Sequence-Based Channel Hopping Algorithms for Dynamic Spectrum Sharing in Cognitive Radio Networks

Prasan Kumar Sahoo, *Member, IEEE*, and Debasish Sahoo

***Abstract*— Cognitive radio network (CRN) is a promising solution to spectrum scarcity that uses the dynamic spectrum access mechanism to increase the efficiency of the underutilized licensed spectrum. In a CRN, a pair of users exchanges their information at a common unused licensed channel to rendezvous. The rendezvous in all available channels and within the bounded time cycle is a challenging issue in CRNs. In this paper, the primary idea is to construct the channel hopping sequences by using primitive roots of the prime number. For guaranteed rendezvous in CRNs, we design three channel hopping protocols for the symmetric and asymmetric environment in synchronous and asynchronous scenarios of the CRN. Extensive simulation is performed to analyze the throughput, maximum time to rendezvous (MTTR), and average time to rendezvous (ATTR). Simulation results show that our protocols can outperform over the existing protocols and can give significant improvements in terms of MTTR, ATTR, and throughput.**

***Index Terms*— Cognitive radio networks, spectrum sharing, channel hopping, rendezvous.**

I. Introduction

COGNITIVE radio networks (CRNs) employ new communication paradigms in more intelligent and flexible ways, which are different from those of the conventional wireless networks. In conventional wireless networks, a portion of the licensed spectrum is under utilized, whereas the unlicensed spectrum sharing is increased significantly by various wireless devices. Cognitive Radio (CR) technology uses the dynamic spectrum access (DSA) [1] mechanism that improves the efficiency of the under utilized licensed spectrum. The licensed spectrum is used by the primary users (PUs) and DSA allows the unlicensed secondary users (SUs) to share the vacant portion of the licensed spectrum. In CRN, a pair of SUs needs to sense the unused licensed spectrum and proceeds for the rendezvous process to achieve a successful communication. The typical licensed spectrum is divided into channels, which are used for the rendezvous process. During the rendezvous process, the pair of SUs meets at a common channel to exchange the control information such as available channel status, link quality etc. However, rendezvous process is a challenging issue as SUs do not know each other in advance and they do not have idea about the common available channels.

Several approaches are found on the basis of common control channel and channel hopping mechanisms in order to mitigate the problems of the rendezvous process. Common Control Channel (CCC) [2], [3] is a centralized approach in which a single or multiple channels are being used for exchanging the control information and other remaining channels are used for the data communication purpose. However, maintaining single common control channel is a weaker solution in comparison to that of the multiple common control channels as the starvation problem may occur on a particular channel due to the continuous presence of the PU on that single channel. Though multiple common control channels can be used to mitigate the starvation problem, it increases the network overhead to know the availability of the control channels and to pass the control information to each SUs in the network. Again multiple common control channels reduce the availability of the data channels. However, the Channel Hopping (CH) approach overcomes many problems of CCC such as long time blocking by PUs and control channel saturation problem. But, it has also many limitations and challenges. The primary challenge in the CH approach is how to increase the degree of rendezvous by minimizing the Maximum Time To Rendezvous (MTTR) between the SUs. Rendezvous is defined as the number of distinct overlapping of channels in different hopping slots between two channel hopping sequences within a channel hopping period. Let, $\{c_0, c_1, \cdots, c_{M-1}\}$ be set of total M number of licensed channels and t number of hopping slots are available in the CRN within a channel hopping period. For a CH sequence $\mathcal{S} = \{(0, s[0]), (1, s[1]), \cdots, (i, s[i]), \cdots, (t-1, s[t-1])\}$, $s[i] \in \{c_0, c_1, \cdots, c_{M-1}\}$ is the channel assigned to the i-th slot, where $i \in [0, t-1]$ is the index of a hopping slot. If $(i, j) \in \mathcal{S}_s \cap \mathcal{S}_r$ between two CH sequences $\mathcal{S}_s$ and $\mathcal{S}_r$, (i, j) is called a rendezvous, where i is the rendezvous slot and j is the rendezvous channel. If R is the set of rendezvous channels between CH sequences $\mathcal{S}_s$ and $\mathcal{S}_r$, then $|R|$ is the number of rendezvous channels and minimum value of $|R|$ among all CH sequences

Manuscript received May 7, 2016; revised August 13, 2016; accepted September 22, 2016. Date of publication October 4, 2016; date of current version November 3, 2016. This work was supported by the Ministry of Science and Technology, Taiwan, under Grant 104-2221-E-182-004 and Grant 105-2221-E-182-050.

P. K. Sahoo is with the Department of Computer Science and Information Engineering, Chang Gung University, Taoyuan 33302, Taiwan, and also with the Department of Cardiology, Chang Gung Memorial Hospital, Taoyuan 333, Taiwan (e-mail: pksahoo@mail.cgu.edu.tw).

D. Sahoo is with the Department of Computer Science and Information Engineering, Chang Gung University, Taoyuan 33302, Taiwan (e-mail: d0221012@stmail.cgu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2016.2615258

is the degree of rendezvous. CH protocols in CRN can be classified into symmetric or asymmetric models based on the availability of channels for the SUs when they coexist with the PUs. In symmetric CH approach, each SU can have same set of available channels located in the same geographical area, whereas in asymmetric CH approach, each SU can have different sets of available channels with at least one common channel among them. If C_i and C_j are the sets of available channels for the SUs i and j, respectively, then $C_i = C_j$ in symmetric models and $C_i \neq C_j$ with at least one common available channel in asymmetric models for each $1 \leq i, j \leq L$, when $L \geq 2$ number of SUs coexist in the CRN.

A number of CH algorithms such as Generated Orthogonal Sequence (GOS) [4], [5], Modular Clock (MC) [5], Modified Modular Clock (MMC) [5], Deterministic Rendezvous Sequence (DRSEQ) [6], First Rendezvous CH (FRCH) [7], Cycle Adjustable CH (CACH) [8], Jump-Stay (JS) [9], Efficient Alternate Hop-and-Wait (E-AHW) [10], Asynchronous CH (ACH) [11], Asynchronous Rendezvous CH (ARCH) [12], Symmetric Asynchronous Rendezvous CH (SARCH) [12], Load based Quorum CH system (L-QCH) [13], and Moving Traversing Pointers (MTP) [14] schemes are proposed in the past. CH methods are specifically considered as a distributed approach for the rendezvous process. It is a promising solution for the issue of rendezvous process, which does not need any dedicated common control channel. Rendezvous process in such distributed approach is referred to as *blind rendezvous* process. In blind rendezvous process, CH is done without any prior knowledge of the CRNs except the number of total channels. In common rendezvous process, each SU hops on the basis of the predefined channel hopping patterns according to each time slot. Once a pair of SUs hops on a common available channel at a specific time instance, they can rendezvous to exchange the control information. Here, common available channels are referred to as *rendezvous channels*. CH patterns are mainly of two different types: Sequence based [4], [6], [7] and Randomized based [9], [10] CH patterns. Though randomized CH patterns are popular in the literature, there are some limitations. In the randomized CH pattern, overlapping of two CH patterns is not regular and these approaches need more time in comparison to our proposed rendezvous algorithms for guaranteed rendezvous. On the other hand, sequence based CH patterns are regular, but rendezvous is not guaranteed for all possible sequences and in all rendezvous channels. Hence, we propose here a deterministic sequence based channel hopping approach in order to solve the issues of the rendezvous process.

A. Motivation and Contributions

Normally, CH protocols in CRN can be categorized as Symmetric Synchronous [12], [13], Symmetric Asynchronous [4], [5], [7], [9], [12], Asymmetric Synchronous and Asymmetric Asynchronous [10], [11], [14]. We are motivated to design the novel deterministic sequence based channel hopping algorithms for the symmetric synchronous, symmetric asynchronous, and asymmetric asynchronous CRN in which every pair of SUs can have the guaranteed rendezvous within a finite period of time. In our approach, any pair of SUs can rendezvous in each and every available channels within a finite duration of time so that no channel is left. All rendezvous channels must be used fairly, i.e., all rendezvous channels have equal probability to be used as common control channel. Accordingly, our goal is to design a set of CH algorithms for different scenarios in CRN depending on the number of available channels and presence of the PUs. We generate the CH sequences using the concept of primitive root of the prime number P and major contributions of our work can be listed as follows:

- We provide efficient sequence based CH mechanisms those utilize the primitive root as the generator of the sequences.
- A Symmetric Synchronous (SSync) CH protocol is designed to increase the degree of rendezvous. Optimality of SSync is considered in terms of Average Time To Rendezvous (ATTR) and Maximum Time To Rendezvous (MTTR). The value of MTTR and ATTR for our proposed SSync protocol is $\frac{N}{k}$ and the expected time to rendezvous on all N number of available channels is also $\frac{N}{k}$. The Expected Inter Rendezvous Interval (EIRI) on all N available channels is $(\frac{N}{k} - 1)$.
- A Symmetric Asynchronous (SAsync) CH protocol is designed, where SUs are located in the same geographical region follow the symmetric model in an asynchronous CR environment. The optimality of SAsync protocol in terms of ATTR is $\frac{N}{2}$ and MTTR is $\frac{N}{k}$. Besides, SAsync protocol is significantly better than SSync in terms of ETTR and EIRI, which is analyzed theoretically.
- An Asymmetric Asynchronous (AAsync) CH protocol is designed, which is applicable in more general environment of CRN, where there is no global clock synchronization and pair of SUs cannot have same set of available channels. Our AAsync protocol achieves MTTR as M^2 and ATTR as $\frac{M \times P}{2}$, where M is the number of total channels and degree of rendezvous is equal to at least the number of rendezvous channels.

The rest of the paper is organized as follows. Related works are discussed in Section II. System models of our proposed protocols are mentioned in Section III. CH sequences based on the primitive roots are generated in Section IV. The Channel Hopping rendezvous algorithms for Symmetric Synchronous and Asynchronous models are designed in Section V and Section VI, respectively. The Channel Hopping rendezvous algorithm for Asymmetric Asynchronous model is presented in Section VII. Simulation results are presented in Section VIII and concluding remarks are made in Section IX.

II. Related Work

In this section, the latest works on dynamic spectrum access strategies [15], [16] in multichannel CRN are analyzed for the coexistence of licensed PUs and unlicensed SUs. Depending on the multichannel communication process including the control message exchange, the existing communication rendezvous mechanisms in multichannel CR networks can be classified into two categories as centralized and decentralized CRN. The centralized system needs a centralized controller such as a server or base station, which manages other users of the network in the rendezvous process. The management of nodes in CRN is easy as channel information is known in advance in a centralized system. Base station and single server approaches are centralized systems used for the cognitive wireless random access network such as TV Band [17]. In the centralized system, common control channel establishment is essential in the rendezvous process, which can be established in single or multiple channels and other remaining channels are considered as data channels. Centralized system faces control channel congestion problem, which leads to under utilization of the channels. Besides, multiple common control channels (CCC) also do not eliminate the control channel congestion problem.

Distributed CRN can include the case of single or multiple CCC or without having any CCC. This system is popular as no dedicated common control channel is required and any channel can be used as a control or data channel. Though distributed CRN can mitigate the problems of centralized approach, designing such system is a challenging issue as no information of rendezvous process is known in prior, which is called blind rendezvous. For common control channel establishment in distributed system, a quorum based approach L-QCH [13] is considered. In L-QCH, all SUs use Quorum-based CH (QCH) approach to establish multiple common control channels for the rendezvous process in synchronous environment. Quorum-based Channel Hopping (QCH) system use the intersection property of quorum systems to generate the CH sequences those enable the rendezvous on multiple channels between any two CH sequences. Chao *et al.* [18] propose a synchronous distributed CRN channel hopping protocol called Quorum and Latin square Channel Hopping (QLCH). In order to guarantee the rendezvous, it utilizes the property of quorum systems and latin squares. In our survey of the related literature, synchronous and asynchronous protocols are considered for the distributed system to compare with our proposed rendezvous algorithms. The recent study includes a distributed cyclic approach based CH mechanism [12], where two different users consider a sequence of channels and hop in opposite order of rotation so that they can meet at a common channel.

Chang *et al.* [12] propose RCCH (Rendezvous Couple Channel Hopping) for synchronous environment and two asynchronous algorithms such as ARCH (Asynchronous Rendezvous Channel Hopping) and SARCH (Symmetric Asynchronous Rendezvous Channel Hopping) to increase the degree of rendezvous. Though RCCH scheme can increase the channel utilization ratio, degree of rendezvous is only N and value of MTTR is $\frac{N}{2}$, which is very large, where N is the number of available channels. Although, rendezvous is possible between the sender and the receiver in ARCH, both need distinct time-parity and probability of rendezvous is only $\frac{1}{N}$. In DRSEQ, Yang *et al.* [6] consider an asynchronous CH in which each SU follows a CH sequence formed by its inverted sequence separated by an empty time slot. Liu *et al.* [19] analyze the performance metrics for channel access delay of CRNs by considering an asynchronous protocol GOS [4], where channel access delay is considered with help of channel availability condition and asynchronous CH rendezvous schemes. Voice over IP (VoIP) communication based two tier CRN model is proposed in [20], where PUs and SUs transmit data with different probabilities to improve the spectrum utilization. However, the work does not speak how rendezvous can occur between the SUs in absence of the PUs and there is no theoretical analysis to justify the improvement in spectrum utilization.

Another symmetric asynchronous algorithm FRCH [7] is proposed, where channel hopping sequence is similar to that of DRSEQ. The CH sequence is formed by concatenating a sequence with its inverted sequence and is followed by the initial channel number. FRCH uses asynchronous environment, where number of available channels may be same or different for each user. However, it suffers from under utilization of available channels, where a user can only rendezvous with another one at some particular available channels during the channel hopping. Ghorbel *et al.* [21] propose a two phase heuristics algorithm to allocate the spectrum and power resources among the users. Though authors propose a joint dynamic multi channel spectrum access with adaptive power allocation, the work does not analyze the percentage of channel utilization for variable traffic rates of the PUs and SUs. The randomized channel hopping approach utilizes the distributed CRN systems in both synchronous and asynchronous environment, where a pair of SUs switches from one channel to another using a randomized sequence till they meet at a rendezvous channel. In JS [9], SUs use a randomized channel hopping sequence that consists of a jump and a stay pattern. SUs jump on the available channels in the jump-pattern and stay on a particular channel in the stay-pattern. A jump-pattern consists of $2P$ time slots, whereas stay-pattern considers P number of time slots, where P is a smallest prime number larger than the number of available channels. The hopping patterns in JS [9] for a pair of SUs are different and do not follow any cyclic pattern, for which each channel cannot be visited uniformly during the CH period. Moreover, the JS pattern is not worthwhile for the symmetric synchronous environment as it uses random seeds to generate different stay-patterns for different users corresponding to each cycle. Besides, the MTTR in JS is at most $3P$. Later Lin *et al.* [22] extend the previous work by considering different cycle length for jump-pattern.

Chang *et al.* [23] propose two channel hopping algorithms T-CH and D-CH. In T-CH, though degree of rendezvous is increased, there is higher chance of collision in multiuser scenario as the sender and the receive use the same CH sequence. In D-CH, degree of rendezvous is very small with large value of MTTR, if both SUs have different IDs. Monemi *et al.* [24] analyze the feasible interference region for the PUs caused by the SUs in a CRN and do not discuss about the spectrum access problems and thereby do not analyze the percentage of spectrum usage and degree of rendezvous. Zhang *et al.* [25] propose a heuristic greedy algorithm for rendezvous channel assignment to form predefined sequences. Based on the predefined sequences, they consider two different sets of channel hopping sequences for synchronous and asynchronous environments. Each user selects the sequences randomly to achieve the rendezvous diversity. In order to increase the rendezvous diversity, recently some works [26], [27] are proposed in which CRN devices are equipped with multiple radios. In [26], they utilize more than one radios to reduce the ATTR for rendezvous channels. Chen *et al.* [27] propose an infrastructure based CRN with centralized base station (BS) and cognitive users, where only BS is equipped with multiple radios in order to deliver the broadcast content to its users. Since, multi-radio based CRN requires more resources, we are of the view that designing CH sequences with single radio based CRN to have same or more number of degree of rendezvous is economical. Based on the survey of the latest literature on CH protocols, we propose here the sequence based CH algorithms for the synchronous, asynchronous, symmetric and asymmetric environment of the CRN as discussed below.

III. Problem Formulation

A. Preliminaries

Consider a Cognitive Radio Network (CRN) that consists of $L \geq 2$ number of secondary users (SUs) equipped with half duplex radios for spectrum sensing, sending or receiving the control message or data and coexist with the primary users (PUs) over a common geographical region. The PUs use the licensed spectrum that is divided into total M number of orthogonal channels $\mathcal{C} = \{c_1, c_2, \cdots, c_M\}$, where c_i denotes the i-th channel. A channel is said to be available to an SU, if it can communicate on the channel without causing any interference to the PUs. Using appropriate sensing model, user SU_i, for $i \in [1, L]$ can be able to find the available channels $C_i \subseteq \mathcal{C}$ before the rendezvous process and after some period of time that consists of several cycles of channel hopping. We consider a symmetric model in which all users share the same number of available channels located in the same geographical region. Accordingly, for any two users SU_i and SU_j, $C_i = C_j \subset \mathcal{C}$ with $1 \leq i, j \leq L$, where N is the number of available channels in the CRN. Besides, we consider an asymmetric model in which SUs can have different sets of available channels such that at least one common channel exists between one pair of SUs in the network. The set of such common channel is $G_{ij} = C_i \cap C_j$ and number of common channels $G = |G_{ij}|$.

It is assumed that the CRN is time slotted with each slot is of equal duration. In each time slot, each user hops on a channel based on its transmitting or receiving mode and scans a channel based on the respective sequences to attempt the rendezvous with its potential neighbors. We consider channel hopping (CH) algorithms for both symmetric and asymmetric models with or without time synchronization. Any cognitive radio device is assumed to be capable of hopping between different channels according to a channel hopping sequence and its local clock. A packet can be exchanged between two users if they hop onto the same channel in the same time slot and it is assumed that one time slot is long enough to exchange multiple packets in order to have successful rendezvous. If multiple nodes happen to rendezvous in the same time slot on the same channel, they can follow a channel contention procedure. If a user wants to send the control information to its neighbors, it finds a common rendezvous channel during the channel hopping process.

g	g^2	g^3	g^4	g^5	g^6	g^7	g^8	g^9	g^{10}	g^{11}	g^{12}
1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	3	6	12	11	9	5	10	7	1
3	9	1	3	9	1	3	9	1	3	9	1
4	3	12	9	10	1	4	3	12	9	10	1
5	12	8	1	5	12	8	1	5	12	8	1
6	10	8	9	2	12	7	3	5	4	11	1
7	10	5	9	11	12	6	3	8	4	2	1
8	12	5	1	8	12	5	1	8	12	5	1
9	3	1	9	3	1	9	3	1	9	3	1
10	9	12	3	4	1	10	9	12	3	4	1
11	4	5	3	7	12	2	9	8	10	6	1
12	1	12	1	12	1	12	1	12	1	12	1

Fig. 1. Powers of integers, modulo 13. Primitive roots modulo 13 are 2, 6, 7, and 11.

B. Primitive Root

Let P be a prime number. A primitive root (or generator) $g \in [1, P)$ is an integer such that when x goes from 1 through $P-1$, $g^x \; mod \; P$ goes through all the integers $1, 2, \cdots, (P-1)$ in some order. Powers of integers modulo prime number P is given in Fig. 1, where $P = 13$. The shaded gray color represents the distinct ordered elements of integers modulo 13. For each $g \in \{2, 6, 7, 11\}$, it is clearly observed that $g^x \; mod \; 13$ for each $x \in [1, 12]$ computes all the distinct elements those belong to the interval [1, 12] in certain order. Hence, 2, 6, 7, and 11 are primitive roots modulo 13. In another case, for each $g \in \{1, 3, 4, 5, 8, 9, 10, 12\}$, it is found that only few (less than 12) elements those belong to the interval [1, 12] are repeated in a cyclic manner.

In our CH system, we consider two distinct primitive roots for generating two different sequences corresponding to each and every SUs. For generating two sequences, we need at least two primitive roots of prime number P and to have two distinct primitive roots, prime number P must be at least 5. The significance of choosing primitive root of a prime number P is that for any pair of sequences formed by these primitive roots, degree of overlapping (k) between any pair of sequences is at least 2 and the value of k is always an even number depending on the number of available channels N in symmetric models, where $N = P-1$ or number of total channels M in asymmetric model, where $M = P - 1$ and $P \geq 5$. It is to be noted that N or M is a whole number >1 and either of this value can be taken to generate the sequences.

Proposition 1: If g_i is a primitive root of prime number P of order N and $g_i \in \mathbb{Z}_P^*$ for $i = [1, \ell]$, then $(g_i)^x \; mod \; P$ for $x \in [1, N]$ are all distinct, where $N = \Phi(P)$ and $\mathbb{Z}_P^* = \{1, 2, \cdots, P-1\}$ is a cyclic group of order $(P-1)$.[1]

Proposition 2: If g_i is primitive root of order N and $g_i \in \mathbb{Z}_P^*$ for $i = [1, \ell]$, then the set of all primitive roots $\mathcal{R}$ has ℓ number of elements of order N, where $N = \Phi(P)$ and $\ell = \Phi(N)$.[2]

It is to be noted that Galois field also can be used to construct the CH sequences instead of primitive root. However, the Galois field as used in CACH [8] for symmetric synchronous environment can construct the CH sequence by

[1]For a prime number P, Euler's quotient function $\Phi(P) = P - 1$.

[2]For any $N = P_1^{r1} P_2^{r2} \cdots P_m^{rm}$, where P_i denotes a prime number. Euler's quotient function $\Phi(N) = N \cdot (1 - \frac{1}{P_1}) \cdot (1 - \frac{1}{P_2}) \cdots (1 - \frac{1}{P_m})$.

which Maximum Conditional Time To Rendezvous (MCTTR) is $(N+1)M$ in CACH, where N and M are the number of available and total channels in the CRN, respectively and $N < M$. To the best of our knowledge, CACH considers all channels and logically replaces the unavailable channels with available one. We use primitive roots to construct the CH sequence. In our symmetric synchronous model MCTTR is N^2/k, where k is at least 2.

C. Generation of Sequences

Let $\mathcal{R}$ be the set of ℓ number of primitive roots of a prime number $P \geq 5$, where $\mathcal{R} = \{g_1, g_2, \cdots, g_\ell\}$. Corresponding to each primitive root $g_i \in \mathcal{R}$, we can find a sequence $\langle g_i \rangle_N$ by evaluating $g_i^x \ mod \ P$ sequentially with all $x \in [1, P)$. The sequences formed with help of these primitive roots can have a unique property such that pair of ordered sequences have N number of elements and can have the common overlapping at k number of positions, where $N = P-1$ and $k \geq 2$. Due to this property, we consider the primitive root as the generator of sequences. In the following subsection, we introduce the generation of two different types of sequences namely *Default* and *Elementary* based on these primitive roots of a prime number P. The *Default* sequence is used for generating the receiver's channel hopping sequence and *Elementary* sequence can be used for generating the sender's channel hopping sequence, which is discussed in Section IV.

1) Default Sequence: Default sequence S_d is generated with help of the generator $g_\ell \in \mathcal{R}$ corresponding to a prime number P. We can find a default sequence from the ordered elements formed by evaluating $(g_\ell)^x \ mod \ P$ for each x from $1, 2, \cdots, (P-1)$. Thus, the formed default sequence is represented as $S_d = \langle g_\ell \rangle_N$, which consists of N number of ordered elements, where $N = P-1$.

For example, if $P = 13$ and $N = 12$, set of primitive roots of P is $\mathcal{R} = \{g_1, g_2, g_3, g_4\} = \{2, 6, 7, 11\}$. Default sequence $\langle g_\ell \rangle_N$ can be generated by g_ℓ, where $g_\ell = g_4 = 11$. In order to find the default sequence, we evaluate $(g_4)^x \ mod \ P$ for each x from $1, 2, \cdots, 12$. Thus, default sequence $S_d = \langle g_4 \rangle_N = \langle 11 \rangle_{12} = \langle 11, 4, 5, 3, 7, 12, 2, 9, 8, 10, 6, 1 \rangle$.

2) Elementary Sequences: If we choose a primitive root $g_i \in \mathcal{R} \setminus \{g_\ell\}$; $i \in [1, \ell-1]$ corresponding to a prime number P, we can find an elementary sequence S_e^i formed by evaluating $g_i^x \ mod \ P$ for each x from $1, 2, \cdots, (P-1)$. Thus, the elementary sequence formed is represented as $S_e^i = \langle g_i \rangle_N$, which consists of N number of ordered elements, where $N = P-1$ and $i \in [1, \ell-1]$.

For example, if $P = 13$ and $N = 12$, set of primitive roots of P is $\mathcal{R} = \{g_1, g_2, g_3, g_4\} = \{2, 6, 7, 11\}$. The elementary sequences can be generated like default sequence by considering the primitive roots $g_i \in \mathcal{R} \setminus \{g_\ell\} = \{g_1, g_2, g_3\} = \{2, 6, 7\}$. Elementary sequences $\langle g_i \rangle_N$ are $\langle 2 \rangle_{12}, \langle 6 \rangle_{12}$, and $\langle 7 \rangle_{12}$. Here, $S_e^1 = \langle 2 \rangle_{12} = \langle 2, \mathbf{4}, 8, \mathbf{3}, 6, \mathbf{12}, 11, \mathbf{9}, 5, \mathbf{10}, 7, \mathbf{1} \rangle$, $S_e^2 = \langle 6 \rangle_{12} = \langle 6, 10, 8, 9, 2, \mathbf{12}, 7, 3, 5, 4, 11, \mathbf{1} \rangle$, $S_e^3 = \langle 7 \rangle_{12} = \langle 7, 10, \mathbf{5}, 9, 11, \mathbf{12}, 6, 3, \mathbf{8}, 4, 2, \mathbf{1} \rangle$.

D. Degree of Overlapping

Degree of overlapping (k) of default sequence with elementary sequences is defined as the number of common elements present in these pair of sequences at a particular ordered position. Channel hopping sequence can be derived from the *default* sequence and *elementary* sequences. We use degree of overlapping as a key feature for generating receiver's CH sequence and sender's CH sequence from the respective Default sequence and Elementary sequences. Degree of overlapping between default sequence S_d with elementary sequence S_e^i is denoted as $k_i = \mathcal{D}(S_d, S_e^i)$; $\forall i \in [1, \ell-1]$. Set of degree of overlapping k_i of default sequence with elementary sequences can be denoted as $\mathcal{K}$. As shown in Fig. 2, degrees of overlapping of default sequence $S = \langle 11 \rangle_{12}$ with elementary sequences $S_1 = \langle 2 \rangle_{12}, S_2 = \langle 6 \rangle_{12}$, and $S_3 = \langle 7 \rangle_{12}$ are $k_1 = 6, k_2 = 2$, and $k_3 = 4$, respectively. Here, $\mathcal{K} = \{2, 4, 6\}$.

Sequence Type	Sequence Notation	Sequence												Degree of Overlapping
Default (S_d)	$\langle g_4 \rangle_N = \langle 11 \rangle_{12}$	11	4	5	3	7	12	2	9	8	10	6	1	
Elementary ($S_e{}^1$)	$\langle g_1 \rangle_N = \langle 2 \rangle_{12}$	2	4	8	3	6	12	11	9	5	10	7	1	$k_1 = 6$
Elementary ($S_e{}^2$)	$\langle g_2 \rangle_N = \langle 6 \rangle_{12}$	6	10	8	9	2	12	7	3	5	4	11	1	$k_2 = 2$
Elementary ($S_e{}^3$)	$\langle g_3 \rangle_N = \langle 7 \rangle_{12}$	7	10	5	9	11	12	6	3	8	4	2	1	$k_3 = 4$

Fig. 2. Sequences generated with help of primitive roots and degree of overlapping of Elementary sequences with Default sequence.

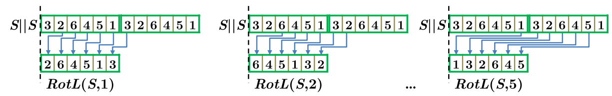


Fig. 3. Rotation of a sequence $S = \langle 3, 2, 6, 4, 5, 1 \rangle$. Rotation function is $RotL(S, x)$, where, $x \in \{1, 2, \cdots, 5\}$.

E. Rotation of Sequence: RotL(S, x)

$RotL(S, x)$ is a function, which is defined as the rotation of a sequence S through x places to the left, where $x \in \{1, 2, \cdots, |S|-1\}$. As shown in Fig. 3, we consider a sequence $S = \langle 3, 2, 6, 4, 5, 1 \rangle$ and $x \in \{1, 2, \cdots, 5\}$. In order to get the left shifts for different values of x, the rotation of S is performed by concatenating the sequence $S \| S$. Thus, $RotL(S, 1) = \langle 2, 6, 4, 5, 1, 3 \rangle$ as the sequence S has to shift left by one place. Similarly, $RotL(S, 5) = \langle 1, 3, 2, 6, 4, 5 \rangle$ as the sequence S has to shift left by five places.

F. Basic Information for Generating CH Sequence

Upon getting the degree of overlapping between the default sequence with each elementary sequences, the maximum degree of overlapping between the default and elementary sequences is calculated. This information of maximum degree of overlapping is used to find the sender and receiver's CH sequence, which is known as the basic information for the CH sequence generation. It is to be noted that the degree of overlapping of default sequence with elementary sequences is first calculated as shown in Fig. 2 before getting the basic information for CH sequence generation. Then, the maximum degree of overlapping (k) is calculated among all degrees of overlapping as shown in Fig. 2 and the corresponding default sequence (S_d) and elementary sequences (S_e) are used for generating the receiver's and sender's CH sequences, respectively. Thus, the key information required for generating such CH sequences are (S_d, S_e, k), where S_d is the default sequence, elementary sequence $(S_e) = \{S_e^i | k_i = k,\ i \in [1, \ell - 1]\}$ and $k = \{Max(k_i) | k_i \in \mathcal{K},\ i \in [1, \ell - 1]\}$.

Let $\{1, 2, \cdots, 12\}$ be the set of channels available for the SUs in the CRN, where $N = 12$ and prime number $P = 13 > N$. In this case, the set of primitive roots $\mathcal{R} = \{g_1, g_2, g_3, g_4\} = \{2, 6, 7, 11\}$ and the sequences corresponding to each primitive roots are $\langle 2 \rangle_{12}, \langle 6 \rangle_{12}, \langle 7 \rangle_{12}$, and $\langle 11 \rangle_{12}$. As shown in Fig. 2, the degree of overlapping of elementary sequence $\langle 2 \rangle_{12}$ with default sequence $\langle 11 \rangle_{12}$ is maximum. Hence, the elementary sequence $S_e = \langle 2 \rangle_{12}$ corresponding to the maximum degree of overlapping is selected as the elementary sequence for generating the CH sequence along with the default sequence $S_d = \langle 11 \rangle_{12}$. Since, the maximum degree of overlapping $k = Max(k_i) = 6,\ k_i \in \mathcal{K} = \{2, 4, 6\}$, corresponding S_d and S_e are $\langle 11 \rangle_{12}$ and $\langle 2 \rangle_{12}$, respectively, the basic information for CH sequence generation for each SU is considered as a triple $(S_d, S_e, k) = (\langle 11 \rangle_{12}, \langle 2 \rangle_{12}, 6)$.

IV. CH Sequence Generation

An SU can generate the channel hopping sequences by using the basic information of CH sequence generation and based on its role as a sender or receiver. The sender's Symmetric Synchronous Channel Hopping (SSCH) sequence can be generated by using the elementary sequence S_e and maximum degree of overlapping k as given in Algorithm 1. The default sequence S_d and maximum degree of overlapping k are used to generate the receiver's Symmetric Channel Hopping (SCH) sequence irrespective of the synchronous or asynchronous environment of the receiver as given in Algorithm 2. Again, receiver's Asymmetric Channel Hopping (ACH) sequence can be generated by using Algorithm 3 irrespective of the synchronous or asynchronous environment of the receiver. Besides, sender's Symmetric or Asymmetric Channel Hopping sequence for the asynchronous environment can be generated by using the basic information (S_e, k) of CH as presented in Algorithm 5 and Algorithm 6, respectively. Generation of sender and receiver's CH sequence for symmetric and asymmetric environment in synchronous or asynchronous scenario is described in the following subsections.

It is to be noted that two SUs may have data to send to each other and therefore choose the same elementary sequence. In order to mitigate this problem, each SU goes for the handshaking procedure in the first slot of the control channel before choosing the default or elementary sequence. In this slot, each SU goes for a random backoff and senses the channel to find the presence of other SUs in that channel. If an SU does not sense others in its vicinity, it transmits a Ready To Send (RTS) message with information about its destination node and data size followed by another period of silence. Then, it waits for a response from the destination node. If the destination node has also data to send to the same sender, it changes its role to a receiver, uses the default sequence S_d to generate its CH sequence and sends the Clear To Send (CTS) message to that sender SU. It is assumed that switching of role between the two SUs is negotiable based on the priority of the data. Upon receiving the CTS message, the sender first establishes a link with its destination node and uses the elementary sequence S_e to generate its CH sequence for sending data in the next hopping slots and channels.

Algorithm 1 Sender's Symmetric Synchronous CH (SSCH) Sequence

Require: Sender $\text{SU}_j,\ j \in [1, L]$, number of available channels N, prime number P, elementary sequence $S_e = \langle c_1^s, c_2^s, \cdots c_N^s \rangle$ and maximum degree of overlapping k;
Ensure: CH Sequence $\mathcal{S}_s^j$ of sender SU_j;

1: $\mathcal{S}_s^j = \emptyset$;
2: Choose random integer $\beta \in [0, \frac{N}{k})$;
3: $S^j = RotL(S_e, \beta \times k)$;
4: $cycle = 0$;
5: **while** in transmit mode **do**
6: $\quad cycle = cycle + 1$;
7: $\quad i = [(cycle - 1)\ mod\ \frac{N}{k}] + 1$;
8: $\quad S = RotL(S^j, (i - 1) \times k)$;
9: $\quad \mathcal{S}_s^j = \mathcal{S}_s^j \| S$;
10: **end while**
11: **return** $\mathcal{S}_s^j$;

A. Sender's Symmetric Synchronous CH (SSCH) Sequence

Let us assume that a secondary user SU_j has data to send for which it switches to the transmitting mode and uses the sender's CH sequence. It needs to hop in such a way that it can rendezvous at a slot in a common channel with the receiver. In symmetric synchronous environment, it is assumed that the cognitive devices are already synchronized and each SU is operated on N number of available channels, where $N = P - 1$. Sender $\text{SU}_j,\ j \in [1, L]$ uses the elementary sequence S_e based on the maximum degree of overlapping k and generates the sequence S as mentioned in Algorithm 1 from line 2 through line 3. In each cycle-i, sequence S^j is generated from the sequence S as mentioned in line 8 of Algorithm 1. To find the Symmetric Synchronous CH (SSCH) sequence $\mathcal{S}_s^j$ of sender SU_j, the sequence S^j can be concatenated at least $\frac{N}{k}$ times for $\frac{N}{k}$ number of cycles as presented in line 8 through line 9 of Algorithm 1. This can meet the property of rendezvous, which suggests that sender's CH sequence can be able to rendezvous with all N number of available channels after $\frac{N}{k}$ number of cycles. For example, for $j = 1$, let SU_1 be the sender that executes the SSCH algorithm. It has to use the elementary sequence $S_e = \langle 2 \rangle_{12} = \langle 2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7, 1 \rangle$ and maximum degree of overlapping $k = 6$ as its inputs based on the conditions given in Section III-F. Assuming $\beta = 0 \in [0, 2)$, sender SU_1 forms the sequence $S^1 = S_e$ during the execution of Algorithm 1. For cycle-1, $i = 1$ and $S = RotL(S^1, 0)$, i.e., $S = S^1$. Sender sequence $\mathcal{S}_s^1 = \mathcal{S}_s^1 \| S = \langle 2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7, 1 \rangle$. Again in cycle-2, $i = 2$ and $S = RotL(S^1, 6)$, i.e. $S = \langle 11, 9, 5, 10, 7, 1, 2, 4, 8, 3, 6, 12 \rangle$. Sender sequence becomes $\mathcal{S}_s^1 = \mathcal{S}_s^1 \| S = \langle 2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7, 1, 11, 9, 5, 10, 7, 1, 2, 4, 8, 3, 6, 12 \rangle$, which continues for other cycles too.

Algorithm 2 Receiver's Symmetric CH (SCH) Sequence

Require: Default sequence $S_d = \langle c_1^r, c_2^r, \cdots c_N^r \rangle$ and maximum degree of overlapping k;
Ensure: Receiver's CH Sequence $\mathcal{S}_r$;
1: $\mathcal{S}_r = \emptyset$;
2: $S = S_d$;
3: Choose random integer $\alpha \in [0, \frac{N}{k})$;
4: $S = RotL(S, \alpha \times k)$;
5: $cycle = 0$;
6: **while** in the receive mode **do**
7: $\quad cycle = cycle + 1$;
8: $\quad \mathcal{S}_r = \langle \mathcal{S}_r \| S \rangle$;
9: **end while**
10: **return** $\mathcal{S}_r$;

B. Receiver's Symmetric/Asymmetric CH Sequence

The receivers generate the CH sequences for symmetric or asymmetric environment irrespective of the synchronous or asynchronous scenario by using the default sequence S_d and maximum degree of overlapping k. A secondary user SU_j has to stay in the receiving mode until and unless it has data to transmit. For symmetric model, receiver's CH sequence $\mathcal{S}_r$ can be generated by using Algorithm 2. As given in Algorithm 2, the default sequence S_d and maximum degree of overlapping k is used to generate a sequence S, which is mentioned from line 2 through line 4. The receiver's CH sequence $\mathcal{S}_r$ can be formed by concatenating the resultant sequence S repeatedly after each cycle of N time slots when it is in the receiving mode.

In the asymmetric model, set of available channels $C_r = \{c_1, c_2, \cdots, c_N\} \subset \mathcal{C}$, where $N < M$. In Algorithm 3, default sequence S_d and maximum degree of overlapping k are used to generate a sequence S, which is mentioned from line 2 through line 6. When an SU is in receiving mode, if a channel $c_i^r \in S$ and $c_i^r \notin C_r$, then c_i^r is replaced by c using line 11. Then, the sequence S is modified to another sequence $Temp$ as mentioned in Algorithm 3 from line 9 through line 16. Consequently, receiver's Asymmetric CH (ACH) sequence $\mathcal{S}_r$ can be formed by concatenating the sequence $Temp$ derived from S and by remapping the unavailable channels with available one repeatedly in each cycle with N number of time slots, when it is in the receiving mode.

C. Channel Remapping

Channel remapping is necessary for the general case taking the number of available channels (N) and number of total channels (M) such that $N < M$ and $M = P - 1$, where P is a smallest prime number larger than N. In this scenario, the sender or receiver's CH sequence consists of M number of channels in each cycle, where $(M - N)$ number of channels are not available. These $(M - N)$ number of channels are replaced by the channels with respect to the cycle-i as $((i-1)$ $mod\ N + 1)$. For example, let set of available channels be $C_r = \{c_1, c_2, \cdots, c_8\} = \{2, 3, 4, 5, 7, 8, 9, 10\} \subset \mathcal{C}$. Here, the number of available channels $N = 8$ and the smallest prime number $P > N$ is $P = 11$. As per the CH sequence,

Algorithm 3 Receiver's Asymmetric CH (ACH) Sequence

Require: Default sequence $S_d = \langle c_1^r, c_2^r, \cdots c_M^r \rangle$, maximum degree of overlapping k, available channels set $C_r = \{c_1, c_2, \cdots, c_N\}$, $N < M$;
Ensure: Receiver's CH Sequence $\mathcal{S}_r$;
1: $\mathcal{S}_r = \emptyset$;
2: $S = S_d$;
3: $Temp = \emptyset$;
4: $cycle = 0$;
5: Choose random integer $\alpha \in [0, \frac{M}{k})$;
6: $S = RotL(S, \alpha \times k)$;
7: **while** in receive mode **do**
8: $\quad cycle = cycle + 1$;
9: $\quad$ **for** $i = 1$ to M **do**
10: $\quad\quad$ **if** $c_i^r \in S$ *and* $c_i^r \notin C_r$ **then**
11: $\quad\quad\quad c = C_r[(cycle - 1)\ mod\ N + 1]$; //by Sec. IV-C
12: $\quad\quad\quad Temp[i] = c$;
13: $\quad\quad$ **else**
14: $\quad\quad\quad Temp[i] = c_i^r$;
15: $\quad\quad$ **end if**
16: $\quad$ **end for**
17: $\quad \mathcal{S}_r = \langle \mathcal{S}_r \| Temp \rangle$;
18: **end while**
19: **return** $\mathcal{S}_r$;

each cycle must consist of $M = 10$ number of channels, where channel numbers 1 and 6 are not available. Hence, in cycle-10 both channels 1 and 6 are replaced by the channel index $c_2 = 3$.

In the following sections, we design the sequence based channel hopping rendezvous algorithms for the symmetric and asymmetric environment in synchronous and asynchronous scenario. For any pair of SUs, Symmetric Synchronous Channel Hopping (SSync) rendezvous algorithm and Symmetric Asynchronous Channel Hopping (SAsync) rendezvous algorithm are designed. Besides, the Asymmetric Asynchronous Channel Hopping (AAsync) rendezvous algorithm is proposed for the more general scenario.

V. SSync Rendezvous Algorithm

Synchronous channel hopping rendezvous process is considered when the local clocks of the sender and receiver are synchronized with each other. Besides, each SU has the knowledge about the number of available channels N as they follow the symmetric model. A user having no data to transmit maintains its role as a receiver as long as it has no data and uses the channel hopping sequence $\mathcal{S}_r$ as generated by Algorithm 2. When a sender SU_j wants to rendezvous with other users, it generates the sender's channel hopping sequence $\mathcal{S}_s^j$ as stated in Algorithm 1. During pairwise rendezvous process, sender scans the channels $c_i^j \in \mathcal{S}_s^j$ at the beginning of each time slot in order to rendezvous with receiver's sequence $\mathcal{S}_r$. The rendezvous process continues until successful rendezvous is achieved as given in Algorithm 4. In each cycle of N time slots, the number of possible rendezvous is k and the Maximum Time To Rendezvous (MTTR)

Algorithm 4 SSync Rendezvous Algorithm (For Sender)

Require: CH Sequence $\mathcal{S}_s^j$ of sender SU_j;
Ensure: Rendezvous channel index c;
$i = 1$;
while (no successful rendezvous) **do**
  Scan on channel $c_i^j \in \mathcal{S}_s^j$ to rendezvous;
  if (rendezvous success) **then**
    $c = c_i^j$;
    return c;
  end if
  $i = i + 1$;
end while

is $\frac{N}{k}$. The rendezvous process can be possible in at most N number of channels in $\frac{N}{k}$ number of cycles.

Let, SU_1 and SU_2 be the sender and receiver, respectively who operates on $N = 12$ number of channels for the rendezvous process, where $P = 13$. SU_2 executes SCH Algorithm 2 in order to form the receiver's symmetric CH sequence $\mathcal{S}_r$ and SU_1 executes symmetric synchronous CH (SSCH) Algorithm 1 in order to generate the sender's CH sequence $\mathcal{S}_s^1$. Being a receiver, SU_2 uses the default sequence $S_d = \langle 11, \mathbf{4}, 5, \mathbf{3}, 7, \mathbf{12}, 2, \mathbf{9}, 8, \mathbf{10}, 6, \mathbf{1} \rangle$ and maximum degree of overlapping $k = 6$ according to the Algorithm 2. Assuming $\alpha = 0$, receiver SU_2 forms the sequence $S(= S_d)$ and forms receiver's symmetric CH (SCH) sequence $\mathcal{S}_r = \langle S \| S \| \cdots \rangle$ by concatenating sequence S repeatedly in each time cycle during execution of Algorithm 2. SU_1 uses the elementary sequence $S_e = \langle 2, \mathbf{4}, 8, \mathbf{3}, 6, \mathbf{12}, 11, \mathbf{9}, 5, \mathbf{10}, 7, \mathbf{1} \rangle$ and maximum degree of overlapping $k = 6$ according to the Algorithm 1. Assuming $\beta = 0$, sender SU_1 forms the sequence $S^1 = S_e$ during the execution of Algorithm 1. In the transmission mode, sequence S is generated from S^1 for each cycle and sender's CH sequence is generated as $\mathcal{S}_s^1 = \langle S \| S \| \cdots \rangle$ by concatenating the sequence S repeatedly in each time cycle. If SU_1 and SU_2 follow the symmetric model and their clocks are synchronized with each other, SU_1 uses the SSync algorithm to have the first rendezvous channel as given in Algorithm 4. Subsequently, if we consider the case of rendezvous on all channels, we need to run the algorithm until successful rendezvous occurs in all channels. As shown in Fig. 4, sender SU_1 follows the sequence S^1 in the first cycle during the first 12 time slots within which maximum 6 numbers of rendezvous are possible in those 6 available channels. Since, maximum number of rendezvous in first cycle is 6, we consider the cyclic left rotation of the sequence S^1 six times. Therefore, it hops sequentially in the second cycle using the sequence generated from the cyclic left rotation $RotL(S^1, k)$ in sequence S^1 with $k = 6$, as a result of which it is found that six rendezvous are possible with the remaining six available channels. In this way, sender can be able to rendezvous on all the $N(= 12)$ available channels within 24 time slots in $N/k(= 2)$ cycles, which satisfies the property of the guaranteed rendezvous in all the available channels.

Lemma 1: In Symmetric Synchronous CH (SSync), any two secondary users can achieve rendezvous in time T_{SS} in each

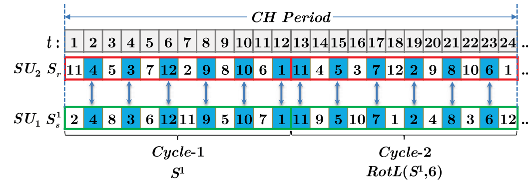


Fig. 4. Example of SSync Rendezvous Algorithm.

cycle, whose lower and upper bound are $\frac{N}{k}$ and $\frac{N^2}{k}$ time slots, respectively, where N is the number of available channels and k is the number of overlapping of sequences in each cycle.

Proof: In SSync, each cycle consists of N number of time slots. At least k number of rendezvous are possible on distinct channels in each cycle as there exists k number of overlapping between a pair of channel sequences (sender/receiver). Hence, the first rendezvous of channel sequence can occur in the initial cycle within $\frac{N}{k}$ time slots, which is the lower bound. In order to meet all N number of channels, it needs $\frac{N}{k}$ cycles. Hence, the rendezvous occurs in $\frac{N^2}{k}$ time slots for all N channels, which is the upper bound. ■

Theorem 1: In Symmetric Synchronous CH (SSync), the lower bound of Maximum Time To Rendezvous (MTTR) is $\frac{N}{k}$, where N is the number of available channels for the SUs and k is the number of overlapping of sequences in each cycle.

Proof: As proved in Lemma 1, the first rendezvous of channel sequence can occur in the initial cycle within $\frac{N}{k}$ time slots, when each cycle in SSync consists of N number of time slots with k number of overlapping between a pair of sender/receiver's channel sequences. Hence, the lower bound of MTTR is $\frac{N}{k}$, which is the lower bound of T_{SS}. ■

Lemma 2: The Average Time To Rendezvous (ATTR) in Symmetric Synchronous CH (SSync) is $\frac{N}{k}$, where N is the number of available channels for the SUs and k is the number of overlapping of sequences in each cycle.

Proof: ATTR is the average time taken for the first rendezvous and the proof is similar to the proof of Lemma 1. ■

Lemma 3: The Expected Time To Rendezvous (ETTR) in Symmetric Synchronous CH (SSync) is $ETTR_{SSync} = \frac{N}{k}$, where N is the number of available channels for the SUs and k is the number of overlapping of sequences in each cycle.

Proof: The period of rendezvous is defined as the least time period in which an algorithm is able to determine N number of overlapping on N different channels with every possible delay of d time slots. It is to be noted that MTTR in SSync is $\frac{N}{k}$ and there is no delay in SSync, i.e., $d = 0$. As rendezvous can occur periodically in at least N/k time slots with $d = 0$, $ETTR_{SSync} = \frac{N}{k}$. ■

Lemma 4: The Expected Inter Rendezvous Interval (EIRI) in Symmetric Synchronous CH (SSync) is $EIRI_{SSsync} = \frac{N}{k} - 1$, where N is the number of available channels for the SUs and k is the number of overlapping sequences in each cycle.

Proof: The Expected Inter Rendezvous Interval (EIRI(d)) for a pair of SUs can be calculated as $EIRI(d) = \frac{IRI(d)}{\mathcal{R}_{\mathcal{T}} - IRI(d)}$ for all possible cases of delay d with different patterns of rendezvous. In SSync channel hopping, $d = 0$ and the

Algorithm 5 SAsync Rendezvous Algorithm (For Sender)

Require: Sender SU_j, $j \in [1, L]$, number of available channels N, elementary sequence $S_e = \langle c_1^s, c_2^s, \cdots c_N^s \rangle$ and maximum degree of overlapping k;
Ensure: Rendezvous channel index c;
1: $S = S_e$;
2: Choose random integer $\beta \in [0, k)$;
3: $S = RotL(S, \beta \times k)$;
4: $S^j = S$; $//S^j = \langle c_1^j, c_2^j, \cdots c_N^j \rangle$
5: $cycle = 0$;
6: $t = 1$;
7: **while** (no successful rendezvous) **do**
8: $cycle = cycle + 1$;
9: **for** $r = 1$ to k **do**
10: **for** $i = 1$ to $\frac{N}{k}$ **do**
11: Scan on channel $c_i^j \in S^j$ to rendezvous;
12: **if** (rendezvous success) **then**
13: $c = c_i^j$;
14: **return** c;
15: **end if**
16: $t = t + 1$;
17: **end for**
18: $S^j = RotL(S^j, \frac{N}{k} + 1)$;
19: **end for**
20: $S^j = RotL(S, cycle)$;
21: **end while**

rendezvous period is $\mathcal{R_T} = \frac{N^2}{k}$. Hence, in SSync, $EIRI_{SSsync} = \frac{IRI(0)}{\mathcal{R_T} - IRI(0)} = \frac{N-k}{k} = \frac{N}{k} - 1$. ∎

VI. SAsync Rendezvous Algorithm

In Symmetric Asynchronous Channel Hopping (SASync) algorithm, it is assumed that two users join the CRN at different instant of time and both sender and receiver know about the number of available channels N in the CRN. Therefore, they execute the SASync rendezvous algorithm in which sender's asynchronous channel hopping rendezvous algorithm is employed. In asynchronous environment, time slots of SUs are not aligned to each other. As defined in IEEE 802.22, the frame transmission duration is T(=10ms) and each time slot must have a duration of $2T$ [9] to ensure the successful exchange of information in asynchronous environment. If SU_2 starts later than SU_1 by $(2\tau T + \delta)$ time slots, we consider the delay d as τ time slots when clock drift $\delta < T$, otherwise $d = (\tau + 1)$ time slots. Henceforth, for simplicity we consider that the time slots are aligned to each other with a delay factor of d time slots corresponding to the CH period. Thus, we consider two different types of cycles based on the fixed and variable delay factor of d units. Small cycles induced due to the delay factor is known as the *penalty* cycle, whose length is $\frac{N}{k}$ time slots and a large *rendezvous* cycle consists of rendezvous channels whose length is N time slots.

During *penalty* cycle, sender SU_j selects the channel indexed from sequence S^j by leaving $mod\left(i, \frac{N}{k} + 1\right), i \in [1, N]$ channels in a cycle of N time slots. In ideal case, rendezvous can occur in the first N time slots, where it

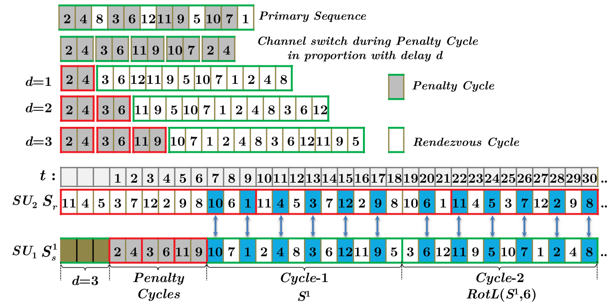


Fig. 5. SAsync Rendezvous with a fixed delay of $d = 3$ units.

consists of $mod(d, k)$ number of *penalty* cycles followed by a *rendezvous* cycle. Rendezvous cycle continues that consists of at most k number of rendezvous channels. In our algorithm, it returns the channel number when rendezvous is successful and the process is terminated. But, in general the rendezvous process continues with respect to its sequence S^j and rendezvous can occur in all N number of channels in the next $\frac{N}{k}$ number of cycles. In an ideal case, all channels can be used by the SUs and therefore MTTR for SAsync algorithm could be N. Hence, rendezvous can be guaranteed within the $(mod(d, k)\frac{N}{k} + \frac{N^2}{k})$ time slots for all N number of channels.

If the rendezvous does not occur in the first N time slots, it implies that no successful rendezvous is possible at least in one rendezvous channel, which passes k number of penalty cycles. After each N time slots, the primary sequence S^j is reconstructed from the sequence S using line 20 in Algorithm 5. Here, N time slots constitute the *non-rendezvous* cycle of k number of penalty cycles. In this scenario, penalty cycles continues until successful rendezvous occurs. Once the rendezvous is successful in a penalty cycle, it converts to the *rendezvous* cycle of N time slots. Each rendezvous cycle can have at most k number of rendezvous channels. In this scenario, Maximum Conditional Time To Rendezvous (MCTTR) for SAsync algorithm is N^2 time slots.

If SU_1 and SU_2 follow the symmetric model without having their clock synchronized with each other, then SU_1 uses SAsync mechanism to have the first successful rendezvous channel as given in Algorithm 5. Suppose, SU_1 starts the rendezvous process with a delay of d time slots after the start of receiver SU_2's clock. If SU_1 hops to rendezvous with SU_2's SCH sequence, it fails in the first $mod(d, k)$ *penalty* cycle. For example, as shown in Fig. 5, $d = 3$ and $k = 6$. Hence, the number of *penalty* cycles in which rendezvous attempt fails is three ($mod(d, k) = 3$) and after three *penalty* cycles (6 time slots), the first rendezvous occurs in the next *rendezvous* cycle. Subsequently, if we consider the case of rendezvous on all channels, we need to consider the rendezvous cycles till all channels get successful rendezvous. As shown in Fig. 5, initially sender SU_1 follows three *penalty* cycles using sequence S^1. After these penalty cycles, rendezvous cycle starts and it consists of maximum $k(= 6)$ number of rendezvous channels. Since, maximum number of rendezvous in the first cycle is six, we consider the cyclic left rotation of the sequence S^1 by six times. Hence, it hops sequentially using the sequence achieved from the $RotL(S^1, k)$ with $k = 6$

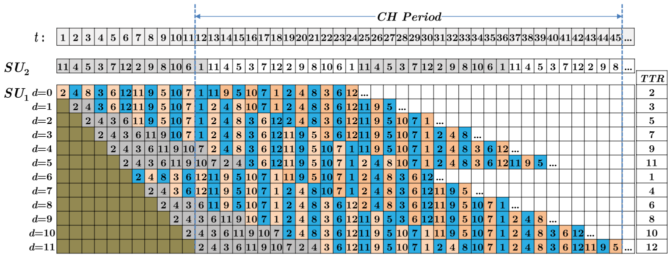


Fig. 6. SAsync Rendezvous with variable delay of d units.

in the next cycle. As a result, it is found that 6 rendezvous are possible with the remaining 6 available channels. Thus, rendezvous attempt fails within the first 6 time slots and all channels are scanned successfully for the rendezvous in the next 24 time slots. Hence, sender is able to rendezvous with all N number of channels during 30 time slots, which satisfies the guaranteed rendezvous property. In Fig. 6, SU_2 hops using the default hopping sequence and SU_1 can hop using its own hopping sequence based on its own local clock with a delay factor of d time slots.

Lemma 5: In Symmetric Asynchronous CH (SAsync), any two users can achieve rendezvous within time T_{SA} in a cycle, whose lower and upper bound are $\frac{N}{k} \times (mod(d,k)+1)$ and $\frac{N}{k} \times (mod(d,k)+N)$ time slots, respectively, where N is the number of available channels, k is the number of overlapping of sequences in each cycle and d time slots is the delay of starting time between two users.

Proof: In SAsync, sender SU_2 starts after SU_1 with a delay of d time slots. In the first phase of the algorithm, SU_2 searches for the rendezvous cycle, which consists of k number of rendezvous channels. Rendezvous attempts fail during the $mod(d,k)$ penalty cycles, which need at least $\frac{N}{k} \times (mod(d,k))$ time slots and do not contain any rendezvous channels. Rendezvous attempt can be succeeded in the next $\frac{N}{k}$ time slots for the the first time rendezvous. Therefore, the rendezvous can have a lower bound of $\frac{N}{k} \times (mod(d,k)+1)$ time slots. In the second phase, SU_2 runs the algorithm for the rendezvous cycles of N time slots in order to meet all distinct N channels, which needs $\frac{N}{k}$ number of cycles for the successful rendezvous. Hence, the upper bound for successful rendezvous is $\frac{N}{k} \times (mod(d,k)+N)$ time slots for all N channels. ∎

Theorem 2: In SAsync channel hopping, Maximum Time To Rendezvous (MTTR) is N, where N is the number of available channels.

Proof: As proved in Lemma 5, the rendezvous can have a lower bound of $\frac{N}{k} \times (mod(d,k)+1)$ time slots in the ideal case of SAsync channel hopping. Rendezvous can occur in the first N time slots, where it consists of $mod(d,k)$ number of *penalty* cycles followed by a *rendezvous* cycle. Hence, MTTR in SAsync is N. ∎

Lemma 6: The Average Time To Rendezvous (ATTR) in Symmetric Asynchronous CH (SAsync) is $\frac{N}{2}$, where N is the number of available channels for the SUs and k is the number of overlapping of sequences in each cycle.

Proof: The proof is similar to Lemma 5 and is not given here to save space. ∎

Lemma 7: The Expected Time To Rendezvous (ETTR) in Symmetric Asynchronous CH (SAsync) is $ETTR_{SAsync} = \sum_{d=0}^{k-1} \frac{1}{k} \times \frac{\mathcal{R}_{\mathcal{T}}}{TTR(d)}$, where $\mathcal{R}_{\mathcal{T}}$ is the period of rendezvous and $TTR(d)$ is the time to rendezvous with delay d.

Proof: In SAsync channel hopping, time to rendezvous with delay d (TTR(d)) can be calculated for each $d \in \{0, 1, \cdots, k-1\}$ with a generalized formula $TTR(d) = \frac{k}{N} \times \mathcal{R}_{\mathcal{T}} - mod(d,k)$, where k is the number of overlapping of sequences. Besides, the period of rendezvous in SAsync $\mathcal{R}_{\mathcal{T}} = (N - \frac{N}{k}) + \frac{N^2}{k}$. Hence, ETTR($d$) for a pair of SUs can be calculated as $ETTR(d) = \frac{\mathcal{R}_{\mathcal{T}}}{TTR(d)}$. For all possible cases of d with different patterns of rendezvous, $ETTR_{SAsync} = \sum_{d=0}^{k-1} \frac{1}{k} \times \frac{\mathcal{R}_{\mathcal{T}}}{TTR(d)}$. ∎

Lemma 8: The Expected Inter Rendezvous Interval (EIRI) in Symmetric Asynchronous CH (SAsync) is $EIRI_{SAsync} = \sum_{d=0}^{k-1} \frac{1}{k} \times \frac{IRI(d)}{\mathcal{R}_{\mathcal{T}} - IRI(d)}$, where N is the number of available channels for the SUs and k is the number of overlapping of sequences in each cycle.

Proof: The Inter Rendezvous Interval (IRI(d)) can be determined for each $d \in \{0, 1, \cdots, k-1\}$ with the generalized formula $IRI(d) = \frac{N-k}{N} \times \mathcal{R}_{\mathcal{T}} + mod(d,k)$, where k is the number of overlapping of sequences in SAsync channel hopping with rendezvous period $\mathcal{R}_{\mathcal{T}} = (N - \frac{N}{k}) + \frac{N^2}{k}$. Hence, EIRI($d$) for a pair of SUs is calculated as $EIRI(d) = \frac{IRI(d)}{\mathcal{R}_{\mathcal{T}} - IRI(d)}$. For all possible cases of d with different patterns of rendezvous, $EIRI_{SAsync} = \sum_{d=0}^{k-1} \frac{1}{k} \times \frac{IRI(d)}{\mathcal{R}_{\mathcal{T}} - IRI(d)}$. ∎

VII. AAsync Rendezvous Algorithm

In the CRN, if the unlicensed users have different sets with variable number of available channels such that there exists at least one common channel between them and they join the network at different instant of time, then they have to execute the Asymmetric Asynchronous Channel Hopping (AAsync) algorithm. Here, the sender has knowledge about the available channels by sensing the presence of the PUs. It is assumed that the sender knows about the number of total channels (M) of the CRN and currently available channel set $C_j^s = \{c_1, c_2, \cdots, c_N\}$. Using primitive root $g_i \in \mathcal{R} \setminus \{g_\ell\}$, sender can find the sequence S_e, which is presented in the Section III and $S^j = \langle c_1^j, c_2^j, \cdots c_M^j \rangle$. In the first cycle of the algorithm, user considers the channel index $c_i^j \in S^j$ in each time slot by checking whether it belongs to C_j^s. If $c_i^j \notin C_j^s$, it is remapped to the channel $C_j^s[mod(cycle-1, N)+1]$ as stated in line 13 of Algorithm 6 and the algorithm checks for the rendezvous on C. If the rendezvous fails using sequence S^j in M time slots, the algorithm continues up to M cycles until successful rendezvous occurs. Hence, Maximum Conditional Time To Rendezvous (MCTTR) in AAsync rendezvous algorithm is M^2 time slots.

Consider an example, where sender SU_1 and receiver SU_2 follow the asymmetric channel hopping model without their clock synchronization. Let, $\mathcal{C} = \{1, 2, 3, 4, 5, 6\}$ be the set of total channels in the CRN. Here, $M = 6$, $P = 7$ and set of primitive roots $\mathcal{R} = \{3, 5\}$. Hence, the sequences corresponding to the sender and receiver are $S_e = \langle 3, 2, \mathbf{6}, 4, 5, \mathbf{1} \rangle$ and $S_d = \langle 5, 4, \mathbf{6}, 2, 3, \mathbf{1} \rangle$, respectively. The maximum degree

Algorithm 6 AAsync Rendezvous Algorithm (For Sender)

Require: Sender SU_j, $j \in [1, L]$, total number of channels M, elementary sequence $S_e = \langle c_1^s, c_2^s, \cdots c_M^s \rangle$ and maximum degree of overlapping k, number of available channels N, set of available channels for SU_j is $C_j^s = \{c_1, c_2, \cdots, c_N\}$;

Ensure: Rendezvous channel index c;

$S = S_e$;
Choose random integer $\beta \in [0, k)$;
$S = RotL(S, \beta \times k)$;
$S^j = S$; $//S^j = \langle c_1^j, c_2^j, \cdots c_M^j \rangle$
$cycle = 0$;
$t = 1$;
while (no successful rendezvous) **do**
  $cycle = cycle + 1$;
  for $i = 1$ to M **do**
    if $c_i^j \in C_j^s$ **then**
      $c = c_i^j$;
    else
      $c = C_j^s[mod(cycle - 1, N) + 1]$; //by Sec. IV-C
    end if
    Scan on channel c to rendezvous;
    if (rendezvous success) **then**
      return c;
    end if
    $t = t + 1$;
  end for
  $S^j = RotL(S^j, 1)$;
end while

of overlapping between these sequences is $k = 2$. The available channel sets of sender SU_1 and receiver SU_2 are $C_r = \{2, 3, 4\}$ and $C_1^s = \{1, 2, 5\}$, respectively. Receiver SU_2 follows the ACH sequence as given in Algorithm 3 to generate the receiver's channel hopping sequence $\mathcal{S}_r$ by using the default sequence S_d and available channel set $C_r = \{2, 3, 4\}$. Receiver SU_2 hops using its channel hopping sequence $\mathcal{S}_r$. Sender SU_1 follows the AAsync rendezvous Algorithm 6 to find its sender's hopping sequence and rendezvous channels. In this example, there is one rendezvous channel, which is 2. As shown in Fig. 7a, SUs hop in all channels without knowing their available channels as a result of which they rendezvous with each other at time slot 7 once in the entire CH period. In the AAsync algorithm, the unavailable channels are replaced with their respective available channels as shown in Fig. 7b. In this case, the SUs can rendezvous on common channel 2 in the time slots 5, 6, 7, 23, and 24.

Lemma 9: In Asymmetric Asynchronous CH (AAsync), any two users can achieve rendezvous within time T_{AA}, whose upper bound is M^2 time slots, where M is the number of total channels.

Proof: If the number of commonly available channels is one and the delay with the available common channel overlapping is $d \in [1, M - 1]$, which may occur at the end of M cycles, the number of cycles is the maximum and the number of time slots in this case is M in each cycle.

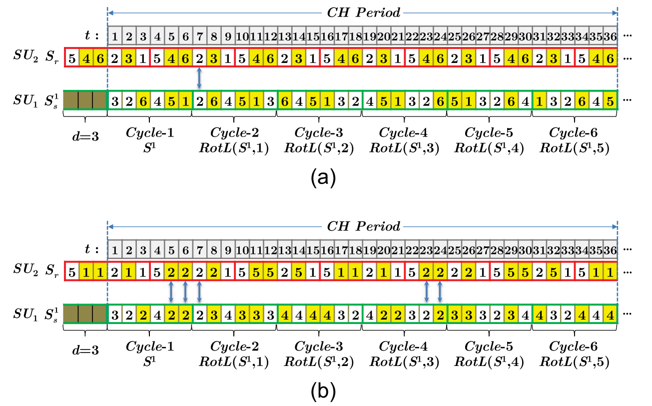


Fig. 7. $\mathcal{C} = \{1, 2, 3, 4, 5, 6\}, M = 6$; Available channels of sender SU_1 and receiver SU_2 are $\{2, 3, 4\}$ and $\{1, 2, 5\}$, respectively. (a) AAsync Rendezvous without replacing the unavailable channels with available channels. (b) AAsync Rendezvous by replacing unavailable channels with available channels.

Hence, the rendezvous is guaranteed in M^2 time slots, if at least one common channel is available between any two users. ■

Theorem 3: The Maximum Time To Rendezvous (MTTR) in Asymmetric Asynchronous CH is M^2, where M is the number of total channels.

Proof: As proved in Lemma 9, rendezvous is guaranteed in M^2 time slots and therefore MTTR is M^2. ■

Lemma 10: The Average Time To Rendezvous (ATTR) in Asymmetric Asynchronous CH is $\frac{M \times P}{2}$, where M is the number of total channels for the SUs and P is a prime number.

Proof: The proof is similar to Lemma 9 and is not given here to save space. ■

A. ATTR With Respect to PUs

We analyze here the probability of an SU that can rendezvous with a PU when it opportunistically tries to access the spectrum in the CRN. Assuming that p is the probability of a PU that appears on a channel for one time slot, probability that the PU will occupy this particular channel during the period is given by

$$\mathcal{P}_{PU} = 1 - (1 - p)^{\mathcal{R}_{\mathcal{T}}} \tag{1}$$

where, $\mathcal{R}_{\mathcal{T}}$ refers to the time period required for hopping a sequence. This time period in our algorithms is represented as $\mathcal{R}_{\mathcal{T}} \in \{\frac{N^2}{k}, (N - \frac{N}{k}) + \frac{N^2}{k}, M^2\}$, where k is the numbers of overlapping in the receiver/sender's sequences. The average number of channels occupied by a PU (n_a) within a hopping sequence period will be

$$n_a = M \cdot \mathcal{P}_{PU} = M\left(1 - (1 - p)^{\mathcal{R}_{\mathcal{T}}}\right) \tag{2}$$

It is to be noted that ATTR increases due to presence of the PU. Let, ATTR and ATTR′ be the average time to rendezvous without or with the influence of PU, respectively. Clearly, ATTR′ > ATTR, i.e. $\frac{\text{ATTR}'}{\text{ATTR}} > 1$. The impact of PU is directly proportional to the fraction of ATTR′ with ATTR. Therefore, the impact of PU (f) can be calculated as follows:

$$f = \frac{\text{ATTR}'}{\text{ATTR}} \tag{3}$$

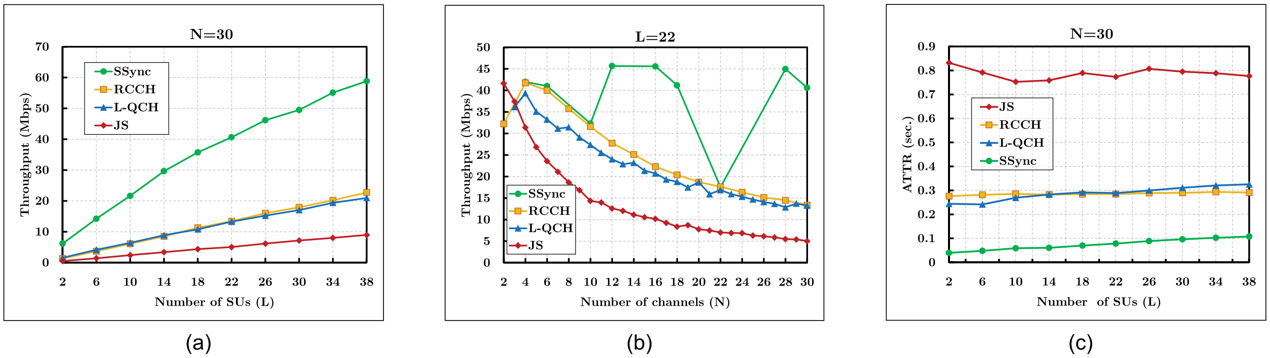


Fig. 8. Symmetric Synchronous protocols. (a) Throughput vs different number of SUs (L) with fixed number of channels ($N = 30$). (b) Throughput vs different number of channels (N) with $L = 22$. (c) ATTR vs different number of SUs (L) with fixed number of channels ($N = 30$).

Hence, we get ATTR′ under the impact of PU's appearance on ATTR by

$$\text{ATTR}' = \frac{M}{M - n_a}\text{ATTR} \tag{4}$$

The impact factor of PU on the performance of our algorithms in terms of ATTR is given as

$$f = \frac{\text{ATTR}'}{\text{ATTR}} = \frac{M}{M - n_a} = \frac{1}{(1-p)^{\mathcal{R}_\mathcal{T}}} \tag{5}$$

where, value of $\mathcal{R}_\mathcal{T} \in \{\frac{N^2}{k}, (N - \frac{N}{k}) + \frac{N^2}{k}, M^2\}$ depends on our SSync, SAsync or AAsync algorithms.

B. Available Channel Ratio (φ)

Available channel ratio is defined as the ratio of average number of available channels of an SU ($N = M - n_a$) to the number of total channels (M), which is the inverse of the impact factor of PU and is given as follows:

$$\varphi = \frac{M - n_a}{M} = (1-p)^{\mathcal{R}_\mathcal{T}} = \frac{1}{f} \tag{6}$$

where, $\mathcal{R}_\mathcal{T} = M^2$. Value of φ decreases, when number of available channels of an SU reduces and accordingly the number of available channels in asymmetric environment can be φM. In order to have at least one common available channel between one pair of SU, φ should be within the range $0.5 < \varphi < 1$. For example, if $M = 20$ and $\varphi = 0.5$, then each SU can have 10 number of available channels and there is a chance that some pairs of SUs do not have any common available channel among them. Therefore, φ must be larger than 0.5. In asymmetric environment, we consider $\varphi = 0.8$ in our simulation in order to have more common available channels among the SUs for the performance evaluation of our algorithms, which is discussed in the next section.

VIII. Performance Evaluation

A. Simulation Setup

We build our CR based simulation environment using OMNeT++ simulator to evaluate the performance of our proposed algorithms to compare with the state-of-art protocols such as FRCH [7], JS [9], RCCH [12], ARCH [12], SARCH [12], L-QCH [13], MTP [14] and EJS [22]. Our simulation platform is implemented with variable number of SUs with variable number of available channels in presence of fixed number of PUs, where SUs are deployed randomly over the CRN. The duration of each time slot in the simulation is considered to be $20ms$ and packet data rate to be $22.69Mbps$ based on the existing standard [17]. The traffic generated at each SU follows the Poisson process of arrival and each SU maintains multiple queues for its one-hop neighbors. Sender SU randomly selects the neighbor to which it has data in its queue based on the availability of the receiver SU. Destination SU is decided when the RTS sent by the source SU is cleared by the CTS during the rendezvous process. Thereafter, the transmission continues until all packets are transmitted during the subsequent time slots, where they rendezvous.

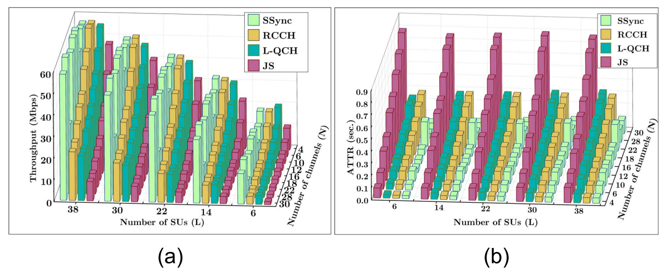


Fig. 9. Symmetric Synchronous protocols. (a) Throughput vs different number of SUs (L) and different number of channels (N). (b) ATTR vs different number of SUs (L) and different number of channels (N).

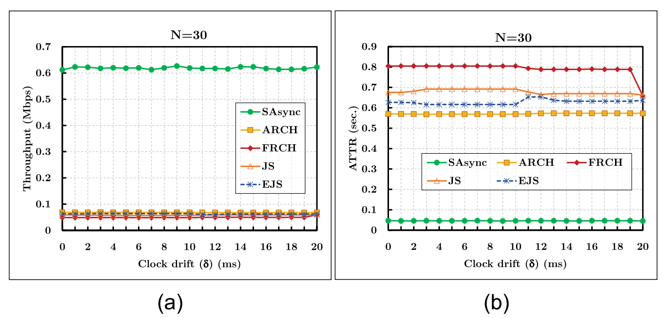
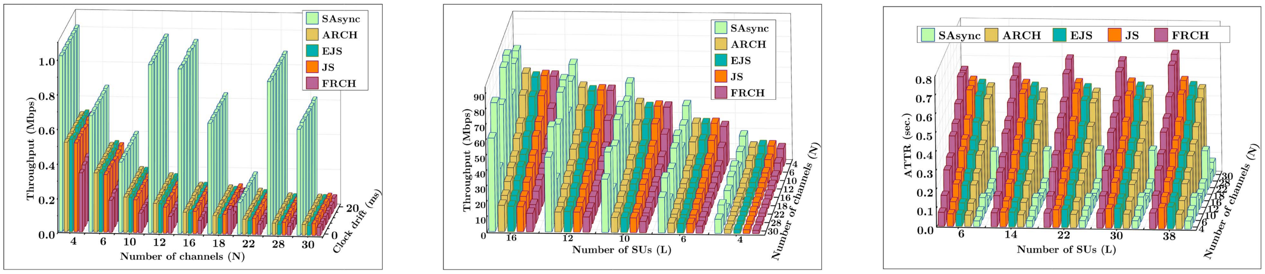


Fig. 10. Symmetric Asynchronous protocols. (a) Throughput vs δ for fixed number of channels ($N = 30$). (b) ATTR vs δ for fixed number of channels ($N = 30$).



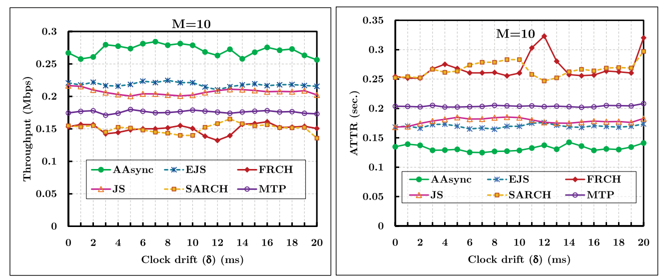
(a) (b) (c)

Fig. 11. Symmetric Asynchronous protocols. (a) Throughput vs different number of channels (N) vs δ. (b) Throughput vs different number of SUs (L) vs different number of channels (N). (c) ATTR vs different number of SUs (L) vs different number of channels (N).

In our simulation, SSync algorithm is compared with symmetric algorithms JS [9], RCCH [12] and L-QCH [13] in synchronous environment. The simulation process for each symmetric synchronous protocols is considered taking same available channel sets for symmetric environment with zero clock drift for synchronization. Besides, we consider FRCH [7], ARCH [12] and EJS [22] to compare with our SAsync algorithm in symmetric asynchronous environment. The simulation process for each symmetric asynchronous protocols includes the channel hopping system of symmetric model, where each SU can transmit asynchronously with variable clock drift. In order to evaluate the performance of our AAsync algorithm, we consider FRCH [7], JS [9], SARCH [12], MTP [14] and EJS [22] algorithms for asymmetric asynchronous environment. The simulation process for each asymmetric asynchronous protocol is implemented by considering asymmetric model, where each SU can have different available channel sets with at least one common channel among them and uses variable clock drifts to satisfy the asynchronous condition. We use throughput and ATTR to compare our protocols with all stat-of-art protocols. In our simulation, throughput is defined as the number of data packets successfully transmitted per unit second, where size of each packet is 2000 bytes. Thus, throughput is measured as $\frac{\text{Number of packets transmitted} \times \text{Size of each packet}}{\text{Simulation time period in seconds}}$.

B. Symmetric Synchronous Scenario

When an SU has data to send to another intended SU, it switches to transmission mode and hops from one channel to another using our SSync algorithm at the beginning of each time slot. In symmetric synchronous scenario, we simulate the existing protocols RCCH, L-QCH and JS in synchronous environment. When the number of SUs increases, overall throughput is also increases, which is clearly seen in the Fig. 8a. Our SSync algorithm outperforms over L-QCH, JS in all available channels in terms of throughput and specifically throughput in our algorithm is significantly higher, where maximum rendezvous channels are $N \in \{12, 16, 18, 28, 30\}$, which is depicted in Fig. 8b. The throughput in our protocol is higher than the JS algorithm when number of channels $N \in \{12, 16, 18, 28, 30\}$. Since, the maximum degree of overlapping of those respective channels is $k \in \{6, 8, 6, 14, 10\}$, throughput of our protocol fluctuates. The throughput of other



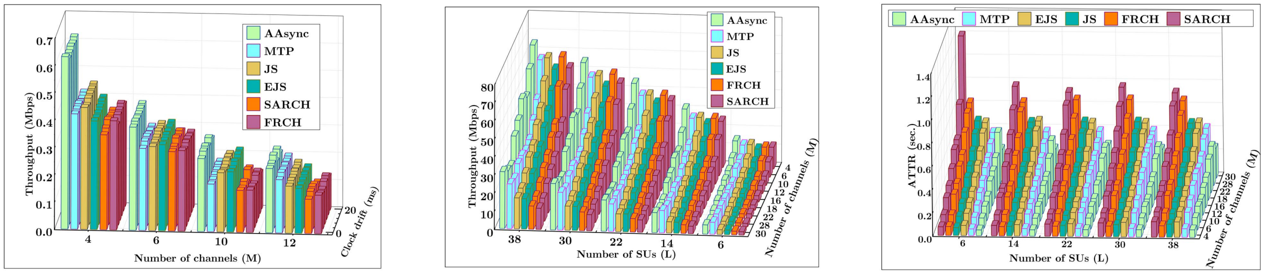
(a) (b)

Fig. 12. Asymmetric Asynchronous protocols. (a) Throughput vs δ for $M = 10$ and $\varphi = 0.8$. (b) ATTR vs δ for $M = 10$ and $\varphi = 0.8$.

protocols such as RCCH decreases when number of channels increases. This happens because of the degree of overlapping between sequences in each cycle is always $k = 2$. However, in our SSync algorithm, there is room for better performance when number of channels $N \in \{12, 16, 18, 28, 30\}$ and degree of overlapping is $k \neq 2$. In RCCH, degree of overlapping is always 2 in each cycle. Our protocol performance is closer to that of RCCH in case of maximum rendezvous channels $N \in \{4, 6, 10, 22\}$ as the number of rendezvous in each rendezvous cycle (N time slots), degree of overlapping is two. However, as shown in Fig. 9a throughput in our protocol is significantly higher due to higher degree of overlapping for different sets of SUs in other available rendezvous channels. The numbers of rendezvous in each rendezvous cycle (N time slots), where $N = 12, 16, 18, 28, 30$ are 6, 8, 6, 14, 10, respectively, which gives high throughput as seen in Fig. 8b for $L = 22$. In Fig. 8c, we find that ATTR of our protocol is less than other protocols for variable number of SUs with fixed number of channels $N = 30$. It is observed that ATTR increases as value of N increases in all existing algorithms as compared to ours. However, in our SSync algorithm there is room for significant reduction of ATTR for the number of available channels (as seen for $N = 12, 16, 18, 28, 30$) as shown in Fig. 9b. Besides, our algorithm also outperforms in terms of throughput for the number of available channels (N), where throughput is higher than other protocols.

C. Symmetric Asynchronous Scenario

In our SAsync algorithm, it is considered that each SU can start at different instant of time with variable clock drift (δ)



(a) (b) (c)

Fig. 13. Asymmetric Asynchronous protocols for $\varphi = 0.8$. (a) Throughput vs M vs δ. (b) Throughput vs L vs M.(c) ATTR vs L vs M.

from the start of the time slot. In two users scenario, one SU is considered as a sender and other one as a receiver with possible amount of δ to compare with other algorithms. As shown in Fig. 10a, though maximum throughput is achieved as compared to other protocols, the achieved throughput does not vary much with respect to the clock drift. The reason is that each SU can transmit multiple data packets within a single time slot after successful rendezvous in our simulation. Accordingly, the numbers of transmitted data packets are less due to rendezvous in the first time slot with variable clock drift. However, maximum number of pending data packets are transmitted to the desired receiver in the subsequent rendezvous time slots as the pair of users are already synchronized. Thus, the impact of first rendezvous time slot reduces the overall throughput of the network with different clock drift though maximum throughput is achieved.

SAsync performs significantly better than other protocols in terms of throughput as well as ATTR, which are shown in Fig. 10a and Fig. 10b, respectively for the fixed number of channels $N = 30$. For different values of δ and varying number of N, throughput is depicted in Fig. 11a in case of two users scenario, where SAsync outperforms over other algorithms. In multi-user scenario, when an SU has data to send, it switches to the transmission mode to a channel using SAsync algorithm. In this scenario, when a pair of SUs meet at a rendezvous channel at any time slot, the sender adjusts its clock with receiver's clock and continues sending data until its all packets are sent successfully to its intended receiver or the receiver switches to the sending mode. Thus, the receiver is able to receive in different time slots from different SUs, if it is idle. In our simulation, our SAsync algorithm is compared with existing ARCH, FRCH, JS, and EJS algorithms. As depicted in Fig. 11b, throughput of our algorithm is found to be better as compared to others. Besides, ATTR is found to be significantly less than those of existing algorithms, which is shown in Fig. 11c.

D. Asymmetric Asynchronous Scenario

In asymmetric asynchronous scenario, it is considered that SUs are within the proximity of each other with different sets of available channels having at least one common channel between them with respect to the presence of PUs. Under such scenario, a new parameter φ is considered such that number of available channels N_j is φM. For each SU_j, the set of available channels C_j^s consists of φM number of different set of available channels out of total M channels. Our AAsync protocol is run with $\varphi = 0.8$ and is compared with SARCH, FRCH, JS, EJS, MTP. In our comparison, SARCH, FRCH, JS, EJS, and MTP are simulated in asymmetric asynchronous environment with respect to different sets of available channels. In two users scenario, taking one SU as sender and another one as the receiver, we simulated our AAsync algorithm with these protocols for possible value of δ. It is observed that AAsync significantly performs better in terms of throughput and ATTR as shown in Fig. 12a and Fig. 12b, respectively for fixed number of channels $M = 10$. For different values of δ and varying number of M, throughput is evaluated for two users case as depicted in Fig. 13a.

Simulation result in Fig 13b shows that the number of unavailable channels increases as the number of total channels increases though the available channel ratio $\varphi = 0.8$. The channel replacement strategy may not guarantee the rendezvous in regular interval of time. As a result of which, average TTR of all protocols increases with increase in value of M as depicted in Fig. 13c. As shown in Fig. 13b and Fig. 13c, our AAsync protocol outperforms over others in terms of throughput and average TTR, respectively for all values of M.

IX. Conclusion

In this paper, the dynamic spectrum sharing between the licensed PUs and unlicensed SUs is studied taking three channel hopping algorithms. All SUs are equipped with cognitive radios to sense the spectrum holes and can use same or different sets of available channels to send or receive data. A two-user and multiuser scenario is considered in the CRN for symmetric and asymmetric environment with/without need of synchronization. Accordingly, SSync, SAsync, and AAsync channel hopping algorithms are designed for symmetric synchronous, symmetric asynchronous and asymmetric asynchronous environments, respectively. All algorithms are analyzed to justify different performance metrics in terms of MTTR, ATTR and degree of rendezvous. Extensive simulations are performed to evaluate our algorithms and to compare them with some latest well known CRN algorithms. It is investigated that our proposed algorithms can outperform over others and to the best of our knowledge our protocols are most efficient as compared to the works published so far.

REFERENCES

[1] M. Song, C. Xin, Y. Zhao, and X. Cheng, "Dynamic spectrum access: From cognitive radio to network radio," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 23–29, Feb. 2012.

[2] J. Jia, Q. Zhang, and X. S. Shen, "HC-MAC: A hardware-constrained cognitive mac for efficient spectrum management," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 1, pp. 106–117, Jan. 2008.

[3] C. Cormio and K. R. Chowdhury, "Common control channel design for cognitive radio wireless ad hoc networks using adaptive frequency hopping," *Ad Hoc Netw.*, vol. 8, no. 4, pp. 430–438, 2010.

[4] L. A. DaSilva and I. Guerreiro, "Sequence-based rendezvous for dynamic spectrum access," in *Proc. 3rd IEEE Symp. New Frontiers Dyn. Spectr. Access Netw., (DySPAN)*, Oct. 2008, pp. 1–7.

[5] N. C. Theis, R. W. Thomas, and L. A. DaSilva, "Rendezvous for cognitive radios," *IEEE Trans. Mobile Comput.*, vol. 10, no. 2, pp. 216–227, Feb. 2011.

[6] D. Yang, J. Shin, and C. Kim, "Deterministic rendezvous scheme in multichannel access networks," *Electron. Lett.*, vol. 46, no. 20, pp. 1402–1404, Sep. 2010.

[7] G.-Y. Chang and J.-F. Huang, "A fast rendezvous channel-hopping algorithm for cognitive radio networks," *IEEE Commun. Lett.*, vol. 17, no. 7, pp. 1475–1478, Jul. 2013.

[8] T.-Y. Wu, W. Liao, and C.-S. Chang, "CACH: Cycle-adjustable channel hopping for control channel establishment in cognitive radio networks," in *Proc. IEEE INFOCOM*, Apr. 2014, pp. 2706–2714.

[9] H. Liu, Z. Lin, X. Chu, and Y.-W. Leung, "Jump-stay rendezvous algorithm for cognitive radio networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 10, pp. 1867–1881, Oct. 2012.

[10] I.-H. Chuang, H.-Y. Wu, and Y.-H. Kuo, "A fast blind rendezvous method by alternate hop-and-wait channel hopping in cognitive radio networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 10, pp. 2171–2184, Oct. 2014.

[11] K. Bian and J.-M. Park, "Maximizing rendezvous diversity in rendezvous protocols for decentralized cognitive radio networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 7, pp. 1294–1307, Jul. 2013.

[12] G. Y. Chang, W. H. Teng, H. Y. Chen, and J. P. Sheu, "Novel channel-hopping schemes for cognitive radio networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 2, pp. 407–421, Feb. 2014.

[13] K. Bian, J.-M. Park, and R. Chen, "Control channel establishment in cognitive radio networks using channel hopping," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 4, pp. 689–703, Apr. 2011.

[14] Z. Gu, H. Pu, Q. S. Hua, and F. C. M. Lau, "Improved rendezvous algorithms for heterogeneous cognitive radio networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 154–162.

[15] I. A. M. Balapuwaduge, F. Y. Li, A. Rajanna, and M. Kaveh, "Channel occupancy-based dynamic spectrum leasing in multichannel CRNs: Strategies and performance evaluation," *IEEE Trans. Commun.*, vol. 64, no. 3, pp. 1313–1328, Mar. 2016.

[16] C. Yang, W. Lou, Y. Fu, S. Xie, and R. Yu, "On throughput maximization in multichannel cognitive radio networks via generalized access strategy," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1384–1398, Apr. 2016.

[17] *IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems Wireless Regional Area Networks (WRAN)—Specific Requirements Part 22: Cognitive Wireless RAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Policies and Procedures for Operation in the TV Bands*, IEEE Standard 802.22, Working Group on Wireless Regional Area Networks, Jul. 2011.

[18] C.-M. Chao, H.-Y. Fu, and L.-R. Zhang, "A fast rendezvous-guarantee channel hopping protocol for cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5804–5816, Dec. 2015.

[19] Q. Liu, X. Wang, B. Han, X. Wang, and X. Zhou, "Access delay of cognitive radio networks based on asynchronous channel-hopping rendezvous and CSMA/CA MAC," *IEEE Trans. Veh. Technol.*, vol. 64, no. 3, pp. 1105–1119, Mar. 2015.

[20] T. Chakraborty, I. S. Misra, and T. Manna, "Design and implementation of VoIP based two-tier cognitive radio network for improved spectrum utilization," *IEEE Syst. J.*, vol. 10, no. 1, pp. 370–381, Mar. 2016.

[21] M. B. Ghorbel, B. Hamdaoui, M. Guizani, and B. Khalfi, "Distributed learning-based cross-layer technique for energy-efficient multicarrier dynamic spectrum access with adaptive power allocation," *IEEE Trans. Wireless Commun.*, vol. 15, no. 3, pp. 1665–1674, Mar. 2016.

[22] Z. Lin, H. Liu, X. Chu, and Y.-W. Leung, "Enhanced jump-stay rendezvous algorithm for cognitive radio networks," *IEEE Commun. Lett.*, vol. 17, no. 9, pp. 1742–1745, Sep. 2013.

[23] G. Y. Chang, J. F. Huang, and Y. S. Wang, "Matrix-based channel hopping algorithms for cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 5, pp. 2755–2768, May 2015.

[24] M. Monemi, M. Rasti, and E. Hossain, "On characterization of feasible interference regions in cognitive radio networks," *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 511–524, Feb. 2016.

[25] Y. Zhang, G. Yu, Q. Li, H. Wang, X. Zhu, and B. Wang, "Channel-hopping-based communication rendezvous in cognitive radio networks," *IEEE/ACM Trans. Netw.*, vol. 22, no. 3, pp. 889–902, Jun. 2014.

[26] L. Yu, H. Liu, Y. W. Leung, X. Chu, and Z. Lin, "Multiple radios for fast rendezvous in cognitive radio networks," *IEEE Trans. Mobile Comput.*, vol. 14, no. 9, pp. 1917–1931, Sep. 2015.

[27] L. Chen, K. Bian, X. Du, and X. Li, "Multichannel broadcast via channel hopping in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 7, pp. 3004–3017, Jul. 2015.

Prasan Kumar Sahoo (M'06) received the M.Sc. degree in mathematics from Utkal University, India, in 1994, the M.Tech. degree in computer science from IIT Kharagpur, Kharagpur, India, in 2000, the Ph.D. degree in mathematics from Utkal University, India, in 2002, and the Ph.D. degree in computer engineering from National Central University, Taiwan, in 2009. He was an Associate Professor with the Department of Information Management, Vanung University, Taiwan. He was with the Software Research Center, National Central University, Taiwan. He is currently an Associate Professor with the Department of Computer Science and Information Engineering and the Director of the International Cooperation Center, Chang Gung University, Taiwan. His current research interests include big data analytics, cloud computing, and cyber-physical systems with cognitive radio networks. He is also an Editorial Board Member of the *International Journal of Vehicle Information and Communication Systems* and has served as the Program Committee Member of several IEEE and ACM conferences. He was the Program Chair of ICCT 2010.

Debasish Sahoo received the M.Sc. degree in mathematics and the M.Tech. degree in computer science from Utkal University, India, in 1999 and 2007, respectively. He is currently pursuing the Ph.D. degree in computer science and information engineering at Chang Gung University, Taiwan. He was a Research Associate with the Department of Computer Science and Engineering, IIT Patna, Patna, India. His current research interests include cognitive radio networks and cognitive Internet of Things.