Novel Channel-Hopping Schemes for Cognitive Radio Networks

Guey-Yun Chang, Wen-Hung Teng, Hao-Yu Chen, and Jang-Ping Sheu, *Fellow*, *IEEE*

Abstract—Recently, cognitive radio (CR) has become a key technology for addressing spectrum scarcity. In CR networks, spectrum access should not interfere the colocate incumbent networks. Due to the requirement above, common control channel approaches, which are widely used in traditional multichannel environments, may face serious CR long-time blocking problem and control channel saturation problem. Although channel-hopping-based approaches can avoid these two problems, existing works still have significant drawbacks including long time-to-rendezvous, unbalance channel loading, and low channel utilization. In this paper, we introduce three channel-hopping approaches, RCCH, ARCH, and SARCH for synchronous and asynchronous environments, respectively. Compared with previous works, our schemes outperform the state of the art in terms of these metrics.

Index Terms—Cognitive radio, channel-hopping, medium access control, ad hoc networks

1 Introduction

Within the current spectrum regulatory framework, all of the frequency bands are exclusively allocated to specific services, and violation from unlicensed users is not allowed. The Federal Communications Commission (FCC) has indicated that the percentage of the assigned spectrum that is occupied only from 15 to 85 percent, varying widely in time and places. To address the critical problem of spectrum scarcity, the FCC has recently approved the use of unlicensed devices in licensed bands. This new field of research foresees the development of cognitive radio networks (CRNs) to further improve the spectrum efficiency [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26].

In cognitive radio networks, spectrum access should not interfere the incumbent network. The entities of an incumbent network are called *primary users* (PUs), while the entities of a CRN are called *secondary user*s (SUs). Each SU is equipped with one or more cognitive radios which are capable of identifying available channels (i.e., not occupied by PUs) and hopping between them. In addition, SUs should locate each other via a "rendezvous" process. In the process of rendezvous, SUs meet and establish a link (i.e., exchange control information) on an available channel, so that data communication can be carried on. However, implementation of rendezvous is challenging because SUs are not aware of the presence of each other before rendezvous and available channels sensed by each SU may be different (depends on the SU's location relative to PUs) [2].

Common control channel is probably the most well-known approach for rendezvous [3], [4], [5], [6], [7], [8], [9], [10], [11]. A dedicated channel is chosen to exchange control information as it is named. However, maintaining a common control channel in CRNs is not easy. The availability of dedicated control channel may change over time. Once a PU continuously occupies the common control channel for a long time, all of the control message exchange will be "blocked" during the long duration, called CR long-time blocking problem [13]. Although a new common control channel can be chosen and established according to channel availability [14], [15], updating channel availability information causes a considerable overhead. Moreover, a single control channel usually becomes a bottleneck and causes the control channel saturation problem in high-node-density or high-traffic-volume environments.

Channel-hopping (CH) is another representative technique for rendezvous [2], [12], [13], [16], [17], [18], [19], [20], [21], [22], [23]. Each SU hops on channels according to the sequence it generated for rendezvous with its potential neighbors. Once two SUs hop on the same available channel at the same time slot, they can exchange control messages. Furthermore, they can choose one of common available channels to start the data transmission.

Channel-hopping-based approaches can be classified into four categories: symmetric synchronous [13], [16], [21], [23], symmetric asynchronous [2], [17], [18], [19], [20], [21], [22], asymmetric synchronous, and asymmetric asynchronous [12], [21] (definitions of the four categories can be found in Section 2.1). In this paper, we consider the latter three categories because there exist many good symmetric synchronous schemes. We propose three channel-hopping schemes that can achieve small time-to-rendezvous, balance channel loading, and high channel utilization.

The rest of this paper is organized as follows: We provide the system model and formulate the problem in Section 2. In Section 3, we describe asymmetric and symmetric RCCH, ARCH, and SARCH systems, respectively. We analytically compare the proposed ACH system designs with existing CH schemes in Section 4. We evaluate

- *G.-Y. Chang, W.-H. Teng, and H.-Y. Chen are with the Department of Computer Science and Information Engineering, National Central University, Jhongli, Taiwan, R.O.C. E-mail: gychang@csie.ncu.edu.tw.*
- *J.-P. Sheu is with the Department of Computer Science, National Tsing Hua University, Hinchu, Taiwan, R.O.C.*

Manuscript received 5 Apr. 2012; revised 14 Aug. 2012; accepted 30 Nov. 2012; published online 20 Dec. 2012.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2012-04-0168. Digital Object Identifier no. 10.1109/TMC.2012.260.

our proposed schemes using simulation results in Section 5, and conclude the paper in Section 6.

2 Preliminary

2.1 System Model

In this paper, each SU is equipped with one half-duplex CR radio transceiver that can perform control message/data transmission and spectrum sensing. Spectrum sensing and channel-hopping-based protocols are distinct issues [25]. Spectrum sensing [26] aims to collect information about spectrum usage, and to maintain a dynamic picture of available channels, while channel-hopping-based protocols aim to design a channel-hopping sequence list that can maximize channel utilization. Our contribution is to increase channel utilization under the same wireless environment (e.g., the same spectrum sensing strategy, modulation and coding, power transmission, and antenna configuration).

Channel-hopping-based protocols can be classified according to two criteria, symmetric versus asymmetric, and synchronous versus asynchronous. Asymmetric approaches require role preassignment (i.e., every SU is preassigned a role as either a sender or a receiver, like the roles of a master and a slave in Bluetooth networks) prior to rendezvous, while symmetric approaches do not. Hence, in asymmetric approaches, SUs which play senders can use a method different from those of SUs that play receivers to generate their CH sequences, while in symmetric approaches, each SU follows the same method for generating its CH sequence. Synchronous approaches are developed under the assumption that there exist a mechanism to make all nodes in a CRN be able to periodically start hopping on CH sequences at the same global time, while asynchronous approaches can be used without the assumption.

Suppose there are N licensed channels in a CRN, labeled as $0, 1, \ldots, N-1$. To formulate the channel-hopping system, we assume that time is divided into multiple time slots, and m time slots make a CH period. A CH sequence determines the order with which a node visits all of the channels. We represent a CH sequence $\boldsymbol{u}$ of m time slots as a set of pairs (2-tuples):

$$\boldsymbol{u} = \{(0, u_0), (1, u_1), \ldots, (i, u_i), \ldots, (m-1, u_{m-1})\},$$

where $u_i \in [0, N-1]$ represents the channel which sequence $\boldsymbol{u}$ visits in the ith time slot of a CH period.

Given two CH sequences $\boldsymbol{u}$ and $\boldsymbol{v}$, if $(i, j) \in \boldsymbol{u} \cap \boldsymbol{v}$, (i, j) is called an *rendezvous* between $\boldsymbol{u}$ and $\boldsymbol{v}$. In this case, the ith time slot is called a *rendezvous slot* and channel j is called a *rendezvous channel* between $\boldsymbol{u}$ and $\boldsymbol{v}$. If two nodes select $\boldsymbol{u}$ and $\boldsymbol{v}$, respectively, as their CH sequences, then the resulting rendezvous channel can be used as a pairwise control channel, i.e., they can exchange control information on channel j at the ith time slot of every CH period.

2.2 Problem Statement

CH systems are commonly evaluated by the following four metrics [18], [19], [21]:

1. *Maximum time-to-rendezvous (MTTR)*. The maximum time for a pair of sequences in a CH system to rendezvous on an available channel when all channels are available (e.g., all SUs in a CR network are geographically close). Since the exchange of control information is not possible without rendezvous, to minimize the MTTR can also minimize the medium access delay of MAC protocols.
2. *Channel loading*. The maximum rendezvous proportion for a time slot. If SUs have a large proportion rendezvous on the same channel at the same time slot, the problem in common control channel approach may occur. A large channel loading leads to the control channel bottleneck problem and high probability of channel congestion. In synchronous approaches, channel loading is the maximum fraction of distinct CH sequences, which rendezvous on the same channel relative to all CH sequences. In asynchronous systems, due to the misalignment of starting time, channel loading of a CH system is the maximum fraction of incomplete-overlap CH sequences to rendezvous on the same channel among all the incomplete-overlap CH sequences. Two sequences are called *incomplete-overlap* if there exists one time slot that they stay at different channels. Notice that two identical sequences with distinct starting time may become incomplete-overlap. For example, DRSEQ [22] has channel loading $= 2/(2N+1)$. This is because DRSEQ [22] has $2N+1$ incomplete-overlap CH sequences (due to the misalignment of starting time), although all SUs use the same CH sequence.
3. *Degree of rendezvous*. The minimum number of rendezvous channels which a pair of CH sequences could have. Define $C(\boldsymbol{u}, \boldsymbol{v})$ to be the set of rendezvous channels of sequence $\boldsymbol{u}$ and $\boldsymbol{v}$, and define $|C(\boldsymbol{u}, \boldsymbol{v})|$ to be the number of rendezvous channels of $\boldsymbol{u}$ and $\boldsymbol{v}$. Formally, degree of rendezvous denotes the minimum $|C(\boldsymbol{u}, \boldsymbol{v})|$ among all sequences $\boldsymbol{u}$ and $\boldsymbol{v}$ in a CH system. A CH system with N degree of rendezvous guarantees that any two sequences can rendezvous on all of N channels. That is, any two sequences can exchange control message successfully if there are at least one available channel for any two SUs. Maximizing the degree of rendezvous can reduce the impact of PU long time blocking problem and maximize the rendezvous probability in a high traffic CRNs.
4. *Maximum conditional time-to-rendezvous (MCTTR)*. The maximum time for two CH sequences to rendezvous on an available channel when not all channels are ensured to be available to every SU. A CH scheme with MCTTR $= X$ means that if there is at least one available channel for any two SUs, then any pair of sequences could be able to exchange control message successfully within X time slots.

In this paper, we introduce three channel-hopping schemes, RCCH, ARCH, and SARCH, for asymmetric synchronous, asymmetric asynchronous, and symmetric asynchronous environments, respectively. RCCH has optimal MTTR $= N/2$, nearly optimal channel loading $= 2/N$, optimal degree of rendezvous $= N$, and MCTTR $= N^2/2$. ARCH has optimal channel loading $= 1/N$, optimal degree

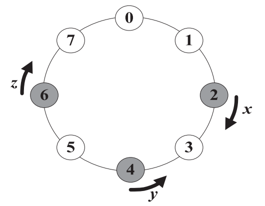

Fig. 1. Sequences x, y, and z.

of rendezvous $= N$, optimal MCTTR $= N^2$, and a significant improvement on MTTR $(= 2N - 1)$. SARCH has MTTR $< 4N + 2$, nearly optimal channel loading $= 3/(2N + 1)$, and MCTTR $= 8N^2 + 8N$.

3 Channel-Hopping System

3.1 Main Idea

Our main idea comes from the observation: Suppose that nodes x and z move clockwise, and node y moves counterclockwise, along a ring with the same speed. Then, both x and z can rendezvous with y twice per round, but x and z never rendezvous.

Consider a discrete instance. There are N channels in a round, each sequence hops k channels per time slot (i.e., CH seed is k). Assume that sequences x and z have clockwise CH direction, and sequence y has counterclockwise CH direction. Does x rendezvous with y twice per round? Do x and z never rendezvous with each other? Before answering these questions, let us see the example shown in Fig. 1, where $N = 8$, CH seed $k = 3$, x, y, and z have initial channels 2, 4, and 6, respectively. That is, $x = \{(0,2), (1,5), (2,0), (3,3), (4,6), (5,1), (6,4), (7,7)\}$, $y = \{(0,4), (1,1), (2,6), (3,3), (4,0), (5,5), (6,2), (7,7)\}$, and $z = \{(0,6), (1,1), (2,4), (3,7), (4,2), (5,5), (6,0), (7,3)\}$. Clearly, x and y can rendezvous on channel 3 and channel 7, but x and z do not. In the example, we observe that x and y can rendezvous when N is even, N and CH seed k are coprime, and initial channels of x and y have the same parity (i.e., both are odd, or both are even). For ease of the following discussion, channels or time slots are called *even* if they are indexed by even numbers (e.g., channel 2), and *odd* otherwise. In the rest of this paper, time slots are indexed according to a global (universal) time slot system.

Definition 1. *A CH system with N channels and CH seed k is called potential if the following conditions hold:*

1. *N is even.*
2. *N and CH seed k are coprime.*

Definition 2. *Two CH sequences x and y are called a potential rendezvous couple (PRC) if the following three conditions are satisfied:*

1. *Initial channels of x and y have the same parity.*
2. *x and y have distinct CH directions.*
3. *x and y are in a potential CH system.*

To have optimal degree of rendezvous (i.e., there are N distinct rendezvous channels for a CH sequence pair), each sequence should visit N distinct channels in a CH period. Lemma 1 shows that sequences in a potential CH system have such a property.

Lemma 1. *Suppose that x is a CH sequence in a potential CH system with N channels. Then, x can visit each of the N channels within N time slots.*

Proof. Suppose to the contrary that sequence x does not visit channel c. Since there are totally N channels, by pigeonhole principle, there exist a channel c' such that x visits channel c' at least twice within N time slots. Without loss of generality, assume that x visits channel c' at time slots j_1 and j_2, and x has initial channel a and CH seed k. If x has clockwise CH direction, then $c' = (a + j_1 \times k) \bmod N = (a + j_2 \times k) \bmod N$. That is, $0 = (j_1 - j_2)k \bmod N$. Since k and N is coprime and $j_1 - j_2 < N$, we have $0 = j_1 - j_2$, a contradiction. Otherwise (x has counterclockwise CH direction), $c' = (a - j_1 \times k) \bmod N = (a - j_2 \times k) \bmod N$, which also implies $0 = j_1 - j_2$, a contradiction. □

Next, we show that a PRC have MTTR $= N/2$ and rendezvous every $N/2$ time slots.

Lemma 2. *Suppose that x and y are a PRC, and have initial channels a and b, respectively. Then, the following two statements hold:*

1. *x and y rendezvous twice; one at channel $(a + b)/2$ and the other at channel $(a + b + N)/2$ mod N.*
2. *x and y rendezvous every $N/2$ time slots.*

Proof. Suppose that x and y rendezvous at time slot $i (< N)$, and have CH directions clockwise and counterclockwise, respectively. Clearly, at time slot i, x visits channel $(a + i \times k) \bmod N$ and y visits channel $(b - i \times k) \bmod N$. Hence, $(a + i \times k) \bmod N = (b - i \times k) \bmod N$ or $a + i \times k = b - i \times k + m \times N$, which implies that

$$i \times k \mod N = (b - a + m \times N)/2 \mod N.$$

That is,

$$i \times k \mod N = (b - a + N)/2 \mod N \tag{1}$$

or

$$i \times k \mod N = (b - a + 2N)/2 \mod N. \tag{2}$$

According to (1) and (2), we have that $(a + i \times k) \mod N = (a + b + N)/2$ or $(a + b)/2 \mod N$.

Next, we prove that x and y rendezvous every $N/2$ time slots. It is sufficient to prove that (1) holds at a certain time slot $i^* \in [0, N - 1]$, and (2) holds at time slot $i^* + N/2$ (if $i^* < N/2$) or $i^* - N/2$ (if $i^* \geq N/2$). Let z be a CH sequence in the same potential CH system, and have initial channel 0, i.e., at time slot i, z visits channel $i \times k \mod N$. According to Lemma 1, there exist $i^* \in [0, N - 1]$ such that

$$i \times k \mod N = (b - a + N)/2 \mod N \text{ when } i = i^*,$$

i.e., (1) holds. Then, it is not difficult to see that (2) holds, i.e., $i \times k \mod N = (b - a + 2N)/2 \mod N$, when $i = i^* + N/2$ (if $i^* < N/2$) or $i = i^* - N/2$ (if $i^* \geq N/2$). So, the theorem follows. □

Notice that MTTR $= N/2$ is optimal. For each sequence $\boldsymbol{x}$, there are $N/2$ sequences that can rendezvous with $\boldsymbol{x}$ (since there are $N/2$ sequences each of whose union with $\boldsymbol{x}$ can form a PRC). To rendezvous with each of these $N/2$ sequences once at distinct time slots, MTTR should be at least $N/2$.

Define $rotate(\boldsymbol{x}, i)$ to be the CH sequence which is i time slots ahead from sequence $\boldsymbol{x}$. For example, if $\boldsymbol{x} = \{(0,0), (1,1), (2,2), (3,3), (4,4), (5,5)\}$, then $rotate(\boldsymbol{x}, 2) = \{(0,2), (1,3), (2,4), (3,5), (4,0), (5,1)\}$. Formally, if sequence $\boldsymbol{x}$ is clockwise with initial channel a and CH seed k, i.e., $\boldsymbol{x} = \{(0,a), (1, (a+k) \bmod N), \ldots, (N-1, (a+(N-1)k) \bmod N)\}$, then $rotate(\boldsymbol{x}, i) = \{(0, (a+ik) \bmod N), (1, (a+(i+1)k) \bmod N), \ldots, (N-1, (a+(i+N-1)k) \bmod N)\}$; if $\boldsymbol{x}$ is counterclockwise, i.e., $\boldsymbol{x} = \{(0,a), (1, (a-k) \bmod N), \ldots, (N-1, (a-(N-1)k) \bmod N)\}$, then $rotate(\boldsymbol{x}, i) = \{(0, (a-ik) \bmod N), (1, (a-(i+1)k) \bmod N), \ldots, (N-1, (a-(i+N-1)k) \bmod N)\}$.

Clearly, $\boldsymbol{x}$ and $rotate(\boldsymbol{x}, 2i)$ have the same initial channel parity, the same CH direction, and the same CH seed. If $\boldsymbol{x}$ and $\boldsymbol{y}$ are a PRC, then $rotate(\boldsymbol{x}, 2i)$ and $\boldsymbol{y}$ are a PRC (refer to Definition 2), also.

Lemma 3. *Suppose that $\boldsymbol{x}$ and $\boldsymbol{y}$ are a PRC. Then, $rotate(\boldsymbol{x}, 2i)$ and $\boldsymbol{y}$ are also a PRC, for $1 \leq i \leq N/2 - 1$.*

Keep in mind that sequences $\boldsymbol{x}$ and $\boldsymbol{z} = rotate(\boldsymbol{x}, j)$ never rendezvous for all integer j. Hence, the rendezvous (time slot, channel) pairs of $\boldsymbol{x}$ and $\boldsymbol{y}$ are different from those of $\boldsymbol{z}$ and $\boldsymbol{y}$.

Lemma 4. *Suppose that $\boldsymbol{x}$ and $\boldsymbol{y}$ are a PRC, and $\boldsymbol{z} = rotate(\boldsymbol{x}, j) \neq \boldsymbol{x}$. Then, $C(\boldsymbol{x}, \boldsymbol{y}) \cap C(\boldsymbol{z}, \boldsymbol{y}) = \phi$.*

3.2 The Proposed Asymmetric Synchronous Approach

In this section, we propose rendezvous couple channel-hopping scheme (*RCCH*) for asymmetric synchronous environments. RCCH guarantees that CH sequences of each pair of sender and receiver is a PRC. According to Definition 2, sequences in a PRC should have initial channels with the same parity and should be in a potential CH system. Below, without loss of the generality, we assume that CH sequences have even initial channels and are in a potential CH system:

- *Alternative CH sequence $\boldsymbol{u}$.* If a node is a sender that has data to transmit, it randomly selects an even initial channel a, and constructs a counterclockwise sequence $\boldsymbol{x} = \{(0,a), (1, (a-k) \bmod N), \ldots, (i, (a-ik) \bmod N), \ldots, (N-1, (a-(N-1)k) \bmod N)\}$ by the aid of default CH seed k. Then, it constructs its alternative CH sequence $\boldsymbol{u}$, by concatenating the following $N/2$ CH sequences: $rotate(\boldsymbol{x}, 0)$, $rotate(\boldsymbol{x}, 2)$, $\ldots$, $rotate(\boldsymbol{x}, 2i), \ldots, rotate(\boldsymbol{x}, N-2)$. Fig. 2a shows all alternative CH sequences when $N = 6$ and $k = 1$.
- *Default CH sequence $\boldsymbol{v}$.* If a node has nothing to send, it randomly selects an even initial channel b, and constructs a clockwise sequence $\boldsymbol{y} = \{(0,b), (1, (b+k) \bmod N), \ldots, (i, (b+ik) \bmod N), \ldots, (N-1, (b+(N-1)k) \bmod N)\}$. The node constructs its default CH sequence, $\boldsymbol{v}$, by repeating $\boldsymbol{y}$ $N/2$ times and concatenating them into a single sequence. Fig. 2b shows all default CH sequences when $N = 6$ and $k = 1$.

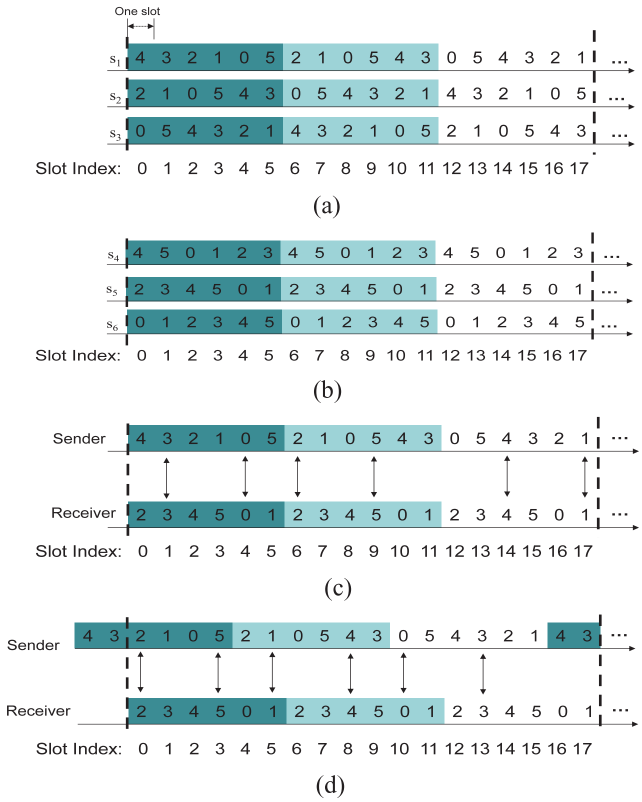

Fig. 2. An RCCH system. (a) All alternative sequences when $N = 6$ and $k = 1$. (b) All default sequences when $N = 6$ and $k = 1$. (c) Rendezvous in a synchronous scenario. (d) Rendezvous in an asynchronous scenario. Sender's clock is two time slots ahead of receiver's.

In Fig. 2a, each alternative sequence $\boldsymbol{u}$ can be obtained by rotating another alternative sequence $\boldsymbol{w}$. For example, alternative sequence s_2 can be obtained either by rotating alternative sequence s_1 $N = 6$ time slots, or by rotating alternative sequence s_3 $2N$ time slots. Similarly, in Fig. 2b, default sequence s_5 can be obtained either by rotating default sequence s_4 four time slots, or by rotating default sequence s_6 two time slots. Since N is even, we have Lemma 5.

Lemma 5. *Given two sequences $\boldsymbol{u}$ and $\boldsymbol{w}$. If both of them are alternative sequences or default sequences, then there exists some integer i such that $\boldsymbol{w} = rotate(\boldsymbol{u}, 2i)$.*

Proof. The proof is easy to obtain and we omit it here. □

Refer to Fig. 2c. An alternative CH sequence $\boldsymbol{u}$ (of sender) and a default CH sequence $\boldsymbol{v}$ (of receiver) can rendezvous every $N/2$ time slots. Besides, $\boldsymbol{u}$ and $\boldsymbol{v}$ have N distinct rendezvous channels, i.e., $C(\boldsymbol{u}, \boldsymbol{v}) = N$.

Theorem 6. *For an alternative sequence $\boldsymbol{u}$ and a default sequence $\boldsymbol{v}$, they have $C(\boldsymbol{u}, \boldsymbol{v}) = N$.*

Proof. Suppose that $\boldsymbol{u} = \{rotate(\boldsymbol{x}, 0), rotate(\boldsymbol{x}, 2), \ldots, rotate(\boldsymbol{x},' 2i), \ldots, rotate(\boldsymbol{x}, N-2)\}$ and $\boldsymbol{v} = \{\boldsymbol{y}, \ldots, \boldsymbol{y}\}$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ have even initial channels, the same CH seed, but distinct CH directions. According to Definition 2, $\boldsymbol{x}$ and $\boldsymbol{y}$ are a PRC. By Lemma 2 and Lemma 3,

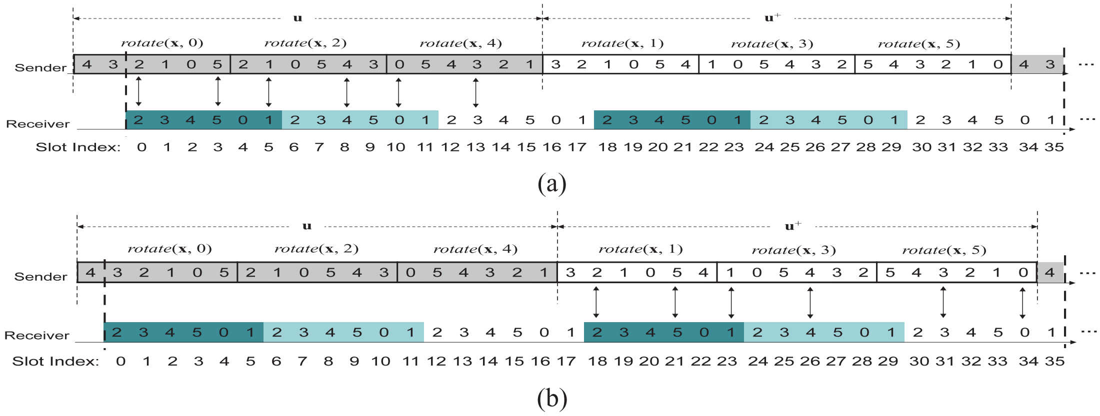

Fig. 3. Rendezvous in an asynchronous scenario. (a) Sequences of sender and receiver have the same time parity. (b) Sequences of sender and receiver do not have the same time parity.

$rotate(\boldsymbol{x}, 2i)$ rendezvous with $\boldsymbol{y}$ twice for all i. According to Lemma 4, rendezvous channels of $rotate(\boldsymbol{x}, 2i)$ and $\boldsymbol{y}$ are completely distinct from those of $rotate(\boldsymbol{x}, 2j)$ and $\boldsymbol{y}$ if $rotate(\boldsymbol{x}, 2i) \neq rotate(\boldsymbol{x}, 2j)$. Thus, $\boldsymbol{u}$ and $\boldsymbol{v}$ have N rendezvous channels, i.e., $C(\boldsymbol{u}, \boldsymbol{v}) = 2 \times N/2 = N$ (since there are $N/2$ $rotate(\boldsymbol{x}, 2i)$s included in $\boldsymbol{u}$). □

RCCH is also applicable to some asynchronous scenarios. In Fig. 2d, sender generates alternative sequence s_1 (see Fig. 2a) and receiver generates alternative sequence s_5 (see Fig. 2b). They can rendezvous with each other because they satisfy one of the following two conditions: 1) both of them visit even channels at even time slots and visit odd channels at odd time slots; 2) both of them visit even channels at odd time slots and visit odd channels at even time slots. For ease of discussion, we call a sequence to be *even-time-parity* if it visits even channels at even time slots (in the global time slot system) and odd channels at odd time slots (in the global time slot system). Similarly, we call a sequence to be *odd-time-parity* if it visits even channels at odd time slots and odd channels at even time slots. Two sequences are called to have the same time parity if both of them are even-time-parity or odd-time-parity.

3.2.1 Evaluation Metrics and RCCH

MTTR of RCCH. According to Lemma 2, sender and receiver rendezvous every $N/2$ time slots, and hence RCCH has optimal MTTR $N/2$. MTTR $= N/2$ is optimal because an alternative sequence has to rendezvous with $N/2$ distinct default sequences at distinct time slots.

Channel loading of RCCH. Refer to Figs. 2a and 2b. There are totally N CH sequences (i.e., $N/2$ alternative sequences and $N/2$ default sequences) in RCCH. Given a time slot t and a channel c. There are two sequences including (t, c). For example, (0, 2) is included in alternative sequence s_2 and default sequence s_5. Hence, the loading of RCCH is $2/N$. This result is proved below.

Theorem 7. *Channel loading of RCCH is* $2/N$.

Proof. Suppose an alternative sequences u and a default sequence v rendezvous at channel c and time slot t. Assume sequence $w \notin \{u, v\}$ also visits channel c at time slot t. If w is an alternative sequence, then $(t, c) \in C(u, v) \cap C(w, v)$ and $w = rotate(u, 2i)$ for some i (by Lemma 5). According to Lemma 4, $C(u, v) \cap C(w, v) = C(u, v) \cap C(rotate(u, 2i), v) = \phi$, a contradiction. Otherwise w is a default sequence, i.e., $(t, c) \in C(u, v) \cap C(u, w)$. Again, according to Lemma 4, $C(u, v) \cap C(u, w) = C(u, v) \cap C(u, rotate(v, 2j)) = \phi$, a contradiction also. □

Degree of rendezvous of RCCH. According to Theorem 6, RCCH has optimal degree of rendezvous N.

MCTTR of RCCH. MCTTR denotes the maximum time-to-rendezvous at an incumbent-free channel between two SUs when incumbent traffic is present and at least one incumbent-free channel is available for them. When there is only one channel that is free of incumbent signals for the two SUs, their CH sequences picked from a CH system have to rendezvous on every channel to guarantee that they can find the incumbent-free channel for rendezvous. Since RCCH has CH period $= N^2/2$ time slots and degree of rendezvous $= N$, RCCH has MCTTR $= N^2/2$.

3.3 The Proposed Asymmetric Asynchronous Approach

In this section, we introduce asynchronous rendezvous channel-hopping scheme (*ARCH*) for asymmetric asynchronous environments. A CH scheme is designed for *asynchronous* environments if it is applicable to the scenario that CH period boundaries are misaligned, but time slot boundaries are aligned [21]. In other words, SUs may start their CH sequences at distinct time slots in asynchronous environments.

Without loss of generality, assume a sender with an alternative sequence $\boldsymbol{u}$ (generated by concatenating $rotate(\boldsymbol{x}, 0), rotate(\boldsymbol{x}, 2), \ldots, rotate(\boldsymbol{x}, 2i), \ldots, rotate(\boldsymbol{x}, N-2)$) wants to exchange data with a SU whose sequence has even-time-parity, where $\boldsymbol{x} = \{(0, a), (1, (a-k) \bmod N), \ldots, (N-1, (a-(N-1)k) \bmod N)\}$. If sender's sequence is also even-time-parity, then sender and receiver can rendezvous as described in RCCH. However, if sender's sequence is odd-time-parity, then no rendezvous occurs. In Fig. 3b, $\boldsymbol{u}$ does not rendezvous with receiver's sequence. Since sender has no information about receiver's time parity, to guarantee rendezvous, a feasible solution is to include one even-time-parity sequence and one odd-time-parity sequence in sender's sequence. In Fig. 3a, the sender sequence concatenate $\boldsymbol{u}$ and $\boldsymbol{u}^+$, where $\boldsymbol{u}^+$ is the sequence generated by

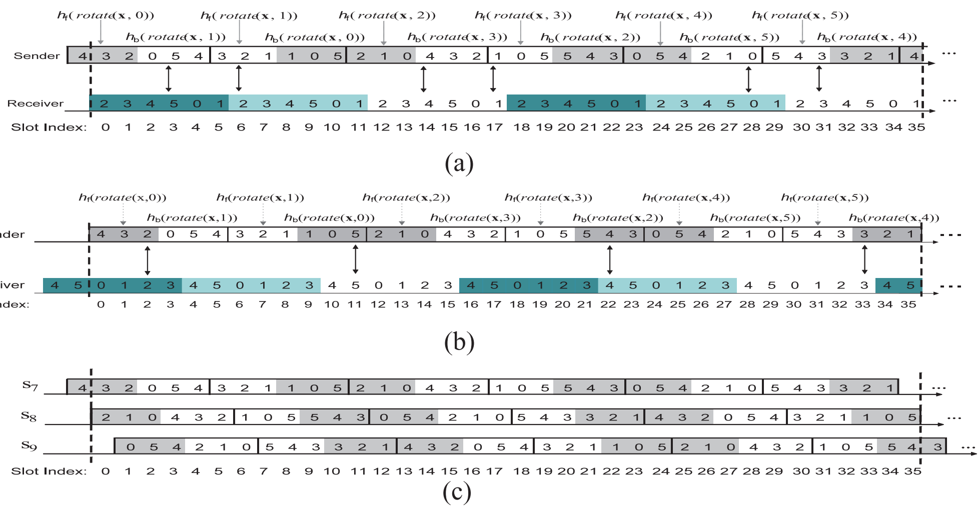

Fig. 4. Rendezvous in ARCH. (a) Sequences of sender and receiver have the same time parity. (b) Sequences of sender and receiver do not have the same time parity. (3) Three a-alternative sequences.

concatenating $rotate(\boldsymbol{x},1), rotate(\boldsymbol{x},3), \ldots, rotate(\boldsymbol{x},2i+1), \ldots,$ and $rotate(\boldsymbol{x}, N-1)$). Obviously, $\boldsymbol{u}$ and $\boldsymbol{u}^+$ have distinct time parity (channel 0 is visited at even time slots in $\boldsymbol{u}$, but channel 0 is visited at odd time slots in $\boldsymbol{u}^+$). Summarizing the discussion above, if sender's sequence including both $\boldsymbol{u}$ and $\boldsymbol{u}^+$, then rendezvous can be guaranteed (refer to Fig. 3). However, MTTR is too long. In Fig. 3b, sender and receiver take $N^2/2$ time slots to rendezvous. Below we aim to design an asymmetric asynchronous approach with MTTR $= 2N-1$ by rearranging subsequences of $\boldsymbol{u}$ and $\boldsymbol{u}^+$.

For ease of the discussion, let $h_{\mathrm{f}}(^{**})$ denote the front half of sequence ** and $h_{\mathrm{b}}(^{**})$ denote the back half. Define an *even-rotated* half-sequence to be a $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 2i))$ or a $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 2i))$ for some integer i, and define an *odd-rotated* half-sequence to be a $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 2j+1))$ or a $h_{\mathrm{b}}(rotate(\boldsymbol{x},\ 2j+1))$.

In Fig. 3a, sequences of sender and receiver have the same time parity, and rendezvous when sender executes one of the following N even-rotated half-sequences: $h_{\mathrm{f}}(rotate(\boldsymbol{x},0)), h_{\mathrm{b}}(rotate(\boldsymbol{x},0)), h_{\mathrm{f}}(rotate(\boldsymbol{x},2)), h_{\mathrm{b}}(rotate(\boldsymbol{x},2)), \ldots, h_{\mathrm{f}}(rotate(\boldsymbol{x}, N-2))$, and $h_{\mathrm{b}}(rotate(\boldsymbol{x}, N-2))$. Although, in Fig. 3b, sequences of sender and receiver do not have the same time parity, they rendezvous when sender executes one of the following N *odd-rotated* half-sequences: $h_{\mathrm{f}}(rotate(\boldsymbol{x},1)), h_{\mathrm{b}}(rotate(\boldsymbol{x},1)), h_{\mathrm{f}}(rotate(\boldsymbol{x},3)), h_{\mathrm{b}}(rotate(\boldsymbol{x},3)), \ldots, h_{\mathrm{f}}(rotate(\boldsymbol{x}, N-1))$, and $h_{\mathrm{b}}(rotate(\boldsymbol{x},\ N-1))$.

Below, we define sequences of senders and receivers for N^2 time slots:

- *A-alternative CH sequence* $\boldsymbol{u}_A$. A sender randomly selects an even initial channel a, and constructs a counterclockwise sequence $\boldsymbol{x} = \{(0,a), (1,(a-k) \bmod N), \ldots, (i,(a-ik) \bmod N), \ldots, (N-1, (a-(N-1)k) \bmod N)\}$. Then, the sender constructs it's a-alternative CH sequence u_{A}, by concatenating the following $2N$ half-sequences: $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 0))$, $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 1))$, $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 1))$, $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 0)), \ldots,$ $h_{\mathrm{f}}(rotate(\boldsymbol{x}, N-2)), h_{\mathrm{b}}(rotate(\boldsymbol{x}, N-1)), h_{\mathrm{f}}(rotate(\boldsymbol{x}, N-1))$, and $h_{\mathrm{b}}(rotate(\boldsymbol{x}, N-2))$.
- *A-default CH sequence* v_A. If a SU has nothing to send, it randomly selects an even initial channel b, and constructs a clockwise sequence $\boldsymbol{y} = \{(0,b), (1,(b+k) \bmod N), \ldots, (i,(b+ik) \bmod N), \ldots, (N-1, (b+(N-1)k) \bmod N)\}$. The node constructs it's a-default CH sequence, $\boldsymbol{v}_{\mathrm{A}}$, by repeating $\boldsymbol{y}$ N times and concatenating them into a single sequence.

3.3.1 Evaluation Metrics and ARCH

Theorem 8. *ARCH has MTTR* $= 2N-1$.

Proof. Consider the case that sequences of sender and receiver have distinct time parity. Clearly, they can rendezvous when sender executes an odd-rotated half-sequences (i.e., $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 2i+1))$ or $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 2i+1))$), see Figs. 3b and 4a. Consider the other case that sequences of sender and receiver have the same time parity. They can rendezvous when sender executes an even-rotated half-sequences (i.e., $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 2i))$ or $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 2i))$), see Figs. 3a and 4b. Keep in mind that within $2N-1$ time slots, sender's sequence contains at least one odd-rotated half-sequence and one even-rotated half-sequence. For example, in Fig. 4a, sender's sequence contains two odd-rotated half-sequences, $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 1))$ and $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 1))$, and one even-rotated half-sequence, $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 0))$, during time slots 0 to 10. Hence, ARCH has MTTR $= 2N-1$. □

Theorem 9. *ARCH has optimal degree of rendezvous* $= N$.

Proof. Consider the case that sender's a-alternative sequence $\boldsymbol{u}_{\mathrm{A}}$ and receiver's a-default sequence $\boldsymbol{v}_{\mathrm{A}}$ have the same time parity. Since both $\boldsymbol{u}_{\mathrm{A}}$ and $\boldsymbol{v}_{\mathrm{A}}$ have even initial channels, without loss of generality, assume that $\boldsymbol{u}_{\mathrm{A}}$ is executed $4k$ time slots later than $\boldsymbol{v}_{\mathrm{A}}$ (if $\boldsymbol{u}_{\mathrm{A}}$ and $\boldsymbol{v}_{\mathrm{A}}$ have distinct time parity, then $\boldsymbol{u}_{\mathrm{A}}$ must be executed $2k'+1$ time slots later than $\boldsymbol{v}_{\mathrm{A}}$ for some k'). Then, rendezvous channels of the sender and its receiver are the same with that of $\boldsymbol{u}_{\mathrm{A}}$ and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2k \bmod N)$ in a synchronous environment. For example, in Fig. 4b, $\boldsymbol{u}_{\mathrm{A}}$ is executed two time slots later than $\boldsymbol{v}_{\mathrm{A}}$. So, the rendezvous channels of the sender and its receiver are the same with that of $\boldsymbol{u}_{\mathrm{A}}$ and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2) = \{(0,0), (1,1), (2,2), (3,3), (4,4), (5,5), \ldots\}$ when $\boldsymbol{u}_{\mathrm{A}}$ and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2)$ are initially executed at the same time slot. In this case, it is sufficient to prove that $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 2i))$ or $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 2\boldsymbol{i}))$ in $\boldsymbol{u}_{\mathrm{A}}$ can rendezvous with $rotate(\boldsymbol{y}, 2k)$ in $rotate(\boldsymbol{v}_{\mathrm{A}}, 2k)$. Notice that sequences $\boldsymbol{x}$ and $\boldsymbol{y}$ in ARCH is a PRC. According to Lemma 2 and Lemma 3, $rotate(\boldsymbol{x}, 2i)$ rendezvous with $rotate(\boldsymbol{y}, 2k)$ every $N/2$ time slots at distinct channels for all i. That is, $h_{\mathrm{f}}(rotate(\boldsymbol{x}, 2i))$ can rendezvous with $rotate(\boldsymbol{y}, 2k)$ at one channel, and $h_{\mathrm{b}}(rotate(\boldsymbol{x}, 2i))$ can rendezvous with $rotate(\boldsymbol{y}, 2k)$ at another channel. According to Lemma 4, rendezvous channels of $rotate(\boldsymbol{x}, 2i)$ and $rotate(\boldsymbol{y}, 2k)$ are completely distinct from those of $rotate(\boldsymbol{x}, 2j)$ and $rotate(\boldsymbol{y}, 2k)$ if $rotate(\boldsymbol{x}, 2i) \neq rotate(\boldsymbol{x},\ 2j)$. Since there are $N/2$ distinct $rotate(\boldsymbol{x}, 2i)$s in $\boldsymbol{u}_{\mathrm{A}}$, $\boldsymbol{u}_{\mathrm{A}}$ and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2k)$ have $N = 2 \times N/2$ rendezvous channels. The proof of the case that $\boldsymbol{u}_{\mathrm{A}}$ and $\boldsymbol{v}_{\mathrm{A}}$ have distinct time parity is very similar. Due to the limitation of space, we omit it here.□

In Fig. 4c, each a-alternative sequence $\boldsymbol{u}_{\mathrm{A}}$ can be obtained by rotating another a-alternative sequence $\boldsymbol{w}_{\mathrm{A}}$. For example, a-alternative sequence s_8 can be obtained either by rotating a-alternative sequence s_7 11 time slots, or by rotating a-alternative sequence s_9 25 time slots. Similarly, a a-default sequence can be obtained by rotating another a-default sequence.

Theorem 10. *Given two sequences $\boldsymbol{u}_{\mathrm{A}}$ and $\boldsymbol{w}_{\mathrm{A}}$. In synchronous environments, if both of them are a-alternative sequences or a-default sequences, then there exists an integer i such that $\boldsymbol{w}_{\mathrm{A}} = rotate(\boldsymbol{u}_{\mathrm{A}}, 2i)$.*

Proof. Consider the case that both $\boldsymbol{u}_{\mathrm{A}}$ and $\boldsymbol{w}_{\mathrm{A}}$ are a-alternative sequences. Without loss of generality, assume that $\boldsymbol{u}_{\mathrm{A}}$ is the sequence concatenating $h_{\mathrm{f}}(rotate(\boldsymbol{x}, ^*))$s and $h_{\mathrm{b}}(rotate(\boldsymbol{x}, ^*))$s, and $\boldsymbol{w}_{\mathrm{A}}$ is the sequence concatenating $h_{\mathrm{f}}(rotate(\boldsymbol{x}', ^*))$s and $h_{\mathrm{b}}(rotate(\boldsymbol{x}', ^*))$s, where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are both counterclockwise sequences with default CH seed k. Since $\boldsymbol{x}$ and $\boldsymbol{x}'$ have the same CH seed k and even initial channels, there exist an integer i such that $\boldsymbol{x}' = rotate(\boldsymbol{x}, 2i)$. Hence, $h_{\mathrm{f}}(rotate(\boldsymbol{x}', ^*)) = h_{\mathrm{f}}(rotate(rotate(\boldsymbol{x}, 2i), ^*)) = h_{\mathrm{f}}(rotate(rotate(\boldsymbol{x}, ^*), 2i))$ and $h_{\mathrm{b}}(rotate(\boldsymbol{x}', ^*)) = h_{\mathrm{b}}(rotate(rotate(\boldsymbol{x}, 2i), ^*)) = h_{\mathrm{b}}(rotate(rotate(\boldsymbol{x}, ^*), 2i))$, which implies that $\boldsymbol{w}_{\mathrm{A}} = rotate(\boldsymbol{u}_{\mathrm{A}}, 2i)$. The proof of the case that both $\boldsymbol{u}_{\mathrm{A}}$ and $\boldsymbol{w}_{\mathrm{A}}$ are a-default sequences is easier, and we omit it here. □

Since degree of rendezvous $= N$ and a CH period $= N^2$ time slots, MCTTR of ARCH is N^2. Below, we show ARCH has optimal channel loading $1/N$.

Theorem 11. *Channel loading of ARCH is $1/N$.*

Proof. According to Theorem 10 and the definition of a-default/a-alternative sequences, there are $N^2/2$ distinct a-alternative sequences (i.e., $rotate(\boldsymbol{u}_{\mathrm{A}}, 2i)$s) and $N/2$ distinct a-default sequences (i.e., $rotate(\boldsymbol{v}_{\mathrm{A}}, 2j)$s) in synchronous environments. However, due to the misalignment of CH period boundaries in asynchronous environments, $rotate(\boldsymbol{u}_{\mathrm{A}}, 2i)$s and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2j)$s may become $rotate(\boldsymbol{u}_{\mathrm{A}}, 2i' + 1)$s and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2j' + 1)$s, respectively. Hence, the rendezvous of sequences in ARCH is the same as that of the following synchronous system (called S): N^2 distinct a-alternative sequences (i.e., $rotate(\boldsymbol{u}_{\mathrm{A}}, 2i)$s and $rotate(\boldsymbol{u}_{\mathrm{A}}, 2i + 1)$s) and N

timeslots	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
rotate($\mathbf{u}_{\mathrm{A}}$, 0)	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1
rotate($\mathbf{u}_{\mathrm{A}}$, 1)	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4
rotate($\mathbf{u}_{\mathrm{A}}$, 2)	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3
rotate($\mathbf{u}_{\mathrm{A}}$, 3)	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2
rotate($\mathbf{u}_{\mathrm{A}}$, 4)	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0
rotate($\mathbf{u}_{\mathrm{A}}$, 5)	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5
rotate($\mathbf{u}_{\mathrm{A}}$, 6)	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4
rotate($\mathbf{u}_{\mathrm{A}}$, 7)	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3
rotate($\mathbf{u}_{\mathrm{A}}$, 8)	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2
rotate($\mathbf{u}_{\mathrm{A}}$, 9)	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1
rotate($\mathbf{u}_{\mathrm{A}}$, 10)	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1
rotate($\mathbf{u}_{\mathrm{A}}$, 11)	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0
rotate($\mathbf{u}_{\mathrm{A}}$, 12)	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5
rotate($\mathbf{u}_{\mathrm{A}}$, 13)	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2
rotate($\mathbf{u}_{\mathrm{A}}$, 14)	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1
rotate($\mathbf{u}_{\mathrm{A}}$, 15)	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0
rotate($\mathbf{u}_{\mathrm{A}}$, 16)	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4
rotate($\mathbf{u}_{\mathrm{A}}$, 17)	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3
rotate($\mathbf{u}_{\mathrm{A}}$, 18)	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2
rotate($\mathbf{u}_{\mathrm{A}}$, 19)	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1
rotate($\mathbf{u}_{\mathrm{A}}$, 20)	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0
rotate($\mathbf{u}_{\mathrm{A}}$, 21)	5	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5
rotate($\mathbf{u}_{\mathrm{A}}$, 22)	4	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5
rotate($\mathbf{u}_{\mathrm{A}}$, 23)	3	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4
rotate($\mathbf{u}_{\mathrm{A}}$, 24)	0	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3
rotate($\mathbf{u}_{\mathrm{A}}$, 25)	5	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0
rotate($\mathbf{u}_{\mathrm{A}}$, 26)	4	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5
rotate($\mathbf{u}_{\mathrm{A}}$, 27)	2	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4
rotate($\mathbf{u}_{\mathrm{A}}$, 28)	1	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2
rotate($\mathbf{u}_{\mathrm{A}}$, 29)	0	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1
rotate($\mathbf{u}_{\mathrm{A}}$, 30)	5	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0
rotate($\mathbf{u}_{\mathrm{A}}$, 31)	4	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5
rotate($\mathbf{u}_{\mathrm{A}}$, 32)	3	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4
rotate($\mathbf{u}_{\mathrm{A}}$, 33)	3	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3
rotate($\mathbf{u}_{\mathrm{A}}$, 34)	2	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3
rotate($\mathbf{u}_{\mathrm{A}}$, 35)	1	4	3	2	0	5	4	3	2	1	1	0	5	2	1	0	4	3	2	1	0	5	5	4	3	0	5	4	2	1	0	5	4	3	3	2

(a)

timeslots	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
rotate($\mathbf{v}_{\mathrm{A}}$, 0)	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1
rotate($\mathbf{v}_{\mathrm{A}}$, 1)	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2
rotate($\mathbf{v}_{\mathrm{A}}$, 2)	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3
rotate($\mathbf{v}_{\mathrm{A}}$, 3)	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4
rotate($\mathbf{v}_{\mathrm{A}}$, 4)	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5
rotate($\mathbf{v}_{\mathrm{A}}$, 5)	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0

(b)

Fig. 5. Sequences in ARCH when $N = 6$ and CH seed $k = 1$. (a) There are N^2 a-alternative sequences in ARCH. (b) There are N a-default sequences in ARCH.

distinct a-default sequences (i.e., $rotate(\boldsymbol{v}_{\mathrm{A}}, 2i)$s and $rotate(\boldsymbol{v}_{\mathrm{A}}, 2i{+}1)$s), see Fig. 5. For ease of discussion, we consider the synchronous system S below.

Without loss of generality, assume that (t_1, c) is the busiest, i.e., it is included in the most sequences. To show the channel loading of ARCH to be $1/N = (1 + N)/(N + N^2)$, it is sufficient to show that (t_1, c) is included in one a-default sequence and N a-alternative sequences in S.

Assume that a-alternative sequence $\boldsymbol{u}_{\mathrm{A}}$ includes (t_1, c). We claim that $\boldsymbol{u}_{\mathrm{A}}$ visits channel c N times. Recall that $\boldsymbol{u}_{\mathrm{A}}$ is obtained by rearranging subsequences of the following N sequences: $rotate(\boldsymbol{x}, 0), rotate(\boldsymbol{x}, 1), \ldots,$ and $rotate(\boldsymbol{x}, N - 1)$, where $\boldsymbol{x}$ is a sequence visiting each of the N channel once. Keep in mind that *rotate* operations and rearranging subsequences of $rotate(\boldsymbol{x}, i)$ do not change the number of channel c to be visited. So, our claim follows.

Next, we show that (t_1, c) is included in N a-alternative sequences. Suppose that $\boldsymbol{u}_{\mathrm{A}}$ visits channel c at time slots $t_1, t_2, \ldots,$ and t_{N}. Notice that, each a-alternative sequence in S could be obtained by rotating $\boldsymbol{u}_{\mathrm{A}}$. Hence, an a-alternative sequence $\boldsymbol{w}_{\mathrm{A}}$ in S includes (t_1, c) if and only if $\boldsymbol{w}_{\mathrm{A}}$ is one of the following sequences: $rotate(\boldsymbol{u}_{\mathrm{A}}, t_1 - t_1), rotate(\boldsymbol{u}_{\mathrm{A}}, t_2 - t_1), \ldots,$ or $rotate(\boldsymbol{u}_{\mathrm{A}}, t_{\mathrm{N}} - t_1)$. So, there are N a-alternative sequences including (t_1, c). In Fig. 5a, $\boldsymbol{u}_{\mathrm{A}} = rotate(\boldsymbol{u}_{\mathrm{A}}, 0)$ visits channel 3

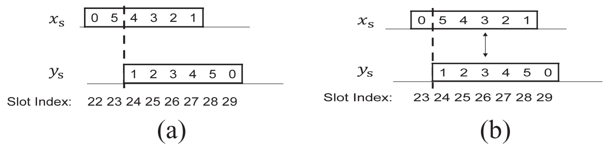

Fig. 6. Rendezvous between x_s and y_s. (a) x_s and y_s have distinct time parity. (b) x_s and y_s have the same time parity.

at time slots 1, 6, 16, 23, 32, and 33. Besides, (1, 3) is included in the following $N = 6$ a-alternative sequences, $rotate(\boldsymbol{u}_A, 1-1)$, $rotate(\boldsymbol{u}_A, 6-1)$, $rotate(\boldsymbol{u}_A, 16-1)$, $rotate(\boldsymbol{u}_A, 23-1)$, $rotate(\boldsymbol{u}_A, 32-1)$, and $rotate(\boldsymbol{u}_A, 33-1)$.

By definition, it is not difficult to see that there is one a-default sequence including (t_1, c). Hence, the theorem follows. □

3.4 The Proposed Symmetric Asynchronous Approach

In this section, we introduce symmetric asynchronous rendezvous channel-hopping scheme (*SARCH*) for symmetric asynchronous environments. Suppose that counterclockwise CH sequence $\boldsymbol{x}_s$ and clockwise CH sequence $\boldsymbol{y}_s$ are mutual reflection sequence. Then, $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ can rendezvous with each other if they have the same time parity. For example, $\boldsymbol{x}_s$ visits channels 0, 5, 4, 3, 2, 1, in order, and $\boldsymbol{y}_s$ visits channels 1, 2, 3, 4, 5, 0, in order. In Fig. 6a, $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ have distinct time parity, and cannot rendezvous with each other, while in Fig. 6b, $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ have the same time parity and can successfully rendezvous. The correctness is proved in the following lemma.

Lemma 12. *Mutual reflection sequences $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ can rendezvous with each other if $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ have the same time parity.*

Proof. Without loss of generality, assume that $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ are initially executed at global time slots t_x and t_y, respectively. Notice that $\boldsymbol{x}_s$ is counterclockwise and $\boldsymbol{y}_s$ is clockwise. If $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ have the same time parity, then t_x and t_y must have distinct parity. In Fig. 6b, both $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ have odd-time-parity, but $t_x = 23$ and $t_y = 24$ have distinct parity. Hence, it is sufficient to show that $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ can rendezvous with each other if t_x and t_y have distinct parity.

At an arbitrary time slot t, sequences $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ should stay at channels $(a - (t - t_x)k) \bmod N$ and $((a - (N-1)k) \bmod N + (t - t_y)k) \bmod N$, respectively, where a is the initial channel of $\boldsymbol{x}_s$ and k is the CH seed of $\boldsymbol{x}_s$. Sequences $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ can rendezvous with each other when (1) holds:

$$(a - (t - t_x)k) \bmod N = ((a - (N-1)k) \bmod N + (t - t_y)k) \bmod N. \quad (1)$$

Clearly, (1) can be simplified to be $-(t - t_x)k + \alpha N = (1 + t - t_y)k$, i.e.,

$$\alpha N = (2t + 1 - t_x - t_y)k. \quad (2)$$

If t_x and t_y have distinct parity, then $(1 - t_x - t_y)$ must be even. Since N is even (in a potential CH system), we have that (2) holds when $\alpha = k$ and $t = (N - (1 - t_x - t_y))/2$. Hence, the lemma follows. □

timeslots	0	1	2	3	4	5	6	7	8	9	10	11	12
rotate($\mathbf{u}_{SA}$, 0)	0	5	4	3	2	1	1	2	3	4	5	0	0
rotate($\mathbf{u}_{SA}$, 1)	5	4	3	2	1	1	2	3	4	5	0	0	0
rotate($\mathbf{u}_{SA}$, 2)	4	3	2	1	1	2	3	4	5	0	0	0	5
rotate($\mathbf{u}_{SA}$, 3)	3	2	1	1	2	3	4	5	0	0	0	5	4
rotate($\mathbf{u}_{SA}$, 4)	2	1	1	2	3	4	5	0	0	0	5	4	3
rotate($\mathbf{u}_{SA}$, 5)	1	1	2	3	4	5	0	0	0	5	4	3	2
rotate($\mathbf{u}_{SA}$, 6)	1	2	3	4	5	0	0	0	5	4	3	2	1
rotate($\mathbf{u}_{SA}$, 7)	2	3	4	5	0	0	0	5	4	3	2	1	1
rotate($\mathbf{u}_{SA}$, 8)	3	4	5	0	0	0	5	4	3	2	1	1	2
rotate($\mathbf{u}_{SA}$, 9)	4	5	0	0	0	5	4	3	2	1	1	2	3
rotate($\mathbf{u}_{SA}$, 10)	5	0	0	0	5	4	3	2	1	1	2	3	4
rotate($\mathbf{u}_{SA}$, 11)	0	0	0	5	4	3	2	1	1	2	3	4	5
rotate($\mathbf{u}_{SA}$, 12)	0	0	5	4	3	2	1	1	2	3	4	5	0

Fig. 7. $rotate(\boldsymbol{u}_{SA}, *)$s when $N = 6$, $a = 0$, and $k = 1$. A gray grid denotes a rendezvous with $rotate(\boldsymbol{u}_{SA}, 0)$.

The detail of SARCH is described below:

- *Sa-default CH sequence*. Each node p constructs the same seed sequence $\boldsymbol{u}_{SA}$ by the aid of default initial channel a and default CH seed k. A seed CH sequence $\boldsymbol{u}_{SA}$ is generated by concatenating mutual reflection sequences $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$, and a parity slot z, where $\boldsymbol{x}_s = \{(0, a), (1, (a - k) \bmod N), \ldots, (i, (a - ik) \bmod N), \ldots, (N-1, (a - (N-1)k) \bmod N)\}, \mathbf{y}_s = \{(0, (a - (N-1)k) \bmod N), \ldots, (N-1, a)\}$ and $z = \{(0, a)\}$. After obtaining the seed sequence $\boldsymbol{u}_{SA}$, node p randomly selects an integer r as its *rotation seed* and constructs its sa-default sequence by concatenating the following sequences: $rotate(\boldsymbol{u}_{SA}, r), rotate(\boldsymbol{u}_{SA}, r)$, $rotate(\boldsymbol{u}_{SA}, 2r \bmod (2N+1)), rotate(\boldsymbol{u}_{SA}, 2r \bmod (2N+1)), \ldots, rotate(\boldsymbol{u}_{SA}, (2N+1) \times r \bmod (2N+1))$ and $rotate(\boldsymbol{u}_{SA}, (2N+1) \times r \bmod (2N+1))$. Notice that $2N+1$ should be prime and an sa-default CH sequence contains two $rotate(\boldsymbol{u}_{SA}, j \times r \bmod (2N+1))$s for each j.

Fig. 7 shows all $rotate(\boldsymbol{u}_{SA}, r)$s when $N = 6$, $a = 0$, and $k = 1$, and Fig. 8a shows the rendezvous between two SUs, Q_1 and Q_2, when their rotation seeds are 1 and 3, respectively.

N degree of rendezvous is a sufficient condition for bounded MCTTR. However, sa-default sequences have only 1 degree of rendezvous (refer to Fig. 8b). To guarantee a bounded MCTTR (i.e., to guarantee rendezvous in the case that not all channels are ensured to be available to every SU), we introduce sa-adaptive CH sequences, which are generated by replacing each unavailable channel of a SU in its sa-default CH sequence with an available channel of the SU. Each node first perform sensing to determine its set of available channels (available channel set, for short), and then determines its sa-adaptive sequences by the aid of its available channel set. Let $A_p = \{c_1, c_2, \ldots, c_h\}$ denote the available channel set of node p, where h is an integer.

- *Sa-adaptive CH sequence*. Each node p initially constructs its sa-default CH sequence by concatenating the following $2 \times (2N+1)$ sequences: $rotate(\boldsymbol{u}_{SA}, r)$, $rotate(\boldsymbol{u}_{SA}, r), \ldots, rotate(\boldsymbol{u}_{SA}, (2N+1) \times r \bmod (2N+1))$ and $rotate(\boldsymbol{u}_{SA}, (2N+1) \times r \bmod (2N+1))$. When all channels are available to p (i.e., $|A_p| = N$), p's sa-adaptive CH sequence is the same as its sa-default sequence. When there exist a channel unavailable to p (i.e., $|A_p| < N$), p's sa-adaptive CH sequence is

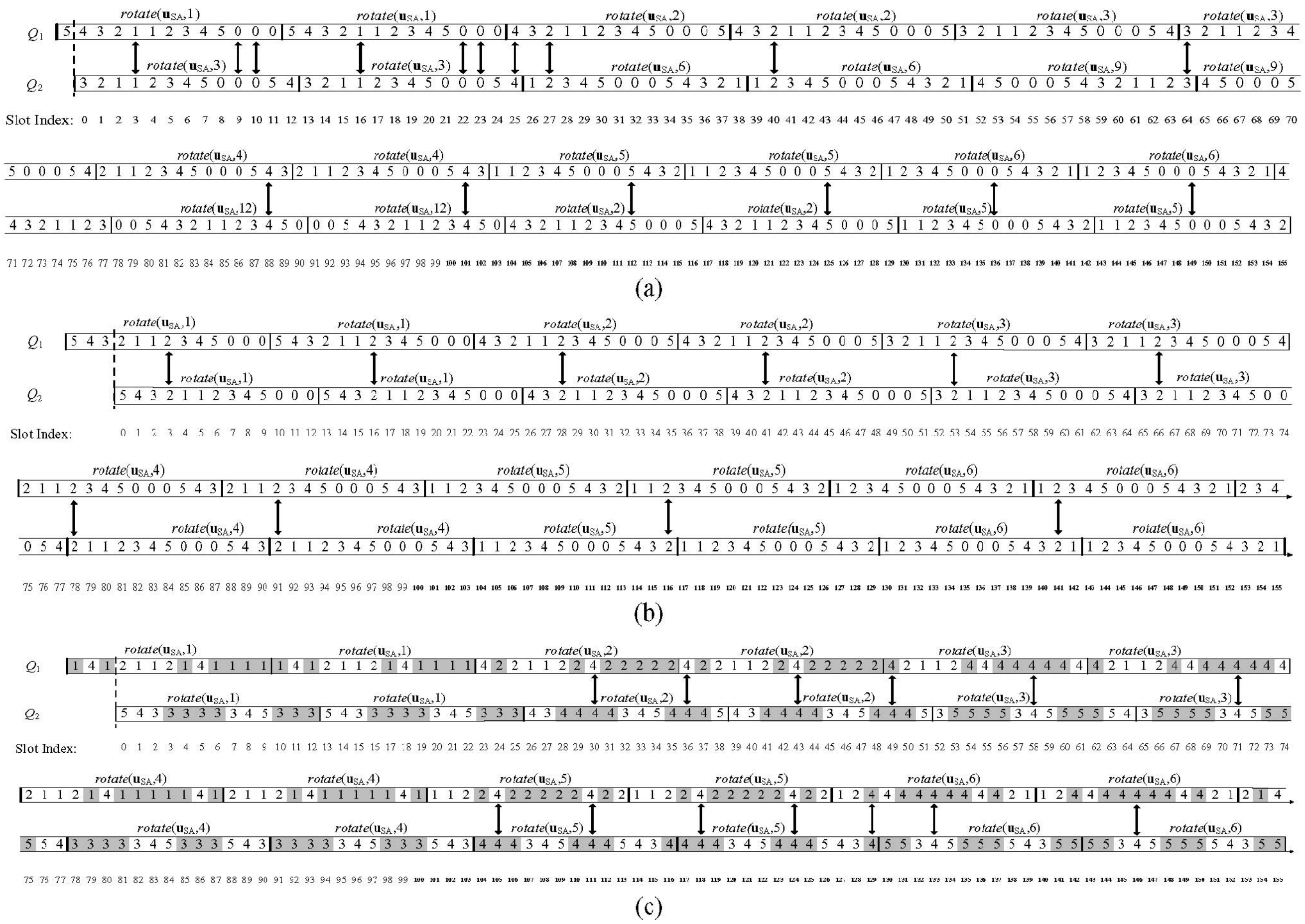

Fig. 8. Rendezvous in SARCH with $N = 6$, $k = 1$, $a = 0$, and the available channel sets of SUs Q_1 and Q_2 are $\{1, 2, 4\}$ and $\{3, 4, 5\}$, respectively. (a) Sa-default CH sequences of SUs Q_1 and Q_2 when their rotation seeds are 1 and 3, respectively. They can rendezvous at common available channel 4. (b) Sa-default CH sequences of SUs Q_1 and Q_2 when their rotation seeds are both 1. They rendezvous at channel 2 which is unavailable to Q_2. (c) Two sa-adaptive CH sequences which are corresponding to the two sa-default CH sequences in (b), respectively, can rendezvous at common available channel 4.

obtained by modifying its sa-default CH sequence as follows: each unavailable channel in $rotate(\boldsymbol{u}_{\text{SA}}, \beta \times r \bmod (2N+1))$ of p's sa-default CH sequence is replaced with the $((\beta - 1) \bmod h+1)$th element of A_{p}, i.e., channel $c_{((\beta-1) \bmod h+1)}$, where $1 \leq \beta \leq 2N+1$. Fig. 8c shows the rendezvous of two sa-adaptive CH sequences corresponding to the two sa-default CH sequences in Fig. 8b when their available channel sets are $\{1, 2, 4\}$ and $\{3, 4, 5\}$, respectively. Channels in gray blocks denote the available channels which are used to replace unavailable channels in sa-default CH sequences. For example, at time slot 4, Q_2's sa-default CH sequence stays at channel 1 (in $rotate(\boldsymbol{u}_{\text{SA}}, 1) = rotate(\boldsymbol{u}_{\text{SA}}, 1 \times r \bmod 13)$) as shown in Fig. 8b, while Q_2's sa-adaptive CH sequence stays at channel 3 (i.e., the first element of Q_2's available channel set $\{3, 4, 5\}$) as shown in Fig. 8c.

3.4.1 Evaluation Metrics and SARCH

Refer to Fig. 8a. Every two consecutive $rotate(\boldsymbol{u}_{\text{SA}}, i)$s in SU Q_1's sa-default CH sequence can rendezvous with at least one $rotate(\boldsymbol{u}_{\text{SA}}, {}^*)$ in SU Q_2's sa-default CH sequence, and vice versa. For example, two $rotate(\boldsymbol{u}_{\text{SA}}, 1)$s in Q_1's sa-default CH sequence can rendezvous two $rotate(\boldsymbol{u}_{\text{SA}}, 3)$s in Q_2's sa-default CH sequence. Below we prove the correctness of the statement.

Lemma 13. *Let D be the sequences concatenating two $rotate(\boldsymbol{u}_{\text{SA}}, r)$s. D and $rotate(\boldsymbol{u}_{\text{SA}}, r')$ can rendezvous with each other if the execution period of $rotate(\boldsymbol{u}_{\text{SA}}, r')$ is completely covered by that of D.*

Proof. Let D' be the sequences concatenating two $rotate(\boldsymbol{u}_{\text{SA}}, r')$s, respectively. When the execution period of $rotate(\boldsymbol{u}_{\text{SA}}, r')$ is completely covered by that of D, it is not difficult to see that D and $rotate(\boldsymbol{u}_{\text{SA}}, r')$ can rendezvous with each other if and only if D and D' can rendezvous with each other.

Let $\boldsymbol{x}_{\text{s}}{}^0$ and $\boldsymbol{y}_{\text{s}}{}^0$ be the substrings of $\boldsymbol{x}_{\text{s}}$ and $\boldsymbol{y}_{\text{s}}$, which are behind z in $rotate(\boldsymbol{u}_{\text{SA}}, {}^*)$, and let $\boldsymbol{x}_{\text{s}}{}^1$ and $\boldsymbol{y}_{\text{s}}{}^1$ be the substrings of $\boldsymbol{x}_{\text{s}}$ and $\boldsymbol{y}_{\text{s}}$, which are ahead z in $rotate(\boldsymbol{u}_{\text{SA}}, {}^*)$, i.e., $\boldsymbol{x}_{\text{s}} = \boldsymbol{x}_{\text{s}}{}^0\boldsymbol{x}_{\text{s}}{}^1$ and $\boldsymbol{y}_{\text{s}} = \boldsymbol{y}_{\text{s}}{}^0\boldsymbol{y}_{\text{s}}{}^1$. Refer to Fig. 7, in $rotate(\boldsymbol{u}_{\text{SA}}, 3)$, z is executed at time slot 9, $\boldsymbol{x}_{\text{s}}{}^0 = 054$ and $\boldsymbol{x}_{\text{s}}{}^1 = 321$. Therefore, $rotate(\boldsymbol{u}_{\text{SA}}, {}^*)$ must be one of the following five sequences, $\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}z$, $\boldsymbol{y}_{\text{s}}z\boldsymbol{x}_{\text{s}}$, $z\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}$, $\boldsymbol{y}_{\text{s}}{}^1z\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}{}^0$, and $\boldsymbol{x}_{\text{s}}{}^1\boldsymbol{y}_{\text{s}}z\boldsymbol{x}_{\text{s}}{}^0$. In Fig. 7, $rotate(\boldsymbol{u}_{\text{SA}}, 0)$ is $\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}z$, $rotate(\boldsymbol{u}_{\text{SA}}, 6)$ is $\boldsymbol{y}_{\text{s}}z\boldsymbol{x}_{\text{s}}$, $rotate(\boldsymbol{u}_{\text{SA}}, 12)$ is $z\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}$, $rotate(\boldsymbol{u}_{\text{SA}}, 7)$ to $rotate(\boldsymbol{u}_{\text{SA}}, 11)$ are $\boldsymbol{y}_{\text{s}}{}^1z\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}{}^0$, and $rotate(\boldsymbol{u}_{\text{SA}}, 1)$ to $rotate(\boldsymbol{u}_{\text{SA}}, 5)$ are $\boldsymbol{x}_{\text{s}}{}^1\boldsymbol{y}_{\text{s}}z\boldsymbol{x}_{\text{s}}{}^0$.

Let d denote the sequence $\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}z\boldsymbol{x}_{\text{s}}$, and d' denote the sequence $\boldsymbol{y}_{\text{s}}z\boldsymbol{x}_{\text{s}}\boldsymbol{y}_{\text{s}}$. Refer to Figs. 9a and 9b, each of D and D' contains either d or d'. For ease of the discussion, d or d' contained in D and d' are called S and S', respectively.

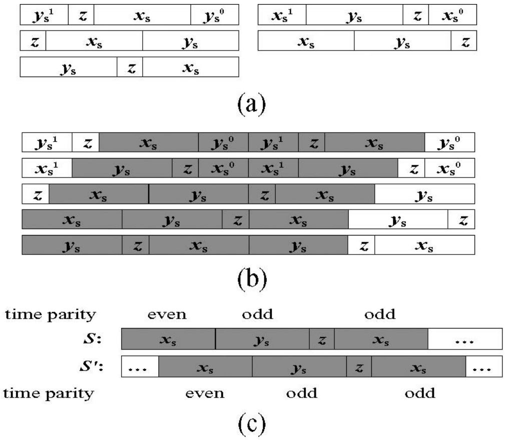

Fig. 9. D and $rotate(\boldsymbol{u}_{\rm SA},^*)$. (a) Types of $rotate(\boldsymbol{u}_{\rm SA},^*)$, where $\boldsymbol{x}_{\rm s.} = \boldsymbol{x}_{\rm s}{}^0\boldsymbol{x}_{\rm s}{}^1$ and $\boldsymbol{y}_{\rm s.} = \boldsymbol{y}_{\rm s}{}^0\boldsymbol{y}_{\rm s}{}^1$. (b) Types of D and D'. (c) Rendezvous of S and S'.

Consider the case that both S and S' are d. Without loss of generality, assume that S is executed earlier than S' (refer to Fig. 9c). According to Lemma 12, if $\boldsymbol{y}_{\rm s}$ in S and the first $\boldsymbol{x}_{\rm s}$ in S' have the same time parity, D and d' can rendezvous with each other (because $\boldsymbol{x}_{\rm s}$ and $\boldsymbol{y}_{\rm s}$ are mutual reflection sequences). If not, we claim that the second $\boldsymbol{x}_{\rm s}$ in S and $\boldsymbol{y}_{\rm s}$ in S' must have the same time parity. Without loss of generality, assume that $\boldsymbol{y}_{\rm s}$ in S and the first $\boldsymbol{x}_{\rm s}$ in $\boldsymbol{S}'$ have odd-time-parity and even-time-parity, respectively. By the aid of parity slot z and the even parity of N, both the second $\boldsymbol{x}_{\rm s}$ in S and $\boldsymbol{y}_{\rm s}$ in $\boldsymbol{S}'$ must have odd parity (refer to Fig. 9c). So, D and D' can rendezvous with each other. The proofs of the remaining three cases of S and S' are very similar. Due to the limitation of space, we omit them here. □

Theorem 14. *SARCH has* $MTTR < 2 \times (2N+1)$.

Proof. It is sufficient to show that any two sa-default CH sequences can rendezvous within $2 \times (2N+1)$ time slots. Suppose that two SUs, Q_1 and Q_2, select r and r', as their rotation seeds, respectively. Let D be the sequence concatenating two consecutive $rotate(\boldsymbol{u}_{\rm SA}, i \times r)$s in Q_1's sa-default CH sequence. It is not difficult to see that there exist a $rotate(\boldsymbol{u}_{\rm SA}, j \times r' \bmod (2N+1))$ in Q_2's sa-default CH sequence such that the execution period of $rotate(\boldsymbol{u}_{\rm SA}, j \times r' \bmod (2N+1))$ is completely covered by that of D. For example, in Fig. 8a, the execution period of the first $rotate(\boldsymbol{u}_{\rm SA}, 3)$ in Q_2's sa-default CH sequence is completely covered by the execution period of two $rotate(\boldsymbol{u}_{\rm SA}, 1)$s in Q_1's sa-default CH sequence. According to Lemma 13, D and $rotate(\boldsymbol{u}_{\rm SA}, j \times r' \bmod (2N+1))$ can rendezvous with each other. Hence, the theorem follows.□

Theorem 15. *SARCH has channel loading* $= 3/(2N+1)$ *when all channels are available to all SUs.*

Proof. When all channels are available to all SUs, every SU executes its own sa-default sequence. Without loss of generality, assume that (t_1, c) is the busiest, and sa-default CH sequence S includes (t_1, c), where S concatenates $rotate(\boldsymbol{u}_{\rm SA}, ra), rotate(\boldsymbol{u}_{\rm SA}, r), \ldots, rotate(\boldsymbol{u}_{\rm SA}, N \times r \bmod (2N+1))$, and $rotate(\boldsymbol{u}_{\rm SA}, N \times r \bmod (2N+1))$, and $\boldsymbol{u}_{\rm SA}$ has initial channel a. We claim that S visits channel c $3 \times 2N$ times if $c = a$, and $2 \times 2N$ times otherwise. By definition, $rotate(\boldsymbol{u}_{\rm SA},^*)$ visits initial channel a three times and each of remaining $(N-1)$ channels two times (refer to Fig. 7). Since S includes $2N$ $rotate(\boldsymbol{u}_{\rm SA},^*)$s, our claim holds.

Consider the case that $c = a$ and S visits channel c at time slots $t_1, t_2, \ldots,$ and $t_{3\times 2N}$ in a CH period. Then, there are $3 \times 2N$ $rotate(S,^*)$s including (t_1, c), i.e., $rotate(S, t_1 - t_1)$, $rotate(S, t_2 - t_1), \ldots,$ and $rotate(S, t_{3\times 2N} - t_1)$. Similarly, when $c \neq a$, there are $2 \times 2N$ $rotate(S,^*)$s including (t_1, c). Notice that there are N distinct rotation seeds in SARCH, i.e., there are N distinct Ss. So, there are $3 \times 2N \times N$ $rotate(^{**},^*)$s including (t_1, c) if $c = a$, and $2 \times 2N \times N$ $rotate(^{**},^*)$s including (t_1, c) otherwise.

Due to the misalignment of CH period boundaries, for each S, there are $(2N+1) \times 2N$ $rotate(S,^*)$s in SARCH, which implies that there are $(2N+1) \times 2N \times N$ distinct $rotate$ $(^{**,*})$s in SARCH. Therefore, the loading of channel c is $3\times 2N\times N/((2N+1)\times 2N\times N) = 3/(2N+1)$ when $c = a$, and $2/(2N+1)$ otherwise. □

According to the proof above, it is not difficult to see that SARCH has optimal average channel loading $1/N (= (3/(2N+1) + (N-1) \times 2/(2N+1))/N)$ when all channels are available to every SU.

Theorem 16. *SARCH has* $MCTTR < 2\times(2N+1)\times(2N+1)$ *if* $2N+1$ *is prime.*

Proof. The proof can be found in the supplemental file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2012.260. □

4 Comparisons

In this section, we compare the proposed CH systems with existing CH schemes using four metrics: degree of rendezvous, channel loading, MTTR, and MCTTR.

4.1 Symmetric Synchronous Approaches

Bind rendezvous (*BR*) *channel-hopping* [16]. In this scheme, each node hops from one channel to another randomly. At a particular instant, a node occupies one of these channels with probability $1/N$, where N is the total number of channels. When two nodes occupy the same channel at the same time, rendezvous occurs. The BR scheme does not guarantee a bounded MTTR between any two CH sequences.

Slotted Seeded Channel-Hopping (*SSCH*) [23]. Each node is allowed to have one or multiple (initial channel, CH seed)-pairs to determine its CH sequences. Each sequence period includes a parity slot at which time instant all nodes with the same seed are guaranteed to rendezvous on a channel indicated by the seed value. Since SSCH has $(N-1)$ CH seeds, channel loading of SSCH is $1/(N-1)$. Each pair of sequences rendezvous exactly once within a CH period, and a CH period is $(N+1)$-time slot long. Thus, the MTTR of SSCH is $(N+1)$. SSCH is a synchronous approach, although results in [23] show that it can tolerate moderate clock skew. The amount of clock skew used in [23] to evaluate SSCH is very small relative to one slot duration.

Dynamic Hopping MAC (*DH-MAC*) [13]. In DH-MAC, each sequence has CH period $= N \times (N+2)$ time slots and

TABLE 1
A Comparison of CH Schemes

	CH schemes	Deg. of rendezv.	Channel Loading	MTTR	MCTTR
Asym. Syn.	RCCH	$N^{*\#}$	$2/N^{\#}$	$N/2^{*\#}$	$N^2/2^{*\#}$
Sym. Syn.	BR [16]	0	-	-	-
	SSCH [23]	1	$1/(N-1)$	$N+1$	-
	L-QCH [21]	N^*	$\geq 1/N^{1/4}$	$\geq N^{1/2}$	$\geq N^{3/2}$
	DH-MAC[13]	1	$1/(N-1)$	$N+2$	-
Asym. Asyn.	A-MOCH [21]	N^*	$1/N^*$	N^2-N+1	N^2
	ACH [12]	N^*	$1/N^*$	N^2-N+1	N^2
	ARCH	$N^{*\#}$	$1/N^{*\#}$	$2N-1^{\#}$	$N^{2*\#}$
Sym. Asyn.	SeqR [17]	1	$1/N^*$	$N(N+1)$	-
	A-QCH [21]	2	$\approx 1/2$	$N/2$	-
	JS [2]	N^*	-	$3P^{\Delta}$	$3NP(P-G^{\ddagger})+3P$
	ASYNC-ETCH[19]	1	$1/N^*$	$N(2N+1)$	-
	CRSEQ [20][2]	N^*	$3/(3P-1)$	$3P^2-3P$	$3P^2-3P$
	DRSEQ [22]	1	$2/(2N+1)$	$2N+1$	-
	GOA[18]	1	$1/N^*$	N^2+N	-
	SARCH	1	$3/(2N+1)^{\dagger}$	$4N+2$	$8N^2+8N$

Asym.: asymmetric
Syn.: synchronous
Sym.: symmetric
Asyn.: asynchronous
**: optimal*
-: unknown
$\triangle$: P is a smallest prime which is larger than N
†: when all channels are available
‡: G is the number of available channels
#: N should be even

is divided into N small cycles. To guarantee rendezvous in a small cycle, two additional time slots (totally $N+2$ time slots per small cycle) are used to visit specific channels. If the specific channels are unavailable (i.e., occupied by PU), the SU redetermines its CH sequences. DH-MAC protocol has MTTR $=N+2$, channel loading $=1/(N-1)$, and degree of rendezvous $=1$.

Quorum-based Channel-Hopping [21]. In [21], symmetric synchronous CH scheme L-QCH selects a rendezvous channel set $c^*=\{c_0,c_1,\ldots,c_{\mathrm{m}}\}$ from N channels, and a time slot set $t^*=\{t_0,t_1,\ldots,t_{\mathrm{k}}\}$ from N time slots. Each sequence has a CH period of N^2 slots. A sequence $\boldsymbol{u}=\{u_0,u_1,\ldots,u_{\mathrm{N^*N}-1}\}$ is determined according to the formula: $u_{\mathrm{i}}=c_{\mathrm{j}}$ if $(i \bmod N)\in t^*$, and $u_{\mathrm{i}}=c$ otherwise, where $j=i/N$ and $c\notin c^*$ is a predetermined channel index. By applying the operation *rotate* to sequence $\boldsymbol{u}$, a number of new sequences can be generated, thereby creating a total of N sequences. L-QCH has MTTR $\geq N^{1/2}$.

Refer to Table 1, RCCH has optimal MTTR, optimal degree of rendezvous, optimal MCTTR, and nearly optimal channel loading.

4.2 Asymmetric Asynchronous Approaches

Asynchronous channel-hopping (ACH) [18]. In ACH, a receiver constructs an $N\times N$ array with each row and column has distinct elements to generate its CH sequence, and a sender constructs an $N\times N$ array whose rows are all the same to generate its sequence. The rendezvous of a sender and a receiver are proved to be over all channels. ACH has MTTR $N\times(N-1)$.

Asynchronous Quorum-based Channel-Hopping (A-MOCH) [21]. In A-MOCH, a receiver sequence is generated by repeating a permutation of elements in $Z_{\mathrm{N}}N$ times, whereas a sender sequence is generated by concatenating N sequences: $rotate(\boldsymbol{x},0), rotate(\boldsymbol{x},1),\ldots,rotate(\boldsymbol{x},N-1)$, where $\boldsymbol{x}$ is a permutation of elements in Z_{N}. A-MOCH's MTTR is N^2-N+1.

Obviously, our ARCH is a better design in terms of the four metrics. Refer to Table 1. ARCH has significant improvements on MTTR, optimal degree of rendezvous, and optimal channel loading, and optimal MCTTR.

4.3 Symmetric Asynchronous Approaches

Sequence-based rendezvous (*SeqR*) [17]. Each sequence generated by the SeqR scheme has CH period $=N\times(N+1)$ time slots. This scheme builds the initial sequence, $\boldsymbol{u}$, by the first selecting a permutation of elements in Z_{N}. Then, it repeats the selected permutation $(N+1)$ times in the sequences $\boldsymbol{u}$ using the following method: the permutation is used contiguously N times, and once the permutation is interspersed with the other N permutations. By applying the operation $rotate(\boldsymbol{u},i)$ for all $i\in[1,N\times(N+1)-1]$ to the initial sequence $\boldsymbol{u}$, a number of new sequences can be generated, thereby creating a total of $N\times(N+1)$ sequences. Collectively, the set of sequences forms an asynchronous CH system that satisfies the rotation closure property. SeqR has MTTR $=N\times(N+1)$, channel loading $=1/N$, and degree of rendezvous $=1$.

Asynchronous Efficient Channel-Hopping [19]. In [19], each sequence $\boldsymbol{u}$ in ASYNC-ETCH is constructed from a sequence, say $\boldsymbol{x}$, in a way that a pair of $\boldsymbol{x}$s are inserted into $\boldsymbol{u}$ following a pilot slot N times, where $\boldsymbol{x}$ is a clockwise sequence with initial channel 0 and a selected CH seed k. The pilot slots of $\boldsymbol{u}$ combined in order are exactly a sequence $\boldsymbol{x}$. An ASYNC-ETCH system has a set of $N-1$ such $\boldsymbol{u}$s (since there are $N-1$ distinct ks) and has MTTR $N\times(2N+1)$.

Jump-stay based Channel-hopping (JS) [2]. Each sequence consists of a jump-pattern and a stay-pattern. Nodes jump on available channels in the jump-pattern and stay on a specific channel in the stay-pattern. A jump-pattern lasts for $2P$ time slots, while a stay-pattern lasts for P time slots, where P is a prime number larger than N. The MTTR of JS has been proved to be $3P$.

Asynchronous Quorum-based Channel-Hopping (A-QCH) [21]. The main idea of A-QCH is the same with that of L-QCH. A-QCH has MTTR $\geq N/2$. However, it is applicable to systems with only two channels.

Generated orthogonal algorithm (GOA) [18]. Each sequence in GOA consists of $(N+1)$ times the permutation of all available channel indices: N times the permutation appears contiguously, and once the permutation appears interspersed with the other N permutations. GOA has MTTR $=N^2+N$, channel loading $=1/N$, and degree of rendezvous $=1$.

Channel rendezvous sequence (CRSEQ) [20]. A CRSEQ is constructed based on triangle numbers and modular operations. CRSEQ has MTTR $=3P^2-3P$, channel loading $=3/(3P-1)$, degree of rendezvous $=N$, and MCTTR $=3P^2-3P$.

Deterministic rendezvous sequence (DRSEQ) [22]. DRSEQ has exactly one sequence which consists of $(2N+1)$ indices, "$1,2,\ldots,N,e,N,N-1,\ldots,1$," where e denotes a null item. DRSEQ has MTTR $=2N+1$, channel loading $=1/N$, and degree of rendezvous $=1$.

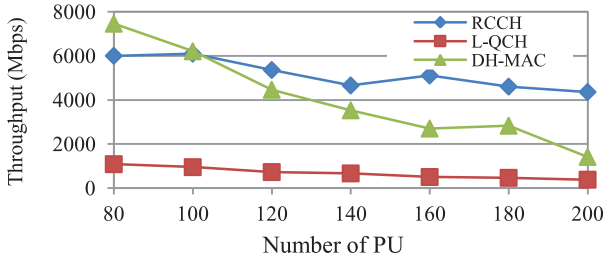

Fig. 10. Throughput versus the number of PUs.

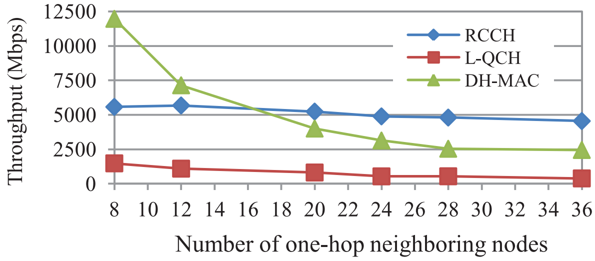

Fig. 11. Throughput versus node density.

SARCH has MCTTR $= 8N^2 + 8N$, MTTR $= 4N + 1$, channel loading $= 3/(2N + 1)$, and degree of rendezvous $= 1$. According to Table 1, no existing symmetric asynchronous channel-hopping algorithms outperform in terms of all metrics. The importance of each metric depends on applications. Our SARCH is a good choice to those applications that have particular requirements on small MTTR, small MCTTR, and balance channel loading.

5 Simulations

In this section, we simulate the proposed algorithm via ns-2 (version 2.31) simulator with the CR transmission model, which is based on Cognitive Radio Cognitive Network Simulator add-on [27]. We use seven reference protocols for comparison: JS [2], ACH [12], DH-MAC [13], ASYNC-ETCH [19], L-QCH [21], A-QCH [21], and A-MOCH [21]. We deploy 80 PUs and 441 SUs in a 1,000 m$\times$1,000 m area. We assume that PUs and SUs have the same communication ranges and node density is 8 unless specified otherwise. SUs can opportunistically access channels, and the channel switching delay is set as 80 μs, which is supported by existing technology [5]. The duration of a time slot is 200 ms. A simulation includes multiple nondisjoint flows, where every SU serves as both a transmitter and a receiver in the multiple flows. SUs randomly generate traffic flow to the one-hop neighboring nodes in an exponential distribution with a mean of 10 sec. The incumbent traffic follows a "busy/idle" pattern on a licensed channel. The busy period and idle period follow an exponential distribution with a mean of 10 and 20 s, respectively. We study the throughput in each CH protocol under varying conditions, including clock skew, number of PU nodes, node density (of SU), and number of available channels.

Rendezvous nodes (i.e., hop to the same channel at the same time slot) contend for the opportunity to exchange control messages/data information. After successful exchanging control messages, for fair comparison, in our simulations, sender has to change its CH sequence in accordance with receiver (that is, each sender has to change its CH sequence and follow the receiver's sequence, called "following state") to exchange data information, and returns back when the exchange is completed. Although following state causes the problem that sender cannot be found in its original CH sequence, we omit the negative effect and focus on the rendezvous opportunity provided by each protocol.

5.1 Time-Synchronous Networks

Our RCCH is the first asymmetric synchronous approach. For fair comparison, we compare our RCCH with three symmetric synchronous protocols, L-QCH, RCCH, and DH-MAC, in symmetric environments. The number of available channels, i.e., N, for L-QCH, RCCH, and DH-MAC are set to 10, 10, and 11, respectively. DH-MAC has 11 channels because DH-MAC is applicable when N is prime, but our RCCH is applicable when N is even. The CH periods for L-QCH, RCCH, and DH-MAC are 70, 100, and 130 time slots, respectively. An L-QCH's CH period is divided into 10 frames and each frame includes seven time slots. For each (time slot, channel), there are three sequences including it. Since there are totally seven distinct sequences in a CH period, L-QCH has channel loading 3/7:

1. *Impact of PU numbers*. Fig. 10 shows that throughput decreases as the number of PUs increases (because the volume of incumbent traffic increases). Throughput of L-QCH is very low even in low-incumbent-traffic environments. This is because L-QCH has a single control channel and high-control channel loading ($= 3/7 >$ RCCHs $(= 2/10)$). A single control channel causes the control channel saturation problem and high channel loading results in high collision rate at incumbent-free channels. DH-MAC has low degree of rendezvous ($= 1 <$ RCCHs $(= N)$), i.e., a pair of SUs may have only one rendezvous channel. In high PU density environments, this property results in long time-to-rendezvous on an incumbent-free channel. In low-PU-density environments, DH-MAC outperforms RCCH because its smaller channel loading results in lower collision rate.
2. *Impact of node density*. In this simulation, communication range of SUs varies from 100 to 250 m to change node density. Fig. 11 indicates that node density has a clear impact on throughput. High node density does not much reduce L-QCH's throughput because its single control channel has resulted in very low throughput. DH-MAC outperforms RCCH in low-node-density environments because of its smaller channel loading. However, in higher node density environments, DH-MAC performance decreases sharply because its long MTTR and its low degree of rendezvous result in long medium access delay.
3. *Impact of number of available channels*. In Fig. 12, we consider various amounts of available channels. Due to the constraint of DH-MAC (i.e., total number of available channels N should be a prime number), there are fewer results for DH-MAC. In high-node-density environments, when the number of available

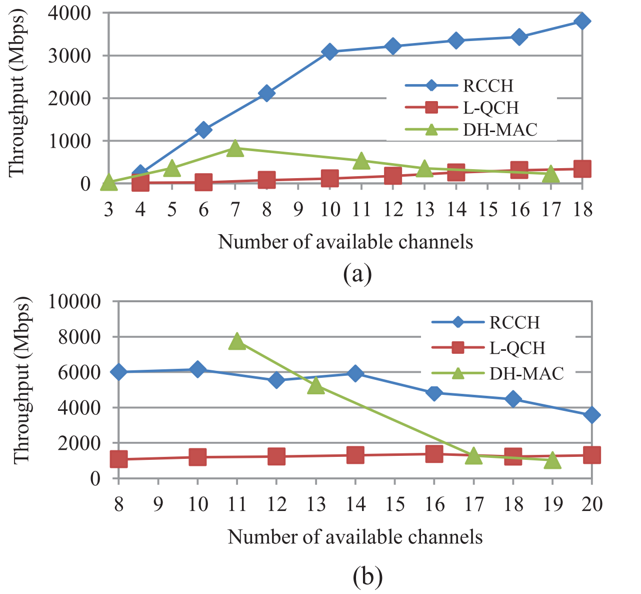

Fig. 12. Throughput versus the number of available channels. (a) High node density. (b) Low node density.

channels increases, more transmissions are allowed, and hence the throughput increases (see Fig. 12a). As expected, RCCH scheme has better performance than L-QCH and DH-MAC. Large N does not much increase L-QCH's throughput because its control channel saturation problem could not be solved (even when N is large). Large N much increases DH-MAC's throughput when $N \leq 7$. However, when $N \geq 11$, DH-MAC's throughput much decreases because of its large MTTR.

In low-node-density environments, large N does not significantly increase the number of successful transmissions, but results in large MTTR of DH-MAC and RCCH (see Fig. 12b). So, their throughput decreases as N increases.

5.2 Time-Asynchronous Networks

Throughout this section, A-QCH's CH period is divided into 10 frames and each frame includes nine time slots. For each (time slot, channel), there are five sequences including it. Since there are totally nine distinct sequences in a CH period, A-QCH has channel loading 5/9. Each value plotted in figures is the average of 20 simulation results:

1. *Impact of PU density*. In this simulation, we consider the throughput with respect to various number of PUs. Fig. 13 shows that throughputs decrease as PU density increases. This is because high PUs density leads to fewer incumbent-free channels, and fewer incumbent-free channels lead to longer expectation time for establishing a communication link, especially for protocols with large MTTR.

   Fig. 13a shows that asymmetric approaches A-MOCH and ACH have nearly the same performance because they have the same MTTR. It also shows that high PU density does not has significant impact on the performance of our ARCH. This is because ARCH has smaller MTTR, optimal MCTTR, and optimal degree of rendezvous, which can

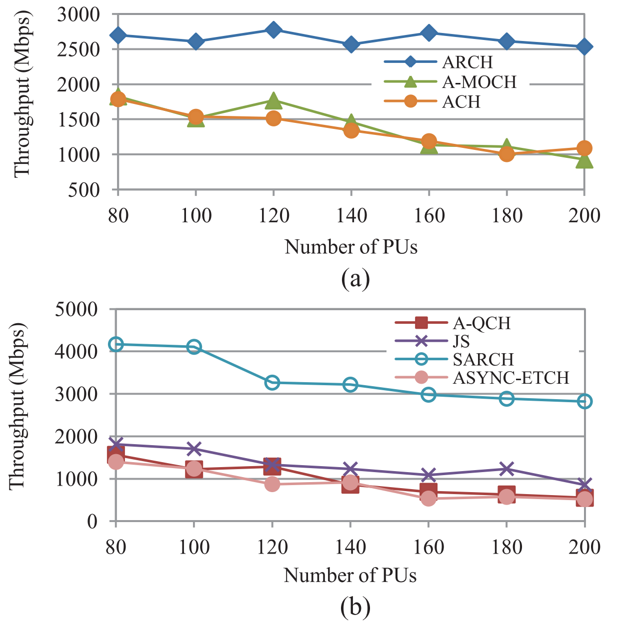

Fig. 13. Throughput versus the number of PUs. (a) Asymmetric. (b) Symmetric.

   contribute to increasing the network capacity. Fig. 13b shows that throughputs of symmetric approaches. Although A-QCH has small MTTR, its single control channel causes control saturation problem and its high channel loading causes high collision rate at incumbent-free channels (especially when the number of incumbent-free channels decreases). SARCH has performance better than JS because SARCH has small MCTTR and small channel loading.

2. *Impact of node density of CRN*. In this simulation, communication ranges of PUs and SUs vary from 100 to 250 m to have various node densities. Figs. 14a and

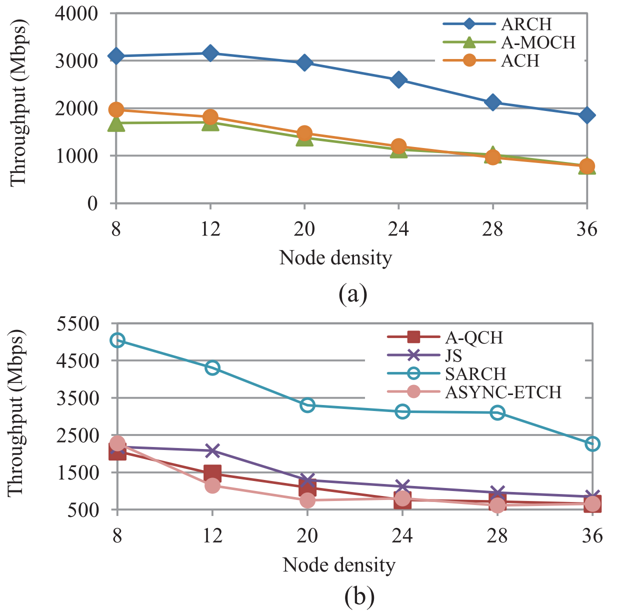

Fig. 14. Throughput versus node density. (a) Asymmetric. (b) Symmetric.

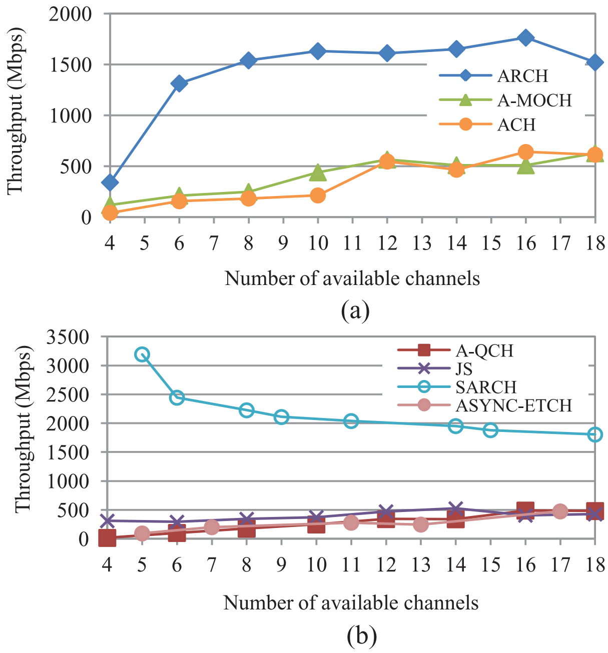

Fig. 15. Throughput versus the number of available channels for high-node-density CRNs. (a) Asymmetric approaches. (b) Symmetric approaches.

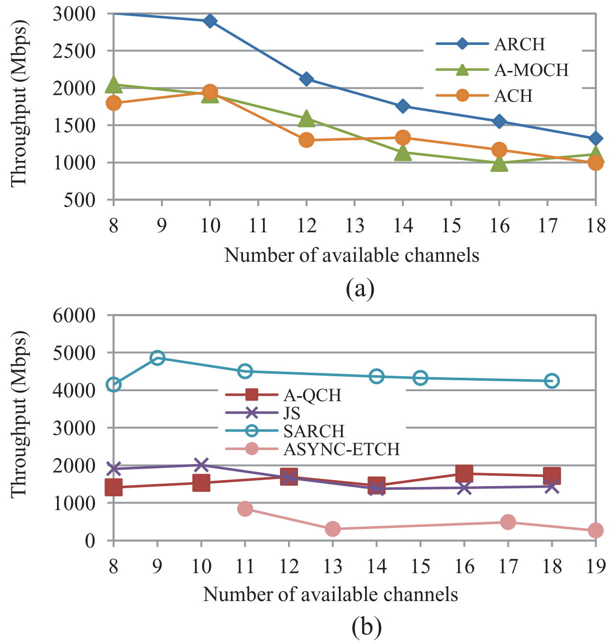

Fig. 16. Throughput versus the number of available channels for low-node-density CRNs. (a) Asymmetric approaches. (b) Symmetric approaches.

14b show that throughput decreases as node density increases. Throughputs decrease because collision rate increases as node density increases, especially for protocols with high channel loading.

Fig. 14b also shows that ASYNC-ETCH and A-QCH have throughputs lower than other protocols. This is because they have high channel loading (high collision rate) and single control channel (causes serious control channel saturation problem), respectively. Although JS also have high channel loading, it has MTTR smaller than ASYNC-ETCH.

3. *Impact of number of available channels*. In Figs. 15 and 16, we consider various amounts of available channels. Fig. 15 shows that in high-node-density environments, when the number of available channels increases, more transmissions are allowable, and hence the throughput increases. Figs. 15a and 15b show that ARCH and SARCH own better performance. This is because both of them have small MTTR, small MCTTR, and small channel loading. SARCH's performance decreases as N increases because larger N leads to longer time-to-rendezvous. Fig. 15b also shows that A-QCH's throughput does not much increase as N increases because its control channel saturation problem could not be solved (even when N is large), while the throughputs of remaining schemes do not much increase because of their larger MTTR and higher channel loading.

Fig. 16 shows that in low-node-density environments, throughputs decrease as the number of available channels increases. More available channels do not much increase successful transmissions (due to unbalance channel loading and fewer transmission requirements), but much increase average time-to-rendezvous (which implies long average time of successful transmission). A-QCH's throughput does not decrease due to its single control channel.

6 Conclusion

In this paper, we introduce three channel-hopping algorithms, RCCH for asymmetric synchronous environments, and ARCH and SARCH for asymmetric asynchronous environments and symmetric asynchronous environments, respectively. Our approaches outperform the state of the art in terms of MTTR, channel loading, degree of rendezvous, and MCTTR.

References

[1] *IEEE P802.22/D0.1 Draft Standard for Wireless Regional Area Networks Part 22: Cognitive Wireless RAN Medium Access Control (MAC) and Physical Layer (PHY)*, IEEE, 2007.

[2] H. Liu, Z. Lin, X. Chu, and Y.-W. Leung, "Jump-Stay Rendezvous Algorithm for Cognitive Radio Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 23, no. 10, pp. 1867-1881, Oct. 2012.

[3] S. Kirshamurthy, M. Thoppian, S. Venkatesan, and R. Prakash, "Control Channel Based MAC-Layer Configuration, Routing and Situation Awareness for Cognitive Radio Networks," *Proc. IEEE Military Comm. Conf. (MILCOM)*, pp. 455-460, 2005.

[4] L. Le and E. Hossain, "OSA-MAC: A MAC Protocol for Opportunistic Spectrum Access in Cognitive Radio Networks," *Proc. IEEE Wireless Comm. & Networking Conf. (WCNC)*, pp. 1426-1430, 2008.

[5] F.H.P. Fitzek, D. Angelini, G. Mazzini, and M. Zorzi, "Design and Performances of an Enhanced IEEE 802.11 MAC Protocol for Multihop Coverage Extension," *IEEE Wireless Comm.*, vol. 10, no. 6, pp. 30-39, Dec. 2003.

[6] Y.T. Hou, Y. Shi, and H.D. Sherali, "Spectrum Sharing for Multi-Hop Networking with Cognitive Radios," *IEEE J. Selected Areas in Comm.*, vol. 26, no. 1, pp. 146-155, Jan. 2008.

[7] H.A.B. Salameh, M.M. Krunz, and O. Younis, "MAC Protocol for Opportunistic Cognitive Radio Networks with Soft Guarantees," *IEEE Trans. Mobile Computing*, vol. 8, no. 10, pp. 1339-1352, Oct. 2009.

[8] Y. Shi, Y.T. Hou, H. Zhou, and S.F. Midkiff, “Distributed Cross-Layer Optimization for Cognitive Radio Networks,” *IEEE Trans. Vehicular Technology,* vol. 59, no. 8, pp. 4058-4069, Oct. 2010.

[9] K.R. Chowdhury and I.F. Akyildiz, “CRP: A Routing Protocol for Cognitive Radio Ad Hoc Networks,” *IEEE J. Selected Areas in Comm.,* vol. 29, no. 4, pp. 794-804, Apr. 2011.

[10] C. Gao, Y. Shi, Y.T. Hou, H.D. Sherali, and Z. Huaibei, “Multicast Communications in Multi-Hop Cognitive Radio Networks,” *IEEE J. Selected Areas in Comm.,* vol. 29, no. 4, pp. 784-793, Apr. 2011.

[11] J. Tang, R. Hincapie, G. Xue, W. Zhang, and R. Bustamante, “Fair Bandwidth Allocation in Wireless Mesh Networks with Cognitive Radios,” *IEEE Trans. Vehicular Technology,* vol. 59, no. 3, pp. 1487-1496, Mar. 2010.

[12] K. Bian and J.-M Park, “Maximizing Rendezvous Diversity in Rendezvous Protocols for Decentralized Cognitive Radio Networks,” *IEEE Trans. Mobile Computing,* vol. 12, no. 7, pp. 1294-1307, July 2013.

[13] C.-F. Shih, T.Y. Wu, and W. Liao, “DH-MAC: A Dynamic Channel Hopping MAC Protocol for Cognitive Radio Networks,” *Proc. IEEE Int’l Conf. Comm. (ICC),* pp. 1-5, 2010.

[14] S. Kaneko, S. Nomoto, T. Ueda, S. Nomura, and K. Takeuchi, “Predicting Radio Resource Availability in Cognitive Radio - an Experimental Examination,” *Proc. IEEE Cognitive Radio Oriented Wireless Networks and Comm. (CrownCom),* pp. 1-6, 2008.

[15] B.F. Lo, I.F. Akyildiz, and A.M. Al-Dhelaan, “Efficient Recovery Control Channel Design in Cognitive Radio Ad Hoc Networks,” *IEEE Trans. Vehicular Technology,* vol. 59, no. 9, pp. 4513-4526, Nov. 2010.

[16] M.D. Silvius, F. Ge, A. Young, A.B. MacKenzie, and C.W. Bostian, “Smart Radio: Spectrum Access for First Responders,” *Proc. SPIE,* vol. 6980, pp. 698008-698008-12, 2008.

[17] L.A. DaSilva and I. Guerreiro, “Sequence-Based Rendezvous for Dynamic Spectrum Access,” *Proc. IEEE Dynamic Spectrum Access Networks Symp. (DySpan),* pp. 1-7, 2008.

[18] N.C. Theis, R.W. Thomas, and L.A. DaSilva, “Rendezvous for Cognitive Radios,” *IEEE Trans. Mobile Computing,* vol. 10, no. 2, pp. 216-227, Feb. 2010.

[19] Y. Zhang, Q. Li, G. Yu, and B. Wang, “ETCH: Efficient Channel Hopping for Communication Rendezvous in Dynamic Spectrum Access Networks,” *Proc. IEEE INFOCOM,* 2011.

[20] J. Shin, D. Yang, and C. Kim, “A Channel Rendezvous Scheme for Cognitive Radio Networks,” *IEEE Comm. Letters,* vol. 14, no. 10, pp. 954-956, Oct. 2010.

[21] K. Bian, J.-M. Park, and R. Chen, “Control Channel Establishment in Cognitive Radio Networks Using Channel Hopping,” *IEEE J. Selected Areas in Comm.,* vol. 29, no. 4, pp. 689-703, Apr. 2011.

[22] D. Yang, J. Shin, and C. Kim, “Deterministic Rendezvous Scheme in Multichannel Access Networks,” *Electronics Letters,* vol. 46, no. 20, pp. 1402-1404, 2010.

[23] P. Bahl, R. Chandra, and J. Dunagan, “SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad Hoc Wireless Networks,” *Proc. ACM MobiCom,* pp. 216-230, 2004.

[24] H.W. So, J. Walrand, and J. Mo, “McMAC: A Multi-Channel MAC Proposal for Ad Hoc Wireless Networks,” *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC),* pp. 334-339, 2007.

[25] I.F. Akyildiz, W.-Y. Lee, M.C. Vuran, and S. Mohanty, “Next Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Networks: A Survey,” *Computer Networks J.,* vol. 50, no. 13, pp. 2127-2159, 2006.

[26] H. Kim and K.G. Shin, “Efficient Discovery of Spectrum Opportunities with MAC-Layer Sensing in Cognitive Radio Networks,” *IEEE Trans. Mobile Computing,* vol. 7, no. 5, pp. 533-545, May 2008.

[27] “Cognitive Radio Cognitive Network Simulator,” http://stuweb.ee.mtu.edu/~ljialian/, 2013.

Guey-Yun Chang received the BS and MS degrees from the National Taiwan Normal University in 1995 and 1998, respectively, and the PhD degree in computer science and information engineering from National Taiwan University in 2005. Currently, she is working as an associate professor in the Department of Computer Science and Information Engineering, National Central University. Her research interests include cognitive radio networks, wireless sensor networks, combinatorial optimization, and system-level diagnosis.

Wen-Hung Teng received the BS degree in computer science from Tankung University, Taiwan, and the MS degree in computer science from National Central University, Taoyuan, Taiwan, in 2011. His research interests include wireless sensor networks and cognitive radio networks.

Hao-Yu Chen received the BS degree in computer and communication engineering from National Kaohsiung First University of Science and Technology, Taiwan, and the MS degree in computer science and information engineering from National Central University, Taiwan, in 2012. Currently, he is fulfilling his mandatory military service. His research interests include wireless sensor networks, cognitive radio networks, and social networks.

Jang-Ping Sheu received the BS degree in computer science from Tamkang University, Taiwan, Republic of China, in 1981, and the MS and PhD degrees in computer science from National Tsing Hua University, Taiwan, Republic of China, in 1983 and 1987, respectively. He is currently a chair professor in the Department of Computer Science, National Tsing Hua University. He was a chair of the Department of Computer Science and Information Engineering, National Central University, from 1997 to 1999. He was the director of the Computer Center, National Central University, from 2003 to 2006. His current research interests include wireless communications and mobile computing. He was previously an associate editor of the *IEEE Transactions on Parallel and Distributed Systems* and is currently an associate editor of the *International Journal of Ad Hoc and Ubiquitous Computing* and *International Journal of Sensor Networks*. He received Distinguished Research Awards from the National Science Council of the Republic of China in 1993-1994, 1995-1996, and 1997-1998, the Distinguished Engineering Professor Award from the Chinese Institute of Engineers in 2003, the K.-T. Li Research Breakthrough Award of the Institute of Information and Computing Machinery in 2007, and the Y.Z. Hsu Scientific Chair Professor Award in 2009. He is a fellow of the IEEE and a member of the ACM and Phi Tau Phi Society.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.