative approach by increasing γ will increase the rate of missed opportunities even though the collision rate is low. Figure 14(a) shows how the selection of an unnecessarily large γ increases the TTR. Consequently, expecting the most desirable performance of TTR, we have to select the value of γ carefully. The value of γ that results in the smallest TTR depends on the channel parameters $\lambda_p^{(m)}$, $\lambda_s^{(m)}$, $\mu_p^{(m)}$, $\mu_s^{(m)}$ and κ_m (note that the TTR for $\kappa_m = 0.4$ starts increasing sooner compared to $\kappa_m = 0$). Large confidence intervals in Figure 14(a) indicate the occurrence of few cases with high TTR. TTR is considered as 0 if the nodes rendezvous in the first slot of their FH sequences.

Figures 13 and 14(a) show that a tradeoff exists between the prediction accuracy and the TTR. By selecting a near-optimal threshold ($\gamma = 0.8$ in this example), our proposed algorithm can guarantee a small TTR even in a heterogeneous spectrum environment. Increasing the size of the multicast group increases the TTR, because of the decrease in the prediction accuracy that results from increasing the lag parameter. Nonetheless, the TTR remains less than 2 slots for all cases, as shown in Figure 14(b).

2) CMQFH and Nested-CMQFH: Similar to AMQFH, increasing γ reduces the collision rate but increases missed opportunities, as shown in Figure 16. However, the prediction accuracy for nested-CMQFH is better than CMQFH, because nested-CMQFH has more deterministically assigned quorum slots than CMQFH (recall that in nested-CMQFH, high-quality channels, as obtained from Problem 3, are assigned to quorum slots, whereas non-quorum slots are randomly assigned). This improvement is particularly significant when γ is large ($0.775 < \gamma \leq 0.89$ in our setup), where nested-CMQFH improves collision rate without dramatically increases missed opportunities. Moreover, nested-CMQFH has a smaller TTR than CMQFH in both homogeneous and heterogeneous cases, as shown in Figure 15. The improvement in the TTR under nested-CMQFH is more noticeable when the group size is large (e.g., 5 in figures 15(a) and 15(c)). Even though nested-CMQFH is faster than CMQFH, it is still slower than

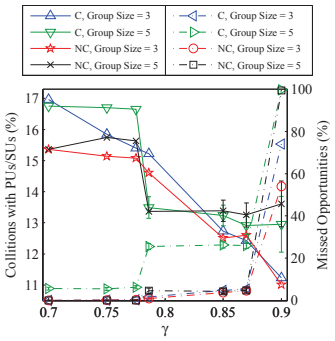
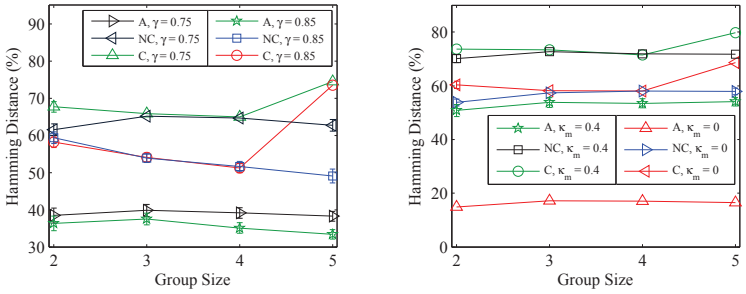


Fig. 16: Prediction accuracy for CMQFH and nested-CMQFH.



(a) $\kappa_m = 0.2$ (b) $\gamma = 0.75$

Fig. 17: HD for AMQFH, CMQFH, and nested-CMQFH.

AMQFH, as can be seen by comparing figures 14 and 15.

B. Percentage HD

In contrast to the TTR, the HD is measured without taking into account the effect of collisions, i.e., only the distance between the generated sequences is considered. Figure 17 depicts the percentage HD for AMQFH, CMQFH, and nested-CMQFH vs. the group size for different values of κ_m and γ. AMQFH has the smallest HD among the three algorithms, which corroborates the analyzed results in Section IV. Nested-CMQFH has statistically the same HD as CMQFH for moderate group sizes. For large group sizes, CMQFH has larger HD than nested-CMQFH due to higher missed opportunities. Increasing γ reduces the HD between the randomly assigned parts of the FH sequences, and hence the total HD. The reason is that when γ is increased, the number of channels that satisfy this threshold goes down, and the cardinality of the pool of available channels that can be used for the random assignment part is reduced. Figure 17(b) shows that HD is increased by increasing κ_m, as expected.

IX. Conclusions

In this paper, we developed three asynchronous algorithms (AMQFH, CMQFH, and nested-CMQFH) for multicast rendezvous in spectrum-heterogeneous DSA networks. To account for PU dynamics, we also designed two channel ordering mechanisms for sequential channel sensing and channel assignment (one for AMQFH and the other for CMQFH and nested-CMQFH). Moreover, we developed a proactive out-of-band sensing based DFH algorithm for online adaptation of the FH sequences used in our rendezvous algorithms.

Simulation results were obtained under varying and heterogeneous channel availabilities. We observed a close match between the simulation results and the numerical results that do not consider channel dynamics. This match between the numerical and simulation results shows the effectiveness of our ordering and prediction mechanisms. AMQFH can provide TTR less than 2 slots even in a heterogeneous environment if γ is selected appropriately, whereas CMQFH attains high HD ($> 50\%$). In between, nested-CMQFH achieves large HD values which are very close to CMQFH but with comparable TTR values to AMQFH, especially when the group size is moderate or the environment is homogenous.

Acknowledgment

This research was supported in part by NSF (under grants CNS-1016943 and CNS-0904681, IIP-0832238, IIP-1231043), 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] 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 ACM MobiCom Conf.*, pages 216–230, 2004.

[2] K. Bian and J. M. Park. Asynchronous channel hopping for establishing rendezvous in cognitive radio networks. In *Proc. of the IEEE INFOCOM Mini-Conf.*, 2011.

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

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

[5] FCC. Spectrum Policy Task Force Report. *ET Docket No. 02-380 and No. 04-186*, Sep. 2010.

[6] H. Garcia-Molina and D. Barbara. How to assign votes in a distributed system. *Journal of the ACM*, 32:841–860, 1985.

[7] J.-R. Jiang, Y.-C. Tseng, C.-S. Hsu, and T.-H. Lai. Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad-hoc networks. *Mobile Networks and Applications*, 10:169–181, Feb. 2005.

[8] Y.-C. Kuo. Quorum-based power-saving multicast protocols in the asynchronous ad-hoc network. *Computer Networks*, 54:1911–1922, 2010.

[9] G. F. Lawler. *Introduction to Stochastic Processes*. Chapman and Hall/CRC, Taylor and Francis Group, 2006.

[10] L. Lazos, S. Liu, and M. Krunz. Spectrum opportunity-based control channel assignment in cognitive radio networks. In *Proc. of the IEEE SECON Conf.*, pages 1–9, June 2009.

[11] Z. Lin, H. Liu, X. Chu, and Y.-W. Leung. Jump-stay based channel-hopping algorithm with guaranteed rendezvous for cognitive radio networks. In *Proc. of the IEEE INFOCOM Conf.*, 2011.

[12] S. Liu, L. Lazos, and M. Krunz. Thwarting inside jamming attacks on wireless broadcast communications. In *Proc. of the ACM WiSec Conf.*, 2011.

[13] B. Lo. A survey of common control channel design in cognitive radio networks. *Physical Communication*, 4:26–39, 2011.

[14] Y. Manabe, R. Baldoni, M. Raynal, and S. Aoyagi. k-Arbiter: A safe and general scheme for h-out of-k mutual exclusion. *Theoretical Computer Science*, 193:97–112, 1998.

[15] D. R. Stinson. *Cryptography: Theory and Practice*. Chapman and Hall/CRC, Taylor and Francis Group, 2006.

[16] C.-H. Wu, J.-H. Hong, and C.-W. Wu. RSA cryptosystem design based on the Chinese remainder theorem. In *Proc. of Asia and South Pacific Design Automation Conf.*, pages 391–395, 2001.