A Channel Hopping Multi-Channel MAC Protocol for Mobile Ad Hoc Networks

Chih-Min Chao[1] and Hsien-Chen Tsai
Department of Computer Science and Engineering
National Taiwan Ocean University, Taiwan

[1] E-mail: cmchao@ntou.edu.tw, Tel: +886-2-24622192 ext 6651, Fax: +886-2-24623249

Abstract

Although multiple channels are supported in the physical layer, the IEEE 802.11 MAC layer mechanism is designed for using a single channel. Exploiting multiple channels enhances spatial reuse and reduces transmission collisions and thus improves network throughput. Designing a multi-channel MAC protocol is much more difficult than designing a single-channel one. New challenges, such as the channel allocation problem and the missing receiver problem, must be overcome. Existing multi-channel MAC protocols suffer from either higher hardware cost (because of applying multiple transceivers) or lower channel utilization (due to limited transmission opportunity). In this paper, a fully distributed channel hopping solution, the Cyclic-Quorum-based Multi-channel (CQM) MAC protocol, is proposed. We use the cyclic quorum in a novel way and the proposed protocol has several attractive features. First, only a single transceiver is needed for each node. Second, any sender is guaranteed to meet its receiver in a short time. Third, each node's channel hopping sequence is derived from its node ID. This avoids exchanging control messages, such as each node's hopping sequence or available channel list. Fourth, multiple transmission pairs can accomplish handshaking simultaneously. The proposed protocol is simple and efficient. Simulation and real system implementation results verify that our mechanism is a promising multi-channel MAC protocol for mobile ad hoc networks.

Index Terms

Mobile Ad Hoc Networks, IEEE 802.11, Multi-Channel MAC Protocols, Quorum Systems.

I. Introduction

The IEEE 802.11 standard [4] has been widely accepted as a single channel MAC mechanism for mobile ad hoc networks. In the IEEE 802.11 standard, the fundamental mechanism to access the shared medium is the distributed coordination function (DCF) which utilizes the RTS/CTS/DATA/ACK four-way handshake mechanism to solve the hidden terminal problem. If only one channel is supported, this mechanism suffers from many collisions in a heavy-loaded network. Using physical carrier sensing improves the performance of the virtual carrier sensing, if a proper carrier sense range can be found [14], [36], [38], but the collision problem remains. Utilizing multiple channels may help to share the traffic loads among different channels and hence alleviate this problem. Allowing users to use multiple channels can also increase spatial reuse, which increase the aggregate throughput of the network. In fact, IEEE 802.1b and IEEE 802.11a support 3 and 13 non-overlapping channels, respectively. It means that designing a multi-channel MAC protocol is feasible and desirable in IEEE 802.11-based mobile ad hoc networks.

There are many design challenges in multi-channel MAC protocols. An important issue is how and which channel should be selected for data transmission. Solutions to this problem can be classified according to the

TABLE I
CLASSIFICATION OF MULTI-CHANNEL MAC SOLUTIONS.

	Single-Rendezvous	Multi-Rendezvous
Multi-Transceiver	[6], [13], [25], [32], [33], [34], [35]	[5], [18], [19], [24]
Single-Transceiver	[12], [15], [17], [20], [22], [23], [27], [30], [37]	[7], [29], [31], ours

following two factors:

- *single-/multi-transceiver*: Whether a user can utilize multiple transceivers or not.
- *single-/multi-rendezvous*: Whether multiple transmission pairs can always accomplish handshaking simultaneously or not.

Based on such classification, we categorize existing typical solutions and the solution proposed in this paper in Table I.

There are many protocols [5], [6], [13], [18], [19], [24], [25], [32], [33], [34], [35] use multiple transceivers to handle the channel allocation job. Nodes running these protocols are equipped with at least two transceivers. In some protocols [6], [13], [25], [32], [33], [34], [35], one transceiver is tuned to a common control channel which is dedicated for negotiating the channel to be used next. Once the channel is selected, the other transceiver(s) will switch to the negotiated channel for data transmission. These protocols are referred to as the single-rendezvous solutions. An undesirable feature of the multi-transceiver, single-rendezvous category is that the dedicated control transceiver is a bottleneck. Without using a dedicated control channel, in [18], [19], [24], each node fixes one of its transceivers to its own fixed channel, waiting for accepting transmission requests, and the other transceivers are free to switch to any channel to initiate a transmission. Another scheme [5] uses two transceivers; one performs a fast channel hopping and the other performs a slow channel hopping. The fast-hopping transceiver is used for transmission while the slow-hopping one is used for reception. We call these solutions multi-rendezvous ones since handshakings for different transmission pairs can be handled at the receivers' fixed channels simultaneously. The downside of using multiple transceivers is increased hardware cost.

To reduce such cost, a number of studies utilize only one transceiver [7], [12], [15], [17], [20], [22], [23], [27], [29], [30], [31], [37] to solve the channel allocation problem. Some of them [20], [22], [23], [27], [37] use a dedicated control channel for control messages exchanging. In such a method, the dedicated control channel will be either over-loaded or under-utilized if the capacity of the dedicated control channel and the data channels is not distributed properly. Some other proposals [12], [15], [17], [30] use a common control period, similar to the

ATIM window concept in IEEE 802.11 power saving mode (PSM), for nodes to negotiate the data channels. This scheme suffers from the same problem as those use a dedicated control channel. Using either a dedicated control channel or a common control period, these protocols belong to the single-rendezvous class since no concurrent handshaking is allowed. Some other solutions utilize the channel hopping concept to switch among data channels to achieve multi-rendezvous [7], [29], [31]. Different nodes have different channel hopping sequences and two nodes are able to communicate with each other when they switch to the same channel at the same time. A multi-channel protocol called multi-channel MAC protocol (McMAC) uses a common linear congruential generator to build each node's channel hopping sequence [29]. An asynchronous efficient multichannel MAC protocol (EM-MAC) also adopts the pseudorandom channel hopping mechanism [31]. In the slotted seeded channel hopping protocol (SSCH), a channel hopping mechanism is proposed such that any two nodes are guaranteed to have a rendezvous once in a cycle [7]. Avoiding using a common control channel/period, multi-rendezvous solutions do not suffer from the bottleneck problem. It is shown that the multi-rendezvous protocols outperform the single-rendezvous ones [7], [29], [31]. A flaw of these channel hopping protocols is that they may suffer from the missing receiver problem (to be described in the next section). This problem occurs frequently, especially in a heavy-loaded environment [27]. A more detailed review of some representative multi-channel protocols can be found in Section 2.

Since the single-transceiver and multi-rendezvous are attractive features, we focus on designing a multi-channel MAC protocol in this class. In this paper, we propose a cyclic-quorum-based multi-channel MAC protocol (CQM) which overcomes the limitations of existing solutions in this class. Our idea is to use one transceiver to emulate the multi-transceiver, multi-rendezvous solutions. The cyclic quorum system, a kind of quorum systems, has the intersection property. Based on an extension of this property and clever allocations of channels, the CQM protocol can utilize any number of channels and guarantees that any sender can meet its intended receiver in a short time.

The rest of the paper is organized as follows. In Section II, we describe the problem to be solved and provide some reviews. In Section III, our new multi-channel MAC protocol is described in detail. We analyze the performance of SSCH and CQM in Section IV. Simulation and real system implementation results are provided in Section V. Finally, in Section VI, we conclude the paper.

II. Design Issues and Literature Review

A. Design Issues

Besides the channel allocation problem, there are two more challenges in designing a multi-channel MAC protocol: the multi-channel hidden terminal problem and the missing receiver problem. The former exists since the control packets sent on a particular channel are unable to notify neighbors currently tuned to different channels. The latter occurs when the sender fails to access its intended receiver since they do not reside on the same channel. Both problems do not bother the multi-transceiver, single-rendezvous solutions. However, for the multi-transceiver, multi-rendezvous category, the missing receiver problem is solved but the multi-channel hidden terminal problem may still remain. For the single-transceiver, single-rendezvous category, these two problems occur if a dedicated control channel is used. For the single-transceiver, multi-rendezvous class, only the missing receiver is a problem.

We focus on designing a multi-channel MAC protocol in the single-transceiver, multi-rendezvous class. Solutions in this class take channel hopping as the basic mechanism. We consider there are three design issues for a single-transceiver, multi-rendezvous multi-channel MAC protocol. First, we have to guarantee a sender and its intended receiver are able to communicate, i.e., they are guaranteed to hop to the same channel simultaneously. Second, this guarantee should be done with the least controlling overhead. The third design issue is that the proposed solution has to avoid the missing receiver problem. To achieve these requirements, we investigate the possibility of utilizing quorum systems to design a multi-channel MAC protocol. We extend our earlier work [11] by providing a detailed analysis of the proposed CQM protocol, more simulation results, and real system implementation results.

B. Related Work

In the multi-transceiver, single-rendezvous class, a representative one, the dynamic channel assignment (DCA) protocol, which assigns channels in an on-demand manner was proposed in [35]. A node running DCA is equipped with two transceivers, one for control and the other for data transmissions. Since the control transceiver is being listened all the time, each node knows the complete channel usage information and thus can select a free channel whenever needed. An enhancement of DCA which reduces the frequency of control packet transmission can be found in [33]. Another enhancement of DCA which controls the transmission power level of data packets to reduce the power consumption was proposed in [34]. The concept of ATIM window in IEEE 802.11 PSM is

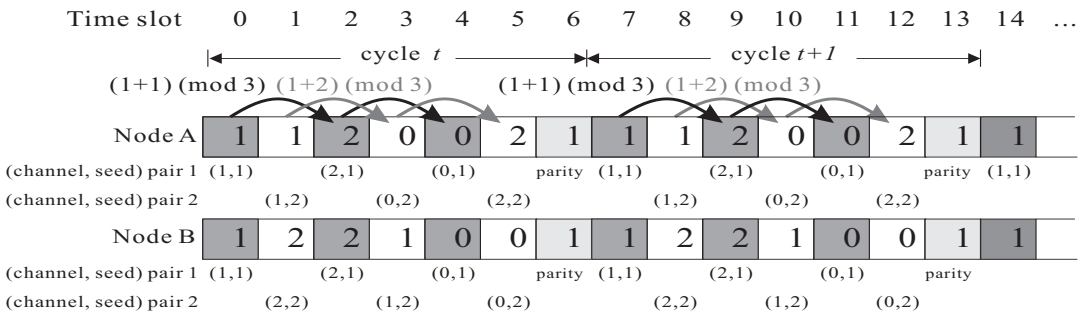

Fig. 1. An example of SSCH operation with 2 (channel, seed) pairs and 3 channels (numbered from 0 to 2).

borrowed by the power-saving multi-radio multichannel MAC protocol (PSM-MMAC) [32]. In PSM-MMAC, each node has one default transceiver serving for traffic indication and channel negotiation. During the ATIM window, each node switches its default transceiver to the default channel to exchange transmission intention through ATIM/ATIM-ACK packets. Also, the source-destination pairs select channels according to the quality and traffic load in each channel. After ATIM window, all channels, including the default channel can serve to exchange data on all the transceivers. The protocols mentioned above have the same control transceiver bottleneck problem and thus system capacity will not be utilized efficiently.

In the multi-transceiver, multi-rendezvous class, the protocols proposed in [18], [19] divide each node's k transceivers into *fixed interface* and *switchable interfaces*. Each node has one fixed interface and is switched to a fixed channel, waiting for receiving data from the other nodes. The other $k-1$ transceivers are switchable interfaces. A data transmission from node A to node B is enabled by tuning one of node A's switchable interfaces to node B's fixed channel. The primary channel assignment based MAC protocol (PCAM) [24] adopts similar mechanism. In PCAM, three transceivers are equipped for each node. The *primary transceiver* stays tuned on a fixed channel. Two nodes communicate with each other using the primary transceiver if their fixed channel is the same. Otherwise, the sender's *secondary transceiver* is switched to the receiver's fixed channel. The third transceiver is used for broadcasting control information. The division of fixed (primary) and switchable (secondary) transceivers makes these protocol multi-rendezvous ones. However, equipping each node with multiple transceivers is undesirable because of increased hardware cost.

To reduce hardware cost, several proposals utilize only one transceiver to solve the channel allocation problem. Some of them belong to the single-transceiver, single-rendezvous class [20], [27], [30], [37]. In the asynchronous multichannel coordination protocol (AMCP) [27], one control channel and n data channels are assumed. Each

node locally maintains an n-entry channel table to keep track of the usage of data channels. Channel negotiation between a transmission pair is achieved through the control channel. Then, both nodes switch to the scheduled channel, say channel x, for data transmission. After data transmission, both nodes will switch back to the control channel and set all data channel except x to be unavailable for a certain period of time. Such settings can avoid the multi-channel hidden terminal problem. Similar to AMCP, the load-balanced MAC protocol (LBM) [37] utilizes one control channel and n data channels. LBM aims to balance load sharing among channels during the channel allocation. Nodes running LBM will use the channel that is available and has the lowest utilization ratio for data transmission. In the Multi-Channel MAC protocol (MMAC) [30], all nodes initially listen on the default channel during the ATIM window. At this window, channel negotiation between a transmission pair is achieved through ATIM/ATIM-ACK/ATIM-RES packets. At the end of ATIM window, both nodes switch to the negotiated channel to fulfill their data transmission. Although there is no dedicated control channel in MMAC, the ATIM window can be considered as a common control period which still produces a bottleneck. An unsatisfactory feature of these single-rendezvous solutions is that the control channel/period becomes a bottleneck which limits the overall network utilization.

Solutions in the single-transceiver, multi-rendezvous class try to mitigate the bottleneck problem. In McMAC [29], nodes generate their hopping sequences by a common linear congruential generator with their MAC addresses as the seeds. The hopping sequences of each node's intended receivers can easily be obtained if their MAC addresses are known. When a node has data to send, it may changes its schedule with probability $P_{deviate}$ to follow the intended receiver's channel hopping sequence. Such a dynamically schedule switching increase rendezvous probability sometimes; however, this scheme fails to solve the missing receiver problem since the intended receiver may also change its hopping sequence.

In SSCH [7], each node hops between channels using its own channel hopping sequences. The channel hopping sequences have been designed such that nodes will overlap with each other at least once in a cycle. Specifically, each node's hopping schedule can be determined by a set of *(channel, seed)* pairs. If there are n available channel in the system, *channel* is an integer in the range of 0 to $n-1$ and *seed* is an integer in the range of 1 to $n-1$. For each pair, the next channel is determined by a modular n adding of $seed$ and the current channel. A parity slot at the end of a cycle is also introduced to prevent the situation that two nodes never switch to the same channel concurrently. In the parity slot, the channel assignment is set to the value of $seed$ for the first pair. If n channels

and k (channel, seed) pairs are used, a cycle contains $kn+1$ slots. Nodes running SSCH exchange their channel hopping schedules with each other every cycle. A node may also change its schedule to its intended receiver's to increase transmission probability. The dwelling time for each hop is set to 10 ms. Within this period, IEEE 802.11 DCF is adopted as the MAC mechanism. Fig. 1 is an example of SSCH operation where each node uses two (channel, seed) pairs and three channels. Initially, node A has the values of (1,1) and (1,2) for the first and the second (channel, seed) pairs, respectively. A cycle of node A's channel hopping sequence is thus given by 1-1-2-0-0-2-1. Following the same mechanism, a cycle of node B's channel hopping sequence is 1-2-2-1-0-0-1. Nodes A and B meet each other at time slots 0, 2, 4, and 6 every cycle.

SSCH avoids the control channel bottleneck by sharing the control overhead among all channels. It is a clever move for multi-channel MAC protocol design. However, maintaining each node's schedule correctly is also an overhead, especially in a high mobility environment. Moreover, allowing a node changing its schedule may produce the missing receiver problem since the intended receiver's schedule may also change.

III. Proposed Multi-Channel MAC Protocol

In this section, we describe our CQM in detail. The CQM is a channel hopping MAC protocol. Nodes running our protocol can choose their channel hopping sequences individually, without signaling through any dedicated control channel or common control period. The same four-way handshake of IEEE 802.11 DCF is adopted which enables nodes running CQM coexist with those running the 802.11 standard. The CQM utilizes the cyclic quorum systems to accomplish channel allocation and to solve the missing receiver problem.

A. Quorum Concept

Quorum systems have been widely used for mutual exclusion in distributed systems [21] and for MAC protocol design in wireless networks [8], [9], [10], [16]. A quorum system can be defined as follows [16].

Definition 1. Given an universal set $U = \{0, ..., n-1\}$, a *quorum system* Q under U is a collection of non-empty subsets of U, each called a *quorum*, which satisfies the *intersection property*:

$$\forall G, H \in Q : G \cap H \neq \emptyset.$$

For example, Q = {{1, 2},{1, 3},{2, 3}} is a quorum system under U = {0, 1, 2, 3}. There are many quorum systems, such as the cyclic quorum system, the grid quorum system, and the torus quorum system. We use the

cyclic quorum system to develop our multi-channel MAC solution since it can provide equal opportunity for a node to transmit and to receive packets, as will be described in Section III-B. To facilitate our description, the following descriptions are needed.

Definition 2. Given a non-negative integer i and a quorum H in a quorum system Q under $U = \{0, ..., n-1\}$, we define $rotate(H, i) = \{j + i \pmod n | j \in H\}$.

Definition 3. A quorum system Q under $U = \{0, ..., n-1\}$ is said to have the *rotation closure property* if

$$\forall G, H \in Q, i \in \{0, ..., n-1\} : G \cap rotate(H, i) \neq \emptyset.$$

For instance, the quorum system Q = {{0, 1, 2},{0, 1, 3},{1, 2, 3}} under {0, 1, 2, 3} has the rotation closure property, while the quorum system Q' = {{0, 1},{0, 2},{0, 3},{1, 2, 3}} under {0, 1, 2, 3} does not possess the rotation closure property because {0, 1} $\cap$ rotate({0, 3}, 3) $= \emptyset$.

A cyclic quorum system is constructed from a difference set. The definitions of difference set and cyclic quorum system are as follows.

Definition 4. A subset $D = \{d_1, ..., d_k\}$ of Z_n is called a *difference set* under Z_n if for every $e \neq 0 \pmod n$ there exist at least two different elements d_i and $d_j \in D$ such that $d_i - d_j = e \pmod n$.

Definition 5. Given any difference set $D = \{d_1, ..., d_k\}$ under Z_n, the *cyclic quorum system* defined by D is $Q = \{G_0, ..., G_{n-1}\}$, where $G_i = \{d_1 + i, ..., d_k + i\} \pmod n$, $i = 0, ..., n-1$.

For instance, $D = \{0, 1, 3\}$ is a difference set under Z_6[1]. The set $Q = \{G_0, G_1, ..., G_5\}$, where $G_0 = D$, $G_1 = \{1, 2, 4\}$, $G_2 = \{2, 3, 5\}$, $G_3 = \{3, 4, 0\}$, $G_4 = \{4, 5, 1\}$, and $G_5 = \{5, 0, 2\}$, is a cyclic quorum system under Z_6. And G_i, $i = 0, ..., 5$, is a cyclic quorum.

Theorem 1. *The cyclic quorum system satisfies the rotation closure property.*

[1]Here we verify that D is a difference set: $e = \{1, 2, 3, 4, 5\}$, we have $1 \equiv 1 - 0 \pmod 6$, $2 \equiv 3 - 1 \pmod 6$, $3 \equiv 3 - 0 \pmod 6$, $4 \equiv 1 - 3 \pmod 6$, $5 \equiv 0 - 1 \pmod 6$.

Proof: The cyclic quorum system has the intersection property since it is a quorum system itself. By definition, for any quorum G belongs to a cyclic quorum system Q, $rotate(G, i)$ is also a quorum in Q for any i. With the intersection property, the cyclic quorum has the rotation closure property. ■

We also list a cyclic quorum system under Z_8 here: Q' = {{0, 1, 2, 4},{1, 2, 3, 5},{2, 3, 4, 6},{3, 4, 5, 7},{4, 5, 6, 0},{5, 6, 7, 1},{6, 7, 0, 2},{7, 0, 1, 3}}. The minimal difference set under Z_n for n = 4 to 111 can be found in [21].

B. The CQM Protocol

Now we are ready to present our CQM. First we list the assumptions of our protocol below:

- Totally m channels are available and all of them have the same bandwidth.
- Each node is equipped with one half-duplex transceiver which is able to switch to any channel dynamically.
- Nodes are time synchronized. Clock synchronization can be achieved by schemes such as [26], [28] or by using GPS devices.
- Each node knows the identifications (IDs) of its one hop neighbors.

To enable a communication, a transmission pair must tune to the same channel at the same time. When they meet each other, the four-way handshake can be applied to fulfill data transfer. In essence, the most critical task is the joint allocation of channel/time for all the nodes in the network. In CQM, time is divided into a series of cycles. Each cycle consists of n time slots, numbered from 0 to $n-1$. The value of n is determined by the integer set from which the adopted difference set is derived. For example, if a difference set of Z_6 is adopted, the value of n is six. The length of a time slot is long enough to transmit at least one data packet. For each cycle, time slots are partitioned into *default slots* and *switching slots*. At default slots, a node will stay on its *default channel*, waiting for transmission requests. At switching slots, a node may switch to its intended receiver's default channel. Each node's default channel is selected from its node ID. To solve the missing receiver problem, what we need to do is providing a rendezvous between a sender and its intended receiver. To achieve this goal, we use a cyclic quorum G_i under Z_n to identify a node's default slots. Specifically, for any node $i \in V$, where V is the set of nodes in the network, node i's default channel (denoted as DC_i) and default slots (denoted as DS_i) are chosen as follows.

$$DC_i = \textit{node\_ID}_i \pmod m,$$

$$DS_i = G_j,\ j=\textit{node\_ID}_i \pmod n, \forall i \in V.$$

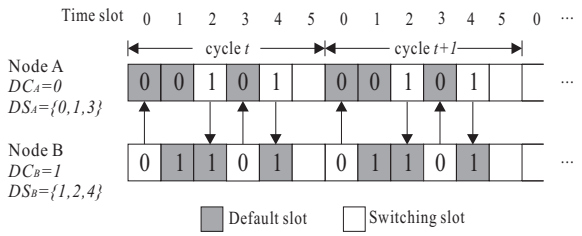

Fig. 2. An example of CQM operation under Z_6 with 3 channels. The number in each slot represents the channel to be switched to.

where $node\_ID_i$ is the ID of node i.

The effectiveness of the CQM protocol relies on the overlapping of the sender's switching slots and the receiver's default slots. The intersection property of the cyclic quorum system is not enough for CQM since it only guarantees the overlapping of default slots. In the following, we justify the correctness of our CQM.

Definition 6. For a given difference set $D = \{d_1, ..., d_k\}$ under Z_n, the complement set of D, $\overline{D}$, is defined as $Z_n - D$. That is, $\overline{D} = \{b_1, ..., b_m\}$, where $k + m = n$, for $i = 1, ..., k$, $j = 1, ..., m$, $d_i \neq b_j$.

For example, $D = \{0, 1, 3\}$ is a difference set under Z_6 with complement set $\overline{D}$ = {2, 4, 5}.

Theorem 2. *Given a cyclic quorum system $Q = \{G_0, ..., G_{n-1}\}$ under Z_n, for i, j = 0, ..., n-1, then $G_i \cap \overline{G_j} \neq \emptyset$ if and only if $G_i \neq G_j$.*

Proof: In the forward direction, we prove it by contraposition. If $G_i = G_j$, $G_i \cap \overline{G_j} = \emptyset$. This proves the forward direction.

In the backward direction, we also prove it by contraposition. If $G_i \cap \overline{G_j} = \emptyset$, according to definition 5, $G_i = G_j$. This proves the backward direction. ■

Theorem 2 verifies the feasibility of CQM. It is guaranteed that a sender's switching slots and its receiver's default slots intersect. The missing receiver problem is solved accordingly. In addition to solving the multi-channel MAC problems in a totally distributed way, our CQM solves them efficiently. We justify this by analyzing the performance of our CQM and another channel hopping MAC protocol, SSCH, in Section IV.

The design of CQM achieves multi-rendezvous in that, at any time slot, multiple transmission pairs can concurrently complete handshaking at the receivers' default channels. This joint channel/time allocation has some more beneficial features. First, given that the node IDs are randomly distributed, the default channels for

all the nodes in the network can also be evenly distributed among all available channels. It means that the traffic load can be evenly shared by all available channels. Second, the scattered overlapping of default and switching slots for different pairs help to reduce packet collisions. Third, it is easy to retrieve a neighbor's channel hopping sequence if the neighbor's ID is available. This greatly reduces the control overhead, such as exchanging each node's hopping sequence or available channel list.

Fig. 2 is an example of CQM operation under Z_6 with 3 channels (numbered from 0 to 2). Nodes A and B, with IDs 0 and 1, respectively, are within each other's transmission range. Each node's default channel and default slots are shown in Fig. 2, supposing that we choose the difference set $\{0, 1, 3\}$ under Z_6 as G_0. The marked time slots are default slots. The number in each slot is the channel that should be switched to. When node A has packets pending for node B, node A switches to node B's default channel (channel 1) at its switching slots. In this example, node A can send some packets to B at time slots 2 and 4 of each cycle. Similarly, if node B has packets to A, the transmission can be done in time slots 0 and 3 by switching B's transceiver to channel 0.

It should be noted that two nodes with the same cyclic quorum have no overlapping of default and switching slots. They may never meet each other if their default channels are different. In a network where the nodes' IDs are uniformly distributed, two nodes may not be able to communicate with each other with a probability of $\frac{m-1}{mn}$. For example, with m=3 and n=6, one out of nine neighbors will be unreachable for each node. Although the probability is not high, the unreachable problem still needs to be solved. This problem can be solved by finding a multi-hop route to relay their traffic, if we handle it in the network layer. This means route discovery between these two nodes must be applied. If this route discovery fails, the node that has pending traffic to the other can temporarily change its quorum until the traffic is delivered. If both nodes have pending traffic to the other, it is possible that they change to the same quorum again. To solve this, one of the nodes, say the one with a larger ID, can change its quorum for the second time. There is a rare situation where a node is isolated because of having a common cyclic quorum but different channel with all its neighbors. In such a case, the isolated node can randomly choose another quorum and announces this change to its neighbors.

C. CQM Variations

Providing broadcast is not easy for channel hopping protocols. A typical solution is to transmit the packet on each channel separately [7]. This may produce longer delay and higher traffic contention. Besides this traditional

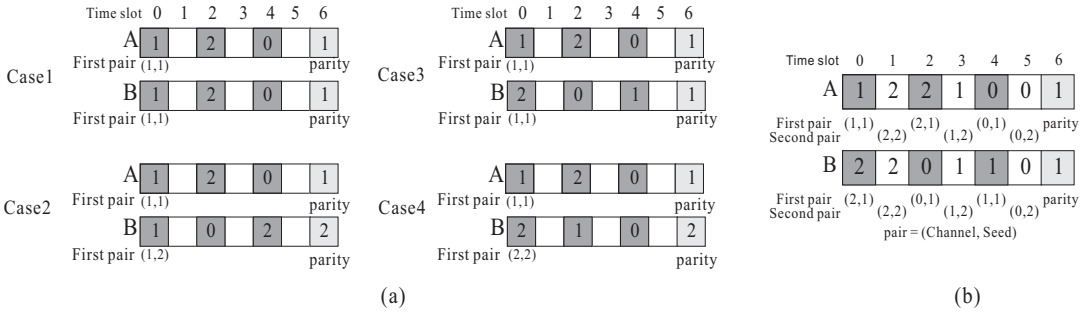

Fig. 3. Relations between the schedules of two nodes with (a) a single (channel, seed) pair and (b) two (channel, seed) pairs. The number in each slot represents the channel to be switched to.

strategy, a possible way for our CQM to support fast broadcast is to arrange a broadcast slot periodically. For instance, a broadcast slot can be allocated once per second (which is equivalent to allocate a broadcast slot every 17 cycles for a cyclic quorum under Z_6). The frequency of broadcast slot depends on the ratio of broadcast traffic and should be identical for all nodes in the same network.

To support a large amount of traffic transfer between two nodes, e.g., file transfer and streaming services, the sender can temporarily change to the receiver's channel hopping sequence after their first communication slot. Supposing that node A has a large file to node B, both nodes follow their original channel hopping sequences until they communicate with each other at some time slot. For the ensuing slots, node A can follow node B's schedule until the transmission is complete. To go one step further, the receiver can change all its switching slots to default slots to enable 100% meeting probability with the sender.

IV. Performance Analysis

In this section, we compare the channel hopping scheme of CQM with that of the SSCH. Since changing a node's channel hopping sequence is possible for both protocols, for simplicity's sake, we compare the performance when nodes do not change their channel hopping sequences. The comparison with a complete SSCH where a node's schedule may change to its receiver's will be presented in Section V. We consider an environment with only two nodes (A and B) and both nodes have data packets for each other. For SSCH, the overhead of broadcasting each node's schedule is ignored.

Two criteria are used in this comparison:

- *The expected ratio of meeting slots in a cycle, denoted as R(m).*
- *The expected number of slots a node has to wait before encountering a meeting slot, denoted as E(w).*

For example, for each cycle in Fig. 2, the default slots of node A and the switching slots of node B overlap

at time slots 0 and 3. On the contrary, the default slots of node B and the switching slots of node A overlap at time slots 2 and 4. The ratio of meeting slots between nodes A and B in a cycle is (2+2)/6=0.667. If node A has a packet for B, it can be transmitted at time slot 2 or 4. That is, if the packet arrives at time slot 2 or 4, it can be delivered immediately. If the packet arrives at time slot 0, 1, or 3, the number of time slots to be waited is 2, 1, or 1, respectively. If the packet arrives at time slot 5, three time slots must be waited before it can be sent at time slot 2 of the next cycle. The expected value of waiting time slots is 2+1+0+1+0+3/6=1.167.

A. Analysis of SSCH

A node running SSCH determines its channel schedule by a (channel, seed) pair. Nodes may use multiple pairs to construct their channel hopping sequences. Each node selects its (channel, seed) pairs independently. For each pair, there are four combinations between the schedules of nodes A and B:

- Case 1: They use identical channel and seed.
- Case 2: They use identical channel but different seeds.
- Case 3: They use different channels but identical seed.
- Case 4: They use different channels and seeds.

Fig. 3 is a snapshot of SSCH hopping schedules using three channels and two (channel, seed) pairs. We first concentrate on the first pair as shown in Fig. 3(a). It can be seen that nodes A and B do not meet each other often except for case 1 which has four meeting slots, including three non-parity time slots and one parity slot. For cases 2 and 4, there is one meeting non-parity slot. In case 2, two nodes meet each other at the first non-parity slot. In case 4, they meet at one of the non-parity slots except the first one. Nodes A and B meet each other only at the parity slot in case 3.

When each node has two (channel, seed) pairs, there are $4 \times 4 = 16$ possible case combinations between A and B. Fig. 3(b) is an example that the relation between the schedules of nodes A and B is case 3 for the first pair and is case 1 for the second pair. The number of meeting slots and the number of slots to be waited, as well as R(m) and E(w), for two pairs with three and five channels are listed in Table II. From this table, it is obvious that using fewer number of channels has higher R(m) and lower E(w).

We have also developed the general form of R(m) for two nodes using n channels and k pairs. The first pair is calculated individually because the parity slot is associated with it. The number of meeting slots of the first pair (including the parity slot) is $n+1$ for case 1 and is one for the other three cases. For the other pairs, the number

of meeting slots for cases 1, 2, 3, and 4 is n, 1, 0, and 1, respectively. Denote the probability of two nodes fall in case i as p_i, for $i = 1$ to 4, we have $p_1 = \frac{1}{n} \cdot \frac{1}{n-1}$, $p_2 = \frac{1}{n} \cdot \frac{n-2}{n-1}$, $p_3 = \frac{n-1}{n} \cdot \frac{1}{n-1}$, and $p_4 = \frac{n-1}{n} \cdot \frac{n-2}{n-1}$. The general form of R(m) with n channels and k pairs is as follows.

$$\begin{aligned} R(m) =& \frac{1}{kn+1}((n+1) \cdot p_1 + 1 \cdot p_2 + 1 \cdot p_3 + 1 \cdot p_4 \\ &+ \sum_{i=1}^{k-1}(n \cdot p_1 + 1 \cdot p_2 + 0 \cdot p_3 + 1 \cdot p_4)) \\ =& p_1 + \frac{k}{kn+1}p_2 + \frac{1}{kn+1}p_3 + \frac{k}{kn+1}p_4 \end{aligned} \tag{1}$$

Special cares need to be taken for case 4 when calculating E(w). The position of the meeting non-parity slot in case 4 may be different for different pairs of nodes. This implies different numbers of waiting slots accordingly. For example, the meeting slot for three channels and two pairs can be located at the second or the last non-parity slot. Thus, two values of the number of waiting slots can be obtained when there is one case 4 in the case combination. If the case combination is case 4 + case 4, four values of the number of waiting slots can be obtained. When using five channels and three pairs, the meeting slot of case 4 for each pair can be either of the four non-parity slots (that is, the second, the third, the fourth, or the last non-parity slot). An element of case 4 in a case combination implies four numbers of waiting slots. There are totally 4^3 possibilities if case 4 is the only element of the case combination (case 4 + case 4 + case 4). In general, if n channels and k pairs are utilized, there are $(n-1)^k$ possibilities if case 4 is the only element of the case combination. Moreover, if k pairs are utilized, there are totally 4^k case combinations. Since it is not easy to obtain a concise general form for E(w), we observe the impact on number of pairs by using some particular scenarios. The results using two to four pairs with three and five channels are shown in Table III. The values for R(m) are also provided for reference purposes. We see that a little negative effect is observed when the number of pairs is increased.

B. Analysis of CQM

Two nodes running CQM meet each other if one's default slots overlap the other's switching slots, and vice versa. Here we calculate the number of meeting slots and waiting slots of CQM under Z_6 and Z_8. We consider the cyclic quorum systems of $G_0 = \{0, 1, 3\}$ under Z_6 and $G_0 = \{0, 1, 2, 4\}$ under Z_8. We fix on the cyclic quorum G_0 and compute the overlapping and waiting slots with the other cyclic quorums. In CQM, the number of channels is irrelevant to R(m) and E(m). The complete results are shown in Table IV[2].

[2]We have tested different G_0 under Z_6 and Z_8, as well as fixing on G_i other than G_0 and the same results are obtained.

TABLE II
PERFORMANCE OF SSCH WITH 3 OR 5 CHANNELS AND 2 PAIRS.

	3 Channels, 2 Pairs		5 Channels, 2 Pairs	
Case Combinations (Pair 1 + Pair 2)	Meeting Slots	Waiting Slots	Meeting Slots	Waiting Slots
1 + 1	7	0	11	0
1 + 2	5	2	7	4
1 + 3	4	3	6	5
1 + 4	5	2	7	4
2 + 1	4	3	6	5
2 + 2	2	15	2	45
2 + 3	1	21	1	55
2 + 4	2	10	2	30
3 + 1	4	3	6	5
3 + 2	2	11	2	37
3 + 3	1	21	1	55
3 + 4	2	12	2	32
4 + 1	4	4	6	6
4 + 2	2	12	2	32
4 + 3	1	21	1	55
4 + 4	2	13.5	2	35.75
Results	R(m)=0.36	E(w)=1.67	R(m)=0.20	E(w)=3.21

R(m) and E(w) are the expected values of all combinations.

TABLE III
PERFORMANCE OF SSCH WITH 3 OR 5 CHANNELS AND 2 TO 4 PAIRS.

Number of Pairs	Number of Channels	R(m)	E(w)
2	3	0.3571	1.6746
	5	0.2046	3.2118
3	3	0.3500	1.8477
	5	0.2031	3.5934
4	3	0.3461	1.9160
	5	0.2024	3.7817

TABLE IV
PERFORMANCE OF CQM UNDER Z_6 AND Z_8.

	Z_6		Z_8	
Quroum Combinations	Meeting Slots	Waiting Slots	Meeting Slots	Waiting Slots
G_0 + G_1	4	7	4	16
G_0 + G_2	4	6	4	13
G_0 + G_3	2	15	6	8
G_0 + G_4	4	10	4	21
G_0 + G_5	4	6	6	15
G_0 + G_6			4	21
G_0 + G_7			4	12
Results	R(m)=0.6	E(w)=1.25	R(m)=0.57	E(w)=1.89

From Table IV, we see the ratios of meeting slots to a cycle is 0.6 and 0.57 under Z_6 and Z_8, respectively. The expected numbers of waiting slots are 1.25 and 1.89, respectively. We conclude that our CQM performs better under Z_6 and is superior to the SSCH protocol.

V. Performance Evaluation

A. Simulation

We have tried to implement our protocol using the well-known ns-2 simulator. Most existing multi-channel protocols implemented by ns-2 use multiple transceivers. Some provide source codes (Hyacinth [2] and TeNS [3]). We do not find any available single-transceiver multi-channel ns-2 module. Since no existing module can be utilized, we try to develop our own single-transceiver module to access multiple channels. Unfortunately, we failed to do so. We have tried to revise the Hyacinth module to support our CQM. Although working very hard for over one year, we still do not create a proper single-transceiver multi-channel module for ns-2. The problem we can't solve is that we failed to implement our channel hopping mechanism using one transceiver. We have also tried to implement our CQM based on another ns-2 extension, CRCN [1], which is designed for cognitive radio networks. It is claimed that single-transceiver channel hopping is supported in this extension; however, after carefully tracing the source codes, we do not find how channel hopping is achieved and do not succeed to modify it. Unable to use existing simulator, we have implemented a simulator[3] to evaluate the performance of the proposed CQM protocol. The SSCH and McMAC protocols were also implemented for comparison purposes. We have tried to implement the mechanisms of SSCH and McMAC as faithful as possible. For SSCH, methods such as the dynamically schedule switching, per-neighbor FIFO queues, receiving slots, once-per-slot schedule broadcasting for each node, etc. have been implemented. Whenever a sender wants to change its own schedule, one non-receiving slot is changed to the receiver's schedule. If all slots are receiving slots, any slot can be changed. In our implementation, if multiple slots are allowed to be changed, we always select the next available slot. For McMAC, mechanisms including the linear congruential generator, the dynamically schedule switching (with probability $P_{deviate}$), per-neighbor FIFO queues, etc. have been implemented.

In our simulations, nodes are uniformly placed in an area of 800 m $\times$ 800 m. Each node may act as a sender where the destination is chosen from its one-hop neighbors. The number of packets for each transmission is 500. The packet size is set to 512 bytes and nodes are supplied with constant bit rate traffic of 20,000 packets

[3]The source code is available at http://140.121.198.19/hsccl/cqm/cqm.rar

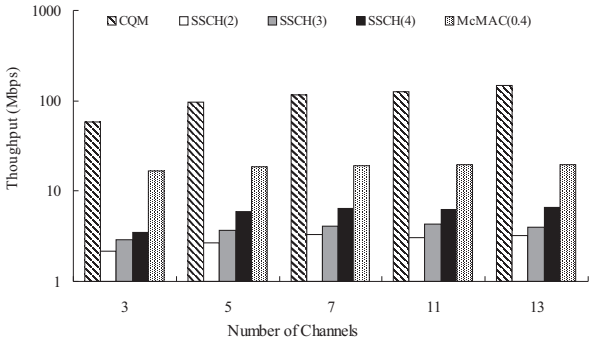

Fig. 4. Impact on number of channels

per second. This means the traffic load is about 80 Mbps. A node keeps a separate FIFO queue for each of its neighbors. The transmission range of a node is 250 meter. A time slot duration is 10 ms long. The capacity for each channel is 11 Mbps. The number of channels is set to 3, 5, 7, 11, or 13 in all three protocols. The channel switching delay is 80 μs. For CQM, a cyclic quorum system under Z_6 is implemented. The number of (channel, seed) pairs in SSCH is between 2 and 4, and the initial values of all pairs are randomly chosen. For McMAC, the default discovery channel is set to channel 0. The probability of temporary deviation, $P_{deviate}$, is varied from 0.2 to 0.8 (identical to the setting in [29]) and only the one that has the best performance is shown in the following fiugers. The notation *SSCH(n)* means n (channel, seed) pairs are utilized in the SSCH protocol while *McMAC(p)* means $P_{deviate}$ is set to p in McMAC. The metric used in our simulation is *Aggregate Throughput*. Each point in the figures, if not specified, is an average of 20 simulation runs with each simulating 50 seconds.

Below, we made observations from six aspects.

A) Impact on Number of Channels: First of all, we varied the number of channels being used in different protocols to observe the effect. Totally 100 nodes were deployed. The results on aggregate throughput are shown in Fig. 4. It is obvious that CQM performs the best which is followed by McMAC and SSCH. We believe the throughput enhancement of CQM results from eliminating the missing receiver problem. When a node A running SSCH changes its channel hopping sequence, the other nodes do not aware of such a change until they receive the new channel schedule. Because the channel schedule information is not guaranteed to be received by all the neighbors, the missing problem occurs and produces significant performance degradation. McMAC performs better than SSCH since nodes running McMAC retain some of their original channel hopping sequences if $P_{deviate} \neq 1$. With such a mechanism, although nodes running McMAC suffer from the missing receiver problem, it is not as

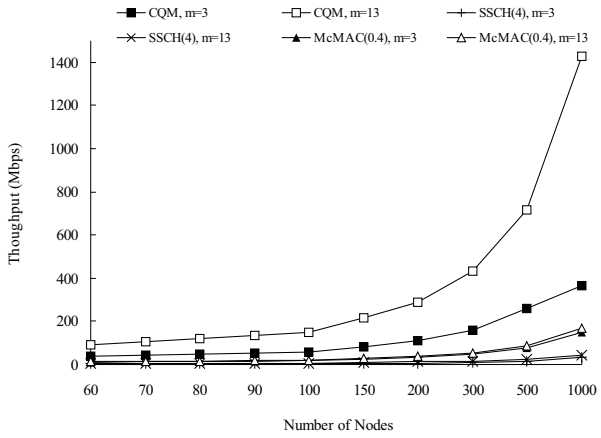

Fig. 5. Impact on number of nodes

severe as what is encountered for nodes running SSCH. Note that the exact slots for nodes running McMAC follow their original schedule are not available for their neighbors, which means nodes cannot precisely estimate the channels their neighbors are resided in. In CQM, each node's default channel and default slots are deterministic and is known by its neighbors. That is, the rendezvous between two nodes can be correctly specified. The results verify the superiority of such a design. As the number of channels increases, the packet collision probability is reduced since nodes are distributed among more channels. Therefore, increased throughput is expected as more channels are utilized.

B) Impact on Number of Nodes: In this experiment, we varied the number of nodes from 60 to 1000 to examine the influence. We have tested 3 and 13 channels to simulate networks with a small and a large number of available channels, respectively. In SSCH, four (channel, seed) pairs are utilized. The results are shown in Fig. 5. As the number of nodes increases, the number of transmission pairs increases. Although the collision probability also increases, its impact is much less than that of increased transmission pair and thus all the protocols achieve higher throughput. Again, successfully solving the missing receiver problem, CQM significantly outperforms the others and the gaps between CQM and the other two protocols become larger when more nodes are deployed in the network.

C) Impact on Transmission Range: We have also changed each node's transmission range to observe the effect on different spatial reuse characteristics. We have deployed 100 nodes and tested three different transmission ranges: 150 m, 250 m, and 350 m, which build a 7-hop, 5-hop, and 3-hop network, respectively. As shown in Fig. 6, the throughput enhancement of CQM over McMAC and SSCH is obvious. A larger transmission range

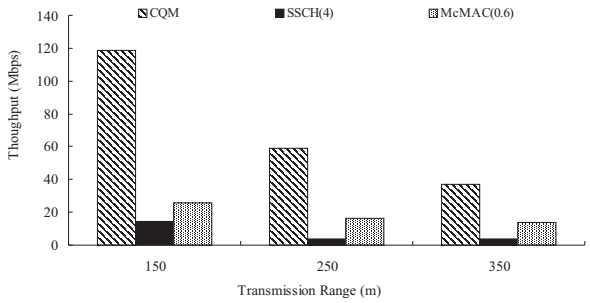

Fig. 6. Impact on transmission range

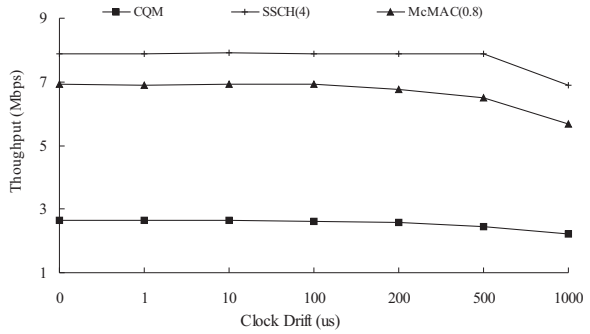

Fig. 7. Impact on clock drift

results in limited spatial reuse, which lowers the aggregate throughput.

D) Impact on Clock Drift: Similar to SSCH and McMAC, synchronization among nodes is needed in CQM. In this experiment, we observe the effect of imperfect synchronization. We have adopted the same two-node topology as in [7]. Clock skew is introduced such that the sender's clock is always faster than that of the receiver's by 0 to 1 ms. The throughput between these two nodes running different protocols can be found in Fig. 7. It is obvious that the throughput achieved by all three protocols is not affected much by clock drifts of less than 200 μs. Existing synchronization protocol is able to keep the clock drift below 200 μs [28]. The performance degraded a little when clock drift is very high (1 ms). This experiment verify that CQM as well as McMAC and SSCH are robust under imperfect synchronization. Note that SSCH has achieved the highest performance since there no missing receiver problem in this experiment and thus the sender can meet the receiver at every slot. In fact, McMAC(1) and a variation of CQM (enabling the sender to follow the receiver's schedule) can both achieve the same high throughput. To provide some more information, we have also showed the performance of McMAC(0.8) and pure CQM under imperfect synchronization. The sender running McMAC(0.8) is able to communicate for around 80% of time while the sender running CQM can deliver its traffic in one third of the

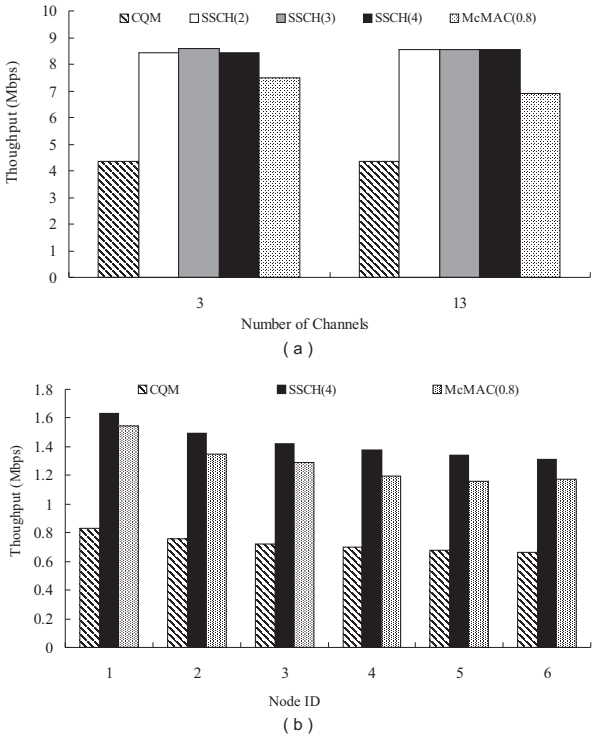

Fig. 8. Aggregate throughput for (a) the whole network and (b) individual nodes of the star topology

total time slots.

E) Impact on Topoogy/Number of Flows: We have also applied three protocols on three different topologies which generate different number of flows. Remember that, for McMAC, different $P_{deviate}$ have different performance and only the best one is showed. The first topology is a star-like one: Node 0 located at the center is the only sender with constant bit rate of 8 Mbps and has the same volume of traffic to any of its six neighbors nodes with IDs 1, 2, ..., 6. Fig. 8 (a) shows the results of different protocols using 3 and 13 channels. We can see that SSCH has the highest throughput since the missing receiver problem does not exist in this scenario. Running McMAC(0.8), node 0 delivered its traffic for 80% of time and thus the achieved throughput is behind that of SSCH. For CQM, node 0 can only transmit at switching slots. Because switching slots occupy half of total slots, the throughput achieved by CQM is half of that of SSCH. The throughput for individual nodes using three channels in the star topology is shown in Fig. 8 (b). We can see that the channel capacity is roughly shared among nodes 1 to 6 for all the protocols.

In the circle topology, four nodes with IDs 0, 1, 2, and 3 form a circle where each node is only able to communicate with its adjacent two nodes. Each node i receives data from node $i-1$ (mod 4) and transmits data

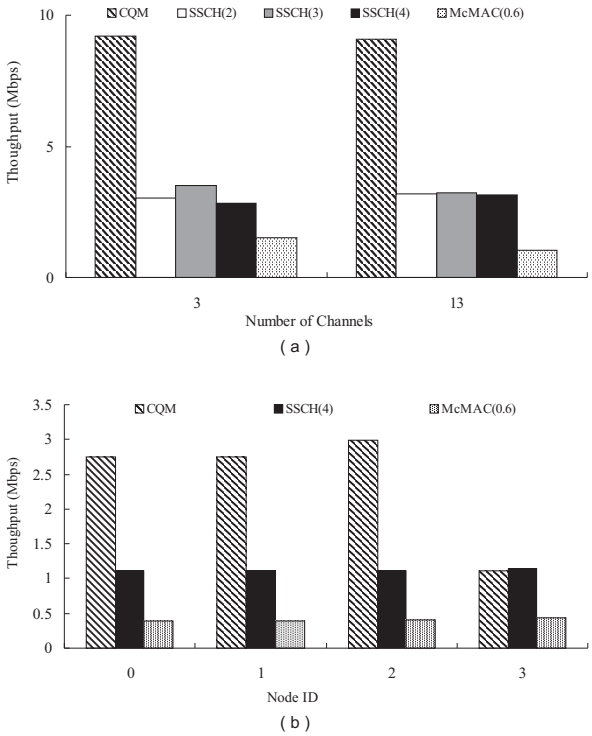

Fig. 9. Aggregate throughput for (a) the whole network and (b) individual nodes of the circle topology

to node $i + 1$ (mod 4). The results for the circle topology using 3 and 13 channels can be found in Fig. 9 (a). A detailed view of the throughput for different nodes using 3 channel is shown in Fig. 9 (b). Obviously, CQM performs the best. When running CQM, node 3 achieves the lowest throughput when compared with the other nodes. It is because only one of node 3's switching slots overlaps one of node 0's default slots while the other nodes have two such overlapping to their receivers. It is a surprise when we found that McMAC performs worse than SSCH. After carefully examining each node's schedule changes during the simulation, we found nodes running SSCH in the circle topology have a very high probability (86%) to form a uniform channel hopping sequence. This results from each node has only one receiver and the time to learn its schedule is uncertain. We illustrate a scenario that produces such a uniform schedule here. Nodes learn their receivers' schedule in the following sequence: 0, 3, 2, and 1. That is, node 0 learns node 1's schedule the earliest, node 3 learns node 0's new schedule schedule one or more slots latter, etc. When node 1 finally learns node 2's schedule, it will find that the schedule is the same as its own – all the nodes adopt the same schedule after node 2 changing its schedule. Having the uniform schedule, in each slot, two out of four nodes running SSCH may communicate with each other (the other two will keep silent because of the IEEE 802.11 CSMA/CA mechanism). For McMAC, poor

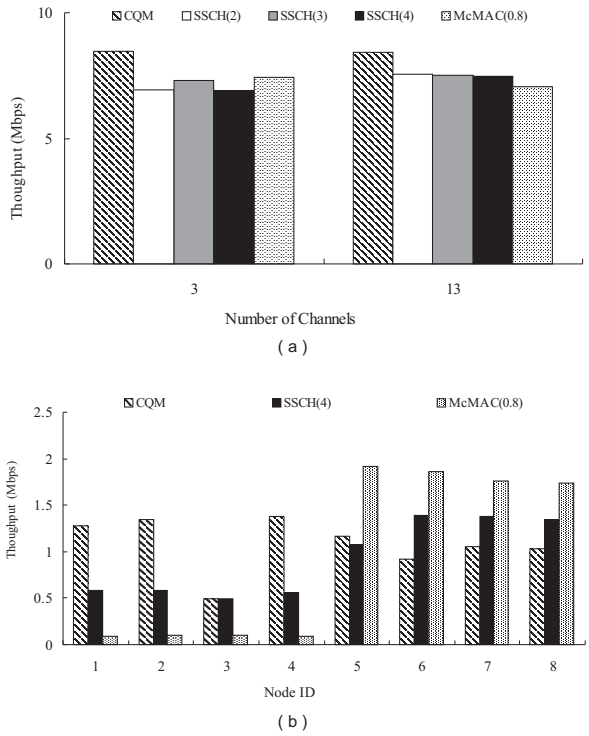

Fig. 10. Aggregate throughput for (a) the whole network and (b) individual nodes of the single-bottleneck topology

performance is observed since nodes suffer from severe missing receiver problem.

In the single-bottleneck topology, Node 0 located at the center receives data from four nodes with IDs 1 to 4 on the left and transmits to the other four neighbors with IDs 5 to 8 on the right. The results for this topology using 3 and 13 channels are shown in Fig. 10 (a). A detailed view of the throughput for different nodes using 3 channel is also presented in Fig. 10 (b). Note that in Fig. 10 (b), for comparison purposes, the throughput for node 0 is represented in terms of the source nodes 1 to 4 that generate traffic to node 0. In this topology, the three protocols do not differ much in achieved performance while CQM still outperforms the others. Because of the missing receiver problem, SSCH and McMAC favor the traffic flows from node 0 to nodes 5 to 8. On the contrary, CQM provides much balanced service to all different flows.

F) Impact on Mobility: Next, we investigate the effect of nodes' movement. We model node mobility by the Random Waypoint model. 100 nodes are randomly distributed in an area of 1 km $\times$ 1 km. Nodes randomly choose a target and move toward it at a speed of 1 to 20 m/s. When the target point is reached, a node stays for 0 to 10 seconds. We observe the time and overhead required for a node to discovery all its one-hop neighbors' channel schedules. A total of 3 channels are available for each node. For SSCH, each node broadcasts its channel

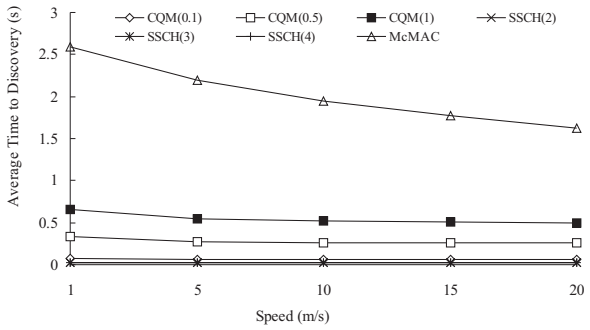

Fig. 11. Neighbor discovery time

schedule once per slot. A node running McMAC broadcasts a beacons once every second and an additional beacon is sent once on its default discovery channel every two seconds. For CQM, three different broadcast periods are implemented, that is, a broadcast slot is allocated every 0.1, 0.5, and 1 second. Each point in the figure is an average of 20 simulation runs with each simulating 600 seconds.

As shown in Fig. 11, nodes running SSCH use the least time to find their neighbors since they broadcast channel schedules frequently. CQM also has pretty good performance where the time needed to discovery neighbors for nodes running CQM(0.1) is very close to what is need for nodes running SSCH. It takes much longer time for nodes running McMAC to identify their neighbors. It is because McMAC has the lowest frequency of schedule broadcasting. Besides, nodes running McMAC and SSCH do not possess broadcast slots. Some nodes may miss some of the schedule advertisements, which reduces the effectiveness of channel schedule advertisements. Fig. 12 shows the overhead spent to achieve neighbor discovery for different protocols. As expected, nodes running SSCH have the highest overhead due to the most frequent schedule exchange. McMAC produces slightly higher overhead than CQM(1) but lower than CQM(0.5). When consider Fig. 11 and Fig. 12 together, we comment that all the three protocols can handle node mobility well while CQM achieves it in an efficient way.

B. Real System Implementation

To verify if the simulation results are trustworthy, we have made a real system implementation. We implemented CQM, McMAC, and SSCH in TinyOS 2.x on Octopus II platform. In our implementation, 10 to 30 nodes were randomly deployed to form a 4-hop (10 nodes) or a 5-hop (20 and 30 nodes) ad hoc network. Each node may act as a sender where the destination is chosen from its one-hop neighbors. Since Octopus II uses an IEEE 802.15.4-compatible CC2420 radio chip, the data packet size is set to the maximum of 127 bytes where the

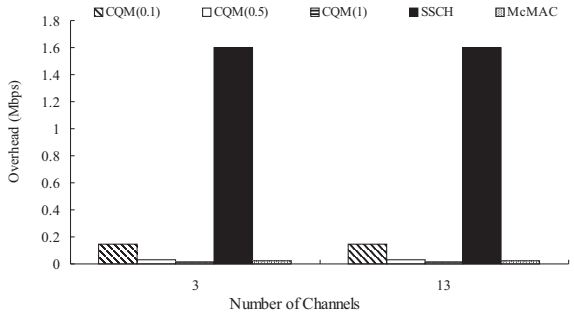

Fig. 12. Neighbor discovery overhead

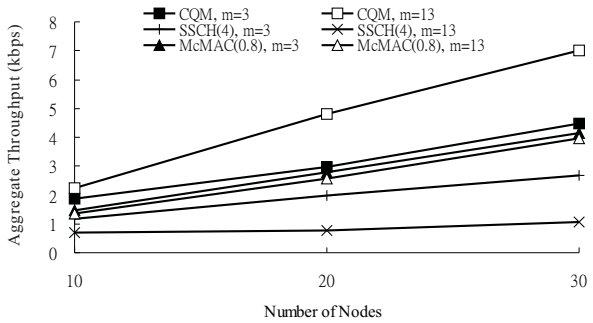

Fig. 13. Impact on number of nodes in real system implementation

payload size is 114 bytes. Constant bit rate traffic of 100 packets per second is applied. The transmission rate of CC2420 is 250 kbps which is unchangeable. The number of channels is set to 3 and 13. A time slot is 1 second long. Each experiment in our implementation ran for 300 seconds while a simple synchronization algorithm (all nodes synchronized to a fixed node in each channel) was executed in each slot to keep nodes synchronized. In CQM, we choose the difference set $\{0, 1, 3\}$ under Z_6 as G_0. The number of (channel, seed) pairs in SSCH is set to 4. For McMAC, the parameter $P_{deviate}$ is set to 0.8. Similar to our simulator, we have implemented SSCH and McMAC as faithful as possible.

The results of the impact of number of nodes can be found in Fig. 13. A higher throughput is achieved for all the protocols when the number of nodes increases. As expected, CQM still performs the best while SSCH the worst. We also varied the number of channels to observe the effect. The number of nodes is set to 30 in this implementation. The results are shown in Fig. 14 where CQM achieves the best throughput again. A higher throughput can be obtained by CQM when more number of channels is available. Note that the throughput for McMAC and SSCH decreases as the number of channels increases, which is different from our simulations.

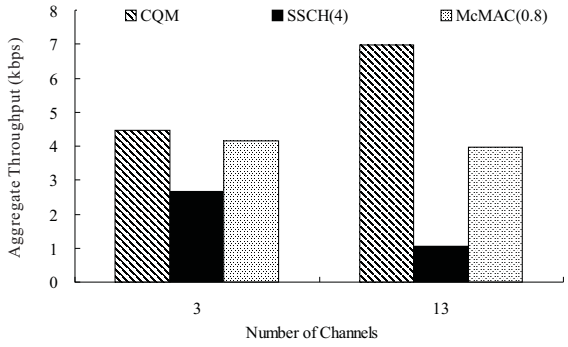

Fig. 14. Impact on number of channels in real system implementation

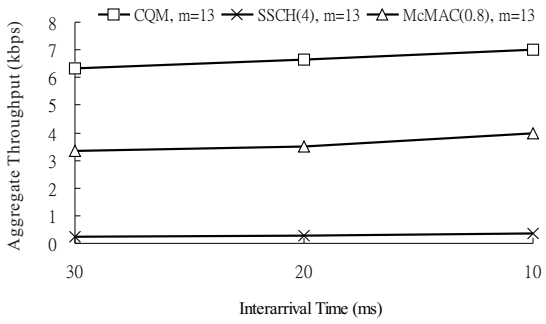

Fig. 15. Impact on packet interarrival time in real system implementation

We believe it is because the node density is reduced in the real system implementation. With less nodes, the throughput enhancement results from increased channels cannot compensate for the throughput degradation due to the missing receiver problem. Fig. 15 shows the results of different packet interarrival time with 30 nodes in the network. It is reasonable to find the throughput increases a little when traffic load becomes heavier. These real system implementation results verify the benefits of using CQM.

VI. Conclusions

In this paper, we proposed an efficient channel hopping MAC protocol for mobile ad hoc networks. The proposed CQM protocol is a multi-rendezvous one and requires only one transceiver for each node. Using an extension of the intersection property of the cyclic quorum systems, the separating of default/switching slots guarantees a sender running CQM to meet its intended receiver. CQM solves the missing receiver problem and the signaling overhead is very low since each node's hopping sequence is determined by its ID. Simulation and real system implementation results verified the superiority of CQM in that it achieves high throughput in different

scenarios. We believe that the proposed scheme achieves a great improvement over existing multi-channel MAC protocols.

REFERENCES

[1] CRCN: an ns-2 implementaton. *http://stuweb.ee.mtu.edu/~ljialian*.
[2] Hyacinth: an ns-2 implementaton. *http://www.ecsl.cs.sunysb.edu/multichannel/, http://www.cse.msu.edu/~wangbo1/ns2/nshowto8.html*.
[3] TeNS: an ns-2 implementaton. *http://www.cse.iitk.ac.in/users/braman/tens/*.
[4] IEEE 802.11 Working Group, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications". 1997.
[5] K. H. Almotairi and X. Shen. Fast and Slow Hopping MAC Protocol for Single-Hop Ad Hoc Wireless Networks. In *IEEE ICC*, pages 1–5, 2011.
[6] K. H. Almotairi and X. Shen. Multichannel medium access control for ad hoc wireless networks. 2011, http://dx.doi.org/10.1002/wcm.1159.
[7] P. Bahl, R. Chandra, and J. Dunagan. "SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks". In *ACM MOBICOM*, pages 216–230, Sep. 2004.
[8] P. Camarda and O. Fiume. "Collision Free MAC Protocol for Wireless Ad Hoc Networks based on BIBD Architecture". *Journal of Communications*, 2(7):1–8, Dec. 2007.
[9] C.-M. Chao and Y.-W. Lee. A quorum-based energy-saving MAC protocol design for wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 59:813–822, 2010.
[10] C.-M. Chao, J.-P. Sheu, and I.-C. Chou. "An Adaptive Quorum-Based Energy Conserving Protocol for IEEE 802.11 Ad Hoc Networks". *IEEE Transactions on Mobile Computing*, 5(5):560–570, Sept.-Oct. 2006.
[11] C.-M. Chao, H.-C. Tsai, and K.-J. Huang. A new channel hopping MAC protocol for mobile ad hoc networks. *Proc. Wireless Communications & Signal (WCSP)*, pages 1–5, 2009.
[12] D. N. M. Dang and C. S. Hong. H-MMAC: A Hybrid Multi-channel MAC Protocol for Wireless Ad hoc Networks. In *IEEE ICC*, 2012.
[13] L. Hongjiang, R. Zhi, G. Chao, and G. Yongcai. A New Multi-channel MAC Protocol for 802.11-based Wireless Mesh Networks. In *IEEE ICCSEE*, pages 27–31, 2012.
[14] F.-Y. Hung and I. Marsic. "Effectiveness of Physical and Virtual Carrier Sensing in IEEE 802.11 Wireless Ad Hoc Networks". In *IEEE WCNC*, pages 143–147, 2007.
[15] O. D. Incel, L. van Hoesel, P. Jansen, , and P. Havinga. MC-LMAC: A multi-channel MAC protocol for wireless sensor networks. *Ad Hoc Networks*, 9:73–94, Jan. 2011.
[16] 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(1-2):169–181, Feb. 2005.
[17] A. C. Khosrowshahi, M. Dehghan, H. Pedram, and M. Alizadeh. A Novel Multi Channel Sensor Network MAC Protocol. In *IEEE ICSNC*, pages 230-235, 2009.
[18] J.-H. Kim and S.-J. Yoo. TMCMP: TDMA based Multi-channel MAC Protocol for Improving Channel Efficiency in Wireless Ad Hoc Networks. In *IEEE MICC*, pages 429–434, Dec. 2009.
[19] P. Kyasanur and N. H. Vaidya. "Routing and Link-layer Protocols for Multi-Channel Multi-Interface Ad Hoc Wireless Networks". In *ACM SIGMOBILE Mobile Computing and Communications Review*, volume 10, pages 31–43, Jan. 2006.
[20] C.-S. Lin, M.-C. Wueng, T.-H. Chiu, and S.-I. Hwang. "Concurrent Multi-Channel Transmission (CMCT) MAC Protocol in Wireless Mobile Ad Hoc Networks". In *IEEE ICACT*, pages 445–449, Feb. 2007.
[21] W.-S. Luk and T.-T. Wong. "Two New Quorum Based Algorithms for Distributed Mutual Exclusion". In *IEEE ICDCS*, pages 100–106, May 1997.
[22] T. Luo, M. Motani, and V. Srinivasan. Cooperative Asynchronous Multichannel MAC: Design, Analysis, and Implementation. *IEEE Transactions on Mobile Computing*, 8(3):338–352, Mar. 2009.
[23] M. Maiya and B. Hamdaoui. An Improved IEEE 802.11 MAC Protocol for Wireless Ad-Hoc Networks with Multi-Channel Access Capabilities. In *IEEE HPCS*, 2011.
[24] J. S. Pathmasuntharam, A. Das, and A. K. Gupta. "Primary Channel Assignment based MAC (PCAM) - A Multi-Channel MAC Protocol for Multi-Hop Wireless Networks". In *IEEE WCNC*, pages 1110–1115, Vol. 2, Mar. 2004.
[25] M. Seo, Y. Kim, and J. Ma. Multi-channel MAC protocol for multi-hop wireless networks: Handling multi-channel hidden node problem using snooping. In *IEEE MILCOM*, 2008.
[26] J.-P. Sheu, C.-M. Chao, and C.-W. Sun. "A Clock Synchronization Algorithms for Multi-Hop Wireless Ad Hoc Networks". *Wireless Personal Communications*, 43(2):185–200, Oct. 2007.
[27] J. Shi, T. Salonidis, and E. W.Knightly. "Starvation Mitigation Through Multi-Channel Coordination in CSMA Multi-hop Wireless Networks". In *ACM MobiHoc*, pages 214–225, May 2006.
[28] H.-S. W. So, G. Nguyen, and J. Walrand. "Practical Synchronization Techniques for MultiChannel MAC". *ACM MOBICOM*, Sep. 2006.
[29] H.-S. W. SO, J. Walrand, and J. Mo. "McMAC: A Parallel Rendezvous Multi-Channel MAC Protocol". In *IEEE WCNC*, pages 334–339, 2007.
[30] J. So and N. Vaidya. "Multi-Channel MAC for Ad Hoc Networks: Handling Multi-Channel Hidden Terminals Using A Single Transceiver". In *ACM MobiHoc*, pages 222–233, May 2004.

[31] L. Tang, Y. Sun, O. Gurewitz, and D. B. Johnson. EM-MAC: A Dynamic Multichannel Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *ACM MobiHoc*, 2011.

[32] J. Wang, Y. Fang, and D. Wu. "A Power-Saving Multi-radio Multi-channel MAC Protocol for Wireless Local Area Networks". *IEEE INFOCOM*, pages 1–12, Apr. 2006.

[33] P.-J. Wu and C.-N. Lee. "On-Demand Connection-Oriented Multi-Channel MAC Protocol for Ad Hoc Network"'. In *IEEE SECON*, pages 621–625, Sept. 2006.

[34] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.-P. Sheu. "A Multi-Channel MAC Protocol with Power Control for Multi-Hop Mobile Ad Hoc Networks". *The Computer Journal*, 45(1):101–110, Jan. 2002.

[35] S.-L. Wu, C.-Y. Lin, Y.-C. Tseng, and J.-P. Sheu. "A New Multi-Channel MAC Protocol with On-Demand Channel Assignment for Multi-Hop Mobile Ad Hoc Networks". In *IEEE I-SPAN 2000*, pages 232–237, Dec. 2000.

[36] K. Xu, M. Gerla, and S. Bae. "How Effective is the IEEE 802.11 RTS/CTS Handshake in Ad Hoc Networks?". *IEEE GLOBECOM*, pages 72 – 76 vol.1, 2002.

[37] X. Zheng, L. Ge, and W. Guo. "A Load-balanced MAC Protocol for Multi-channel Ad-hoc Networks". In *IEEE ITST*, pages 642–645, June 2006.

[38] J. Zhu, X. Guo, L. L. Yang, and W. S. Conner. "Leveraging Spatial Reuse in 802.11 Mesh Networks with Enhanced Physical Carrier Sensing". In *IEEE ICC*, pages 4004–4011 Vol.7, 2004.