How Local Information Improves Rendezvous in Cognitive Radio Networks

***Abstract*—Cognitive Radio Network (CRN) is a promising technique aiming at solving the wireless spectrum scarcity problem. Rendezvous is the fundamental process of CRN. We are devoted to design faster rendezvous algorithms for CRN. We find that local information such as user's ID and the label of an available channel is very useful for designing faster rendezvous algorithms. First, we propose the Sequence-Rotating Rendezvous (SRR) algorithm. SRR algorithm can guarantee rendezvous for any two users i, j in $(2P^2 + 2P)$ timeslots, where P is the least prime not less than the total number of channels in the network. SRR algorithm has a lower MTTR than the extant asynchronous anonymous non-oblivious global-sequence-based rendezvous algorithms. Second, we utilize the user's identifier (ID) to design an ID-based Rendezvous (IDR) algorithm. IDR algorithm can guarantee rendezvous for any two users i and j in $(l+1)(P_i+2)(P_j+2)$ timeslots, where P_i and P_j are the smallest primes which are not less than the numbers of available channels of user i and j respectively. IDR algorithm has a lower MTTR than the extant asynchronous non-anonymous oblivious local-sequence-based rendezvous algorithms. Third, we propose a Channel-Label-based Rendezvous (CLR) algorithm which can guarantee rendezvous for any two users in $((P_i+2)(P_j+2)+P_N)(\lceil log_2 N \rceil + 1)$ timeslots, where N is the overall amount of channels in the network and P_N is the least prime which is not less than N. All of our algorithms can be used in multi-user scenario. We conduct plenty of experiments comparing our algorithms with state-of-the-art rendezvous algorithms under different scenarios, which confirm our analysis.**

***Index Terms*—Cognitive Radio Network, Local Information, Rendezvous Algorithms, Channel Label, identifier**

I. Introduction

Wireless spectrum is an invaluable resource for transmission between wireless devices. Normally speaking, wireless spectrum is divided into licensed spectrum which is occupied by paying users (also called primary users, PUs) and unlicensed spectrum which is free for unlicensed users (also called secondary users, SUs). With the development of wireless technology, unlicensed spectrum is becoming more and more severe due to rapid growth of wireless devices [12] and wireless services. However, the utilization rate of licensed spectrum is relatively low [16] [8]. Cognitive radio networks (CRNs) are thus proposed to alleviate the spectrum scarcity problem, which allows SUs to sense and utilize vacant licensed spectrum for communication by equipping with cognitive radios. [1]

In CRN, there exist many important processes such as broadcasting, routing, data gathering, etc. *Rendezvous* is a fundamental process of CRN, which targets finding a licensed channel on which two neighboring users can communicate. Since PUs have higher priority than SUs, when a PU is occupying a channel, every SU cannot access it. To construct a communication link on a common channel, some works assume a central controller or a common control channel exists which gets full knowledge of users' available channel sets, then two users can be assigned with a common available channel for communication [9] [10] [15] [17]. However, establishing a central controller is cost and vulnerable if a huge number of users require communication concurrently or the central controller gets attacked. Therefore, more distributed rendezvous algorithms are proposed which assume two users do not know each other's information and the users run their algorithms separately [13] [6] [18] [19] [20]. This kind of algorithms is also called "blind" rendezvous algorithms and it becomes more practical than centralized algorithms. Channel Hopping (CH) technique [1] [2] is a widely adopted method in blind rendezvous algorithms, in which a user selects a channel for rendezvous attempt according to a pre-generated channel hopping sequence. Most blind rendezvous algorithms utilize the labels of channels to construct CH sequences. Some blind rendezvous algorithms also utilize users' IDs in their construction of CH sequences.

Existing blind rendezvous algorithms can be categorized into different categories as follows.

(i) Synchronous/asynchronous algorithms. Synchronous algorithms [25] [21] suppose there exists a global clock and every user in the network starts rendezvous process at the same time. Asynchronous algorithms [5] [6] [3] don't require a global clock and every user can start rendezvous process at any time.

(ii) Anonoymous/non-anonymous algorithms. Anonymous algorithms [6] [13] don't require every user to have a unique ID. Non-anonymous algorithms [5] [4] suppose there exists a unique identifer (ID) for every user and take use of a user's ID to generate the channel hopping sequence.

(iii) Oblivious/non-oblivious algorithms. Oblivious algorithms [5] [4] don't require there exists a global labeling of all the channels and every user labels channels in the same way. Non-oblivious algorithms [5] [6] [13] require global labeling of channels and leverages it to guarantee rendezvous.

(iv) Global-sequence-based/local-sequence-based algorithms. Global-sequence-based algorithms [6] [23] construct channel hopping sequences containing all the channels. If a channel is not in the available channel set of a user, replace it with one available channel. Local-sequence-based algorithms [20] [19] [5]generates channel hopping sequences utilizing only the local available channels as if a user is not aware of the existence of other channels.

[1]Unless specifically stated, "users" in the rest of the paper refer to SUs.

TABLE I
Comparisons between rendezvous algorithms

Algorithm	MTTR	Time	Anonymous or Non-Anonymous	Oblivious or Non-Oblivious	Sequence	Symmetric or Asymmetric
DRDS [6]	$3P^2 + 2P$	Asynchronous	Anonymous	Non-Oblivious	Global	Asymmetric
CBH [5]	$2l_pP^2$	Asynchronous	Non-Anonymous	Oblivious	Local	Asymmetric
A-HCH-Optimal [4]*	$\approx (l + log_2 l)P_iP_j$	Asynchronous	Non-Anonymous	Oblivious	Local	Asymmetric
A-HCH-η_1 [4]	$(2l + 3)P_iP_j$	Asynchronous	Non-Anonymous	Oblivious	Local	Asymmetric
A-HCH-η_2 [4]	$(l + \lceil\sqrt{l}\rceil(2$ $+ \lceil log_2 l\rceil)P_iP_j$	Asynchronous	Non-Anonymous	Oblivious	Local	Asymmetric
MTP [7]	$2(max\{\|V_i\|, \|V_j\|\})^2$ $\times 32(\lceil loglog\ n\rceil + 1)$	Asynchronous	Anonymous	Oblivious	Local	Asymmetric
SRR(this paper)	$2P^2 + 2P$	Asynchronous	Anonymous	Oblivious	Global	Asymmetric
IDR(this paper)	$(l + 1)(P_i + 2)(P_j + 2)$	Asynchronous	Non-Anonymous	Oblivious	Local	Asymmetric
CLR(this paper)	$((P_i + 2)(P_j + 2) + P_N)$ $\times(\lceil log_2 N\rceil + 1)$	Asynchronous	Anonymous	Oblivious	Local	Asymmetric

Remark: n is the total number of channels in the network; P is the smallest prime which is not less than n; l is the length of user ID; $|V_i|$ and $|V_j|$ are the sizes of the available channel sets of user i and j respectively; P_i and P_j are the smallest primes which are not less than $|V_i|$ and $|V_j$ respectively; P_N is the smallest prime which is not less than n; * denotes that this algorithm is a centralized algorithm.

(v) Single-radio/multi-radio algorithms. Single-radio algorithms [20] [19] [5] [5] [6] [13] suppose every user in the network is equipped with only one cognitive radio. Hence, one user can once can only hop to one channel. Multi-radio algorithms [11] [24] [22] assume every user in the network is equipped with more than one cognitive radios. Hence, one user can hop to one or more channels simultaneously.

Rendezvous algorithms can be judged according to some common metrics, such as *Maximum Time To Rendezvous (MTTR)*, *Average Time To Rendezvous (ATTR)*. *Time To Rendezvous (TTR)* is the number of timeslots consumed from the last user starts rendezvous until the completion of rendezvous. ATTR is the average TTR of an algorithm. MTTR is the maximum TTR needed to achieve rendezvous. In other words, MTTR is the TTR in the worst case.

In this paper, we introduce some simple but time-efficient rendezvous algorithms utilizing local information. The following are the main contributions of our paper:

1) We propose the Sequence-Rotating Rendezvous (SRR) algorithm utilizing the label of an arbitrary available channel of a user, which has lower MTTR comparing with the extant asynchronous anonymous non-obvious global-sequence-based rendezvous algorithms;
2) We propose the ID-based Rendezvous (IDR) algorithm which has lower MTTR comparing with the extant asynchronous non-anonymous obvious local-sequence-based rendezvous algorithms;
3) We propose the Channel-Label-based Rendezvous (CLR) algorithm utilizing the label of an arbitrary available channel of a user, which has lower MTTR comparing with the extant asynchronous anonymous non-obvious local-sequence-based rendezvous algorithms;
4) We conduct plenty of experiments to compare our algorithms with state-of-the-art algorithms. The results show that our algorithms achieve better performance.

The rest of this paper is organized as follows. Section II introduces the background and some related works. Section III introduces the fundamental model and problem formulation. Section IV presents the details of SRR algorithm and analyzes its performance. Section V presents the details of IDR algorithm and analyzes its performance. Section VI presents the details of CLR algorithm and analyzes its performance. Simulation results are displayed in Section Section VII. And we conclude this paper in Section VIII.

II. Background and Related Works

A. Prime number and Co-prime Numbers

Prime number is a very important concept in number theory. We first introduce the definition of prime number:

Definition 2.1: A prime number is a natural number which is larger than 1 and can be divided with no remainder only by 1 and itself.

The smallest prime number is 2. There is an important theory about prime number:

Theorem 1: If n is a positive integer and $n \geq 2$, there must exists at least one prime number between n and 2n.

Then we introduce the definition of composite number:

Definition 2.2: A composite number is a natural number which is larger than 1 and has factors besides 1 and itself.

The smallest composite number is 4. There is a property about prime and composite numbers:

Lemma 2.1: Any positive integer larger than 1 is either a prime number or a composite number.

Then we introduce the definition of co-prime numbers:

Definition 2.3: Two nonzero natural integers a and b are said to be co-prime if the only common factor which could divide them with no remainder is 1.

Co-prime numbers have many important properties:

Lemma 2.2: If a and b are co-prime, the least common multiple of them is their product: $a \times b$.

Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
mod 4	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0
mod 5	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0

Fig. 1. An example of Theorem 2.

Lemma 2.3: If a and b are two consecutive positive integers, a and b are co-prime.

Lemma 2.4: If a and b are two consecutive positive odd numbers, a and b are co-prime.

Lemma 2.5: Suppose a is a prime number and b is a composite number. If a is larger than b, they are co-prime. If b is larger than a, but b is not a multiple of a, they are co-prime.

We then introduce an important theorem for rendezvous:

Theorem 2: If m and n are co-prime numbers, then for any integer a, the integers $a, a+n, a+2n, \cdots, a+(m-1)n$ are m distinct numbers under modulo-m arithmetic.

Proof: Choose any two integers from $a, a+n, a+2n, \cdots, a+(m-1)n$, the absolute value of their difference is kn, where $0 < k < m$. If kn mod m equal 0, then $kn = lm$, where l is a positive integer. Then kn or lm is a common multiple of m and n. From Lemma 2.2 we know mn is the least common multiple of m and n, and because $0 < k < m$, kn is a common multiple of m and n that is less than mn, which is a contradiction. Hence, Theorem 2 holds. ■

Fig. 1 illustrates an example of Theorem 2. In Fig. 1, the first row is a string of numbers starting from 1. The second row is the results of these numbers under modulo 4 arithmetic. The third row is the results of these numbers under modulo 5 arithmetic. We can clearly see that 1, 5, 9, 13, 17 all correspond to 1 under modulo 4 arithmetic, but correspond to 1, 0, 4, 3, 2 respectively under modulo 5 arithmetic, which are all the possible results under modulo 5 arithmetic. Theorem 2 plays an important role in rendezvous algorithms and is the main tool we use to design the following algorithms.

B. Related Works

Rendezvous algorithms can be classified into different categories according to different standards. One method is to divide rendezvous algorithms into Global-sequence-based rendezvous algorithms and Local-sequence-based rendezvous algorithms.

(i) Global-sequence-based rendezvous algorithms.

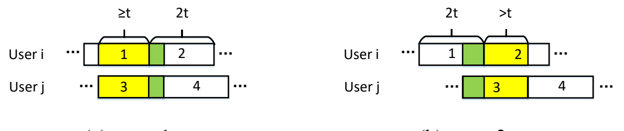
Global-sequence-based rendezvous algorithms design channel hopping sequence based on all the channels in the network. When a channel is not available to a user, the user replaces it with an arbitrary available channel. Jump-Stay (JS) [14] algorithm is a representative global-sequence-based algorithm which is composed of two patterns: jump-pattern and stay-pattern. JS algorithm uses the thought of number theory and has an MTTR of $O(n^3)$ where n is the number of channels in the network. Enhanced JS [13]algorithm is an enhanced version of JS algorithm which improves the MTTR to $O(n^2)$ level. Disjoint Relaxed Difference Set (DRDS) [6] algorithm is another global-sequence-based rendezvous algorithm. The key of DRDS algorithm is to construct a disjoint relaxed difference set. DRDS algorithm can guarantee rendezvous in $O(n^2)$ timeslots. Channel Rendezvous Sequence (CRSEQ) [18] algorithms utilizes the triangle number to generate a channel hopping sequence which also guarantee rendezvous in $O(n^2)$ timeslots. DSC-based Rendezvous (DSCR) [23] algorithm is based on Disjoint Set Cover (DSC), which is an NP-hard problem. DSCR problem generates channel hopping sequences by invoking an approximation construction algorithm of DSC. DSCR algorithm has an MTTR of $(2P + \lfloor \frac{P}{2} \rfloor)(P - G + 1)$ timeslots, where G is the number of common channels that the two users share.

(ii) Local-sequence-based algorithms.

Local-sequence-based algorithms generate channel hopping sequences based on the set of available channels of a user. Heterogeneous Hopping (HH) [18] algorithm is a representative local-sequence-based rendezvous algorithm which is composed of three subsequences named fixed, rotating and parity sequence respectively. HH algorithm uses the thought of number theory and has an MTTR of $O(|C_a||C_a|)$ where $|C_a|$ and $|C_b|$ denotes the sizes of the sets of continuous sensible channels of user a and b respectively. Interlocking Channel Hopping (ICH) [19] algorithm is based on HH algorithm and takes the congestion problem into consideration. Conversion Based Hopping (CBH) [5] algorithm is a local-sequence-based rendezvous algorithm which utilizes user' ID. CBH algorithm is also applicable in oblivious scenarios. CBH algorithm has an MTTR of $2l_p \max\{P_i^2, P_j^2\}$ where l_p is determined by the IDs of user i,j and P_i,P_j are the smallest primes which is not less than the sizes of available channel sets of user i and j respectively. A-HCH [4] algorithm is another local-sequence-based rendezvous algorithm which utilizes ID. Adv.rdv algorithm constructs channel hopping sequences by generating slow and fast channel hopping sequences and interleaving them. Adv.rdv algorithm has three versions, which are Adv.rdv-Optimal [4], Adv.rdv-η_1 [4] and Adv.rdv-η_2 [4]. The difference of them is the construction of symmetrization classes. Adv.rdv-Optimal is a centralized algorithm which is not applicable to distributed system. Moving Traversing Pointers (MTP) [7]algorithm utilizes two pointers to generate channel hopping sequences and can guarantee rendezvous in $2(max\{|V_i|, |V_j|\})^2 \times 32(\lceil loglogn \rceil + 1)$ timeslots.

III. Model and Problem Formulations

In this section, we will introduce the foundation model and some other models which are based on the foundation model. Models are of great importance in the design of rendezvous algorithms, because models constrain the resources that we can use. Hence, we need to design different algorithms under different models to achieve high performance for rendezvous process. We will also give the formulation for rendezvous problem in this section.



(a) case 1 (b) case 2

Fig. 2. An example of different cases of timeslot overlapping.

A. Foundation Model

Suppose the licensed spectrum is divided into n channels which are non-overlapping. Denote the set of channels as $U = \{1, 2, \cdots, n\}$, where n is the total number of channels in U. We assume that every user in the network is equipped with a CRN. User i has a set of available channels as $V_i = \{v_{i1}, v_{i2}, \cdots, v_{im_i}\}$, where m_i is the number of channels in set V_i. For simplicity, we suppose that V_i doesn't change during the rendezvous process. Let the length of every timeslot be 2t, where $t = 10ms$ according to IEEE 802.22.

Because the lengths of all users' timeslots are equal, user j's one timeslot can overlap with at most two consecutive timeslots of user i. Fig. 2 illustrates this problem in two cases. Suppose timeslot 3 is one timeslot of user j. And the left timeslot of user i which overlaps with timeslot 3 is timeslot 1. The right timeslot of user i which overlaps with timeslot 3 is timeslot 4. The yellow part stands for the overlapping part of timeslot 1 and 3, while the green part stands for the overlapping part of timeslot 2 and 3. In Fig. 2(a), the length of the yellow part is greater than or equal to t. In Fig. 2(b), the length of the green part is greater than t. There is a special case that the length of the yellow part or the green part is zero, which means that the timeslots of user i and user j are aligned. Hence, no matter when the users start their rendezvous process, the maximum overlap of their timeslots is at least t, which is enough for two users to find each other and exchange information.Foundation model is the basis of other models we will introduce in the following subsections.

B. Problem Formulations

In this paper, we focus on designing rendezvous algorithms for two-user scenario, because it is the fundamental process for multiple-user rendezvous. Two-user rendezvous algorithms can be expanded into multiple-user rendezvous algorithms. Then we give the formulations of the three problems we will solve:

Problem 1: Utilize an arbitrary available channel of a user to design an asynchronous anonymous non-oblivious global-sequence-based rendezvous algorithm with low MTTR.

Problem 2: Utilize the ID of a user to design an asynchronous non-anonymous oblivious local-sequence-based rendezvous algorithm with low MTTR.

Problem 3: Utilize an arbitrary available channel of a user to design an asynchronous anonymous non-oblivious local-sequence-based rendezvous algorithm with low MTTR.

IV. Sequence-Rotating Rendezvous Algorithm

In this section, we will propose a global-sequence-based rendezvous algorithm which is based on the channel label.

We first propose an algorithm to generate the subsequences to be used.

Algorithm 1 Subsequence Generating Algorithm

```
1: Input: the set of available channels $V = \{v_1, v_2, \cdots, v_m\}$,
   and length $L$;
2: Initialize array $S$ with length $L$;
3: Initialize $i := 0$;
4: while $i < L$ do
5:     if $i \leq m$ then
6:         $S[i] := v_i$;
7:     else
8:         Randomly choose a channel from $V$, denote it as $c$;
9:         $S[i] := c$;
10:    end if
11:    $i := i + 1$;
12: end while
13: Output: Sequence $S$.
```

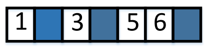
The main idea of Alg.1 is to generate a subsequence S with length L according to V. The process is rather simple. We fill the first m elements in S with the channels in V. For the rest $(L - m)$ elements, we randomly pick channels from V to fill them.

Algorithm 2 Sequence-Rotating Rendezvous Algorithm

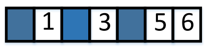
```
1: Input: the set of available channels $V = \{v_1, v_2, \cdots, v_m\}$
   and the total number $N$ of all channels;
2: Find the least prime $P$ that ensure $P \geq N$;
3: Randomly choose a channel from $V$, denote it as channel
   $(c)$;
4: Initialize $t := 0$;
5: while $0 \leq t < 2P$ do
6:     Access channel $(c)$;
7:     $t := t + 1$;
8: end while
9: Invoke Alg.1 to generate a subsequence $S$ with channel
   set $V$ and length $P$;
10: while Not rendezvous do
11:    if $t \neq 2P$ and $t$ mod $(2P) = 0$ then
12:        Rotate $S$ to the right by $c$ steps;
13:    end if
14:    Access channel $S[t \bmod P]$;
15:    $t := t + 1$
16: end while
```

In Alg. 2, we first find the least prime which is not less than the total number of all channels. Then we randomly choose a channel from V, denote it as channel (c), and access this channel for the first $2P$ timeslots. We call this the **first stage** , the following is the **second stage** .

Then, we will generate a subsequence S with length P. If channel (a) is in the available channel set V, the $a - th$



(a) Original subsequence



(b) The first rotated subsequence

Fig. 3. An example of generated subsequence when $V = \{1, 3, 5, 6\}$, $N = 6$ and the first rotated subsequence.

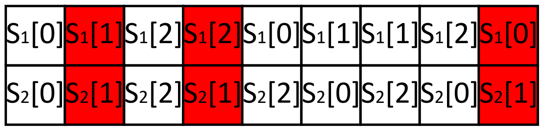

S1[0] S1[1] S1[2] S1[2] S1[0] S1[1] S1[1] S1[2] S1[0]
S2[0] S2[1] S2[2] S2[1] S2[2] S2[0] S2[2] S2[0] S2[1]

Fig. 4. An example of Theorem 3.

channel in S is channel (a). For the channels that are not exist in V, their places will be replaced by channels which are randomly picked from V. And the last $(P - n)$ channels in S are randomly picked from V. The period is $2P$ timeslots. Then we will rotate the sequence by c steps to the right every period. In every period, we will hop to channels according to rotated sequence.

Fig.3 illustrates an example of the original generated subsequence and the first rotated subsequence. In this example, $V = \{1, 3, 5, 6\}$ and $n = 6$, so $P = 7$. We generate a sequence as Fig.3(a), where the blue rectangles correspond to randomly picked channels. And a rectangle with number in it, denote it as i, represents the Channel (i). Because the least label in V is 1, so the sequence is rotated by 1 step to the right every period. The first rotated sequence is shown in Fig.3(b).

We then introduce a theorem which will be used in the later discuss.

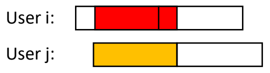
Theorem 3: Suppose there are two aligned subsequences S_1 and S_2 with identical length P and P is a prime. Suppose $0 \leq k_1 \leq P, 0 \leq k_2 \leq P$ and $k_1 \neq k_2$. If in every period we rotate S_1 and S_2 to the right by k_1 and k_2 steps respectively and concatenate them to S_1 and S_2 respectively. Then after P periods, every element of S_1 will meet every element of S_2 and vice versa.

Proof: Because S_1 and S_2 are aligned, then $S_1[i]$ will meet $S_2[i]$ in the first period, where $0 \leq i < P$. Without lose of generality, suppose $k_1 > k_2$. Let $k_1 - k_2 = m$, then in the later $(P - 1)$ periods, $S_1[i]$ will meet $S_2[(i + m) \bmod P]$, $S_2[(i + 2m) \bmod P]$, $\cdots$, $S_2[(i + (P - 1)m) \bmod P]$ respectively. According to Theorem 2, $i, i+m, i+2m, \cdots, i+(P-1)m$ are P distinct numbers under modulo-P arithmetic. Hence, $S_1[i]$ meets all the elements of S_2 in P periods and Theorem 3 holds. ■

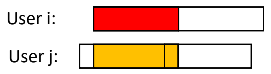
Fig.4 illustrates an example of Theorem 3, where the length of S_1 and S_2 is 3. S_1 is rotated to the right by 1 steps every period, and S_2 is rotated to the right by 2 steps every period. We can clearly see that $S_2[1]$ meets $S_1[1], S_1[2], S_1[0]$ in the three consecutive periods.

Theorem 4: The MTTR of Alg.2 is $2P^2 + 2P$.

Proof: We show discuss this problem in two cases.



(a) Situation 1



(b) Situation 2

Fig. 5. Two situations of periods overlapping of user i and j

Case 1 : Suppose two users choose the same channel, denote it as channel c. Without loss of generality, suppose user i starts rendezvous process earlier than user j. For clarity, we divide this case into two subcases.

Subcase 1.1 : Suppose user j starts rendezvous process when user i is still in the first stage, they will both access channel c at the same time. Hence, rendezvous is achieved.

Subcase 1.2 : Suppose user j starts rendezvous process when user i is already in the second stage. Then in the following $2P$ timeslots, user i will access each available channel including channel c at least once, while user j will stay on channel c in the $2P$ timeslots. Hence, rendezvous can be achieved in the $2P$ timeslots.

Case2 : Suppose user i and j choose different channels, denote them as channel (c_i) and channel (c_j). When user i and j both get into the second stage, they will both rotate their sequences according to it. There are two situations of periods overlapping of user i and j, we will discuss them separately.

Subcase 2.1 : Suppose the periods of user i and j are aligned, then the scenario is the same as the setting in Theoren 2 except for that we duplicate the current subsequences before rotate it and attach them to the corresponding sequences. The difference result in double time delay. According to Theorem 3, rendezvous will be achieved in $2P^2 + 2P$ timeslots.

Subcase 2.2 : Suppose the periods of user i and j are not aligned. There are two situations of periods overlapping of user i and j as illustrated in Fig.5. The yellow rectangle in Fig.5(a) represents the first half of user j's period, while the red rectangle represents the corresponding part in user i's period. The red rectangle includes all the elements of subsequence S_i, because the two halves of user i's period are the same rotated subsequence of S_i. Then this situation is similar to Subcase 2.1, and according to Theorem 3, rendezvous will be achieved in $2P^2+2P$ timeslots. Situation 2 can be analysed in the same way, so we omit it here.

Combine these together, we have shown that Theorem 4 holds. ■

V. ID-based Rendezvous Algorithm

In this section, we will propose an ID-based asynchronous non-anonymous oblivious rendezvous algorithm. The main idea of this algorithm is to create a channel hopping sequence composed of $(l + 1)$ subsequences, where l is the length of ID.

In Alg.3, we creates a channel hopping sequence consisting of $(l+1)$ subsequences. The $(l+1)$ subsequences are generated by Alg.1. There are three kinds of subsequences, which are S_1, S_2 and S_3. The lengths of S_1, S_2 and S_3 are P, P+1, P+2 respectively.

Algorithm 3 Local-sequence-based Non-Anonymous Rendezvous Algorithm

```
1:  Input: the set of available channels $V = \{v_1, v_2, \cdots, v_m\}$,
    and its binary ID whose length is $l$ bits;
2:  Find the smallest prime $P$ that ensure $P \geq m$;
3:  if $P = 3$ then
4:      $P := 5$;
5:  end if
6:  Initialize $t := 0, i := 0, j := 0$;
7:  Invoke Alg.1 to generate three sequences $S_1, S_2, S_3$ with
    channel set V and length $P, P+1, P+2$ respectively;
8:  Find the channel with the least label in $V$, denote it as
    channel $(c)$;
9:  while Not rendezvous do
10:     $i := \lfloor t/(l+1) \rfloor$;
11:     $j := t \bmod (l+1)$;
12:     if $j < l$ then
13:         if ID$[j] = 0$ then
14:             Access channel $S_2[i \bmod (P+1)]$;
15:         else
16:             Access channel $S_3[i \bmod (P+2)]$;
17:         end if
18:     else
19:         Access channel $S_1[i \bmod P]$;
20:     end if
21:     $t := t + 1$;
22: end while
```

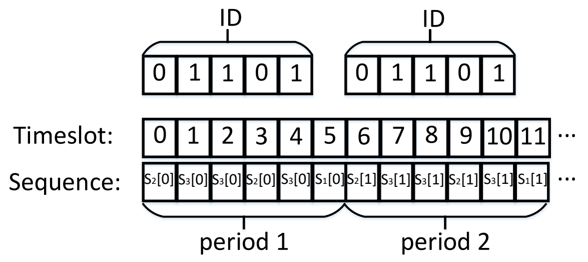


Fig. 6. An example of Alg. 3.

Fig.6 illustrates an example of Alg.3. In this example, the length of ID is 5 and user i's ID is 01101. In round 1, the first l elements of round 1 are either $S_2[0]$ or $S_3[0]$. If the corresponding digit of the i-th element in ID is 0, then the i-th element will be $S_2[0]$. If the corresponding digit of the i-th element in ID is 1, then the i-th element will be $S_3[1]$. The last element of round 1 is $S_1[0]$.

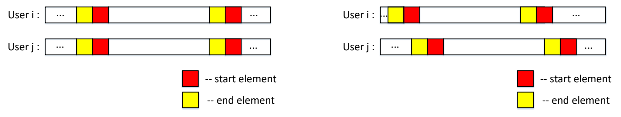
Theorem 5: Alg.3 can guarantee rendezvous for two asynchronous users in $(l+1)(P_i+2)(P_j+2)$ timeslots if $V_i \cap V_j \neq \emptyset$.

Proof: We discuss in two cases:

Case 1 :

Suppose $P_i = P_j$. Let $P = P_i = P_j$ We divide this case into two subcases:

Subcase 1.1 : Suppose user i and j's periods are aligned,



(a) subcase 1 (b) subcase 2

Fig. 7. Two subcases of case 1 and case 2.

as illustrated in Fig.7(a). Because user i and user j are two different users, their ID must be different in at least one digit. Without lose of generality, suppose the k-th digit in ID_i is 0 while the k-th digit in ID_j is 1. Then the k-th element in user i's period corresponds to subsequence S_{2i}, whose length is $P+1$. The k-th element in user j's period corresponds to subsequence S_{3j}, whose length is $P+2$. Then in every period, one element of S_{2i} will meet one element of S_{2j}. According to Lemma 2.3, P+1 and P+2 are co-prime. Then rendezvous can be achieved in $(l+1)(P+1)(P+2)$ timeslots according to Theorem 2, as long as user i and j have at least one common channel.

Subcase 1.2 : Suppose user i and j's periods are not aligned, as illustrated in Fig.7(b). In this case, the last element in user i's period will meet one of the first $(l+1)$ elements of user j's period. The last element in user i's period corresponds to subsequence S_{1i}. The corresponding element of user j corresponds to subsequence S_{2j} or S_{3j}. From Lemma 2.3 and 2.4, we know that P and P+1 are co-prome, P and P +2 are also co-prime. Hence, according to Theorem 2, rendezvous can be achieved in $(l+1)P(P+1)$ or $(l+1)P(P+2)$ timeslots respectively.

Case 2 : Suppose $P_i \neq P_j$. We divide this case into two subcases:

Subcase 2.1 : Suppose user i and j's periods are aligned, as illustrated in Fig.7(a). In this case, the last element in user i's period will meet the last element in user j's period. The last element in user i's period corresponds to subsequence S_{i1}. The last element in user j's period corresponds to subsequence S_{j1}. Because $P_i \neq P_j$, rendezvous can be achieved in $(l+1)P_iP_j$ timeslots according to Theorem 2, as long as user i and j have at least one common channel.

Subcase 2.2 : Suppose user i and j's periods are not aligned, as illustrated in Fig.7(b). Without lose of generality, suppose $P_i > P_j$. In this case, the last element in user i's period will meet one of the first $(l+1)$ elements of user j's period. The last element in user j's period will meet one of the first $(l+1)$ elements of user i's period. The last element in user j's period corresponds to subsequence S_{1j}, whose length is P_j. The corresponding element of user j corresponds to subsequence S_{2i} or S_{3i}, whose lengths are P_i+1 and P_i+2 respectively. Because P_i and P_j are two different primes, $P_i > P_j+2$ or $P_i = P_j+2$. If $P_i > P_j+2$, from Lemma 2.5, we know P_i and P_j+1 are co-prime, P_i and P_j+2 are co-prime. According to Theorem 2, rendezvous can be achieved in $(l+1)P_i(P_j+1)$ or $(l+1)P_i(P_j+2)$ timeslots respectively. If $P_i = P_j+2$, then $P_i+1 = P_2+3$, $P_i+2 = P_j+4$. If $P_j = 2$, $P_i = 2+2 = 4$

is not a prime. In Alg.3, we set P to be 5 if $P = 3$. Because $P_2 \neq 2$ and $P_2 \neq 3$, $P_2 + 3$ and $P_2 + 4$ can not be divided by P_2 without remainder. From Lemma 2.5, we know that both $P_i + 1$ and $P_i + 2$ are co-prime with P_j. According to Theorem 2, rendezvous can be achieved in $(l+1)(P_i+1)P_j$ or $(l+1)(P_i+2)P_j$ timeslots respectively.

Combine the above together, Theorem 5 holds.

■

VI. Channel-Label-based Rendezvous Algorithm

In this section, we will propose a Channel-Label-based Rendezvous (CLR) algorithm which utilizes the label of an arbitrary channel in the available channel set.

Algorithm 4 Channel-Label-based Rendezvous Algorithm

```
1: Input: the set of available channels $V = \{v_1, v_2, \cdots, v_m\}$
   and the number of all channels: $N$;
2: Find the smallest prime $P_N$ that ensure $P_N \geq N$ and
   $P_N \geq 5$;
3: Find the smallest prime $P$ that ensure $P \geq m$ and $P \geq 5$;
4: Initialize $l := \lceil log_2 N \rceil$;
5: Initialize $t := 0$;
6: Randomly choose a channel $(c)$ from $V$;
7: Convert $c$ into a binary string $CH$ whose length is $l$;
8: while Not rendezvous and $t < (l+1)P_N$ do
9:     Access channel $(c)$;
10: end while
11: Invoke Alg.3 to achieve rendezvous with channel set $V$,
    binary string $CH$ and length $l$;
```

The main idea of Alg.4 is to randomly choose an available channel c from V and use the binary representation of c as the user's ID. It is obvious that the available channel chosen by two users can be identical, but our algorithm can solve this problem, because the label of an available channel is very useful. Alg.4 is composed of two phases, the first phase is the first $(l+2)P_N$ timeslots, the following is the second phase. In the **first phase** , the user stays on the chosen channel (c). In the **second phase** , the user invoke Alg.3 to achieve rendezvous with the binary representation of c as ID.

Theorem 6: Alg. 4 can guarantee rendezvous for two asynchronous users i and j in $((P_i+2)(P_j+2)+P_N)(\lceil log_2 N \rceil+1)$ timeslots as long as $V_i \cap V_j \neq \emptyset$.

Proof:

We discuss this problem in two cases.

Case 1: Suppose the channels c_i and c_j chosen by user i and j respectively are different, then the corresponding binary representations of c_i and c_j are different in at least one digit. Therefore, they can replace the roles of IDs for user i and j respectively. Hence, by invoking Alg.3, rendezvous can be achieved in $((P_i+2)(P_j+2)+P_N)(\lceil log_2 N \rceil+1)$ timeslots.

Case 2: Suppose the channels c_i and c_j chosen by user i and j respectively are the same, let $c = c_i = c_j$. Then channel (c) must be a common channel of user i and j. Without lose of generality, suppose user i starts rendezvous process earlier than user j. We then divide this case into two subcases.

Subcase 2.1 Suppose when user j starts rendezvous process, user i is still in the **first phase** . Then user j will hop to channel (c) in its first timeslot and user i is also in channel (c), hence rendezvous is achieve is achieved in 1 timeslot.

Subcase 2.2 Suppose when user j starts rendezvous process, user i is already in the **second phase** . Because user j stays on channel (c) for $P_N(\lceil log_2 N \rceil + 1)$ timeslots, and $P_N \geq P_i$, hence channel (c) will meets every elements of user i's subsequence S_1 (defined in Alg.3). Therefore, rendezvous will be achieved in the **first phase** of user j.

Combine these together, Theorem 6 holds.

■

VII. Simulation

In this section, we will show the simulation results of comparing our algorithms with state-of-the-art algorithms. We implemented the algorithms in C++ language. And for each experiment, we run the corresponding algorithm for 10000 times.

We first compare our SRR algorithm with a representative asynchronous non-anonymous non-oblivious global-sequence-based rendezvous algorithm – DRDS. Denote R_i and R_j as the ratios of $|V_i|$ and $|V_j|$ to n respectively. First, we set $V_i = \{0.3n, 0.3n+1, \cdots, 0.5n\}, V_j = \{0.5n, 0.5n+1, \cdots, 0.7n-1\}$ where n is the total number of channels in the network. In this case, $R_i = R_j = 0.3$. We set the number of common channels to be 1 in order to reduce the impact of the randomly channel hopping in the both algorithms. We increase n from 10 to 100 by 10 steps each time. The result is illustrated in Fig.9, from which we can clearly see that the MTTR of SRR grows slower than that of DRDS. Then we set $V_i = \{1, 2, \cdots, 0.5n\}$ and $V_j = \{0.5n, 0.5n+1, \cdots, n-1\}$. In this case, $R_i = R_j = 0.5$. The result is illustrated in Fig. 8. We can clearly see that the MTTR of SRR grows slower than that of DRDS. And we can find the ratio of the MTTR of DRDS to that of SRR in Fig.9 is larger than that in Fig.8 and more close to the theoretical analysis. The reason is the ratio of the number of common channel to the total channel number decreases with the increase of R_i and R_j, which makes it harder for user i and j to choose the same channel simultaneously.

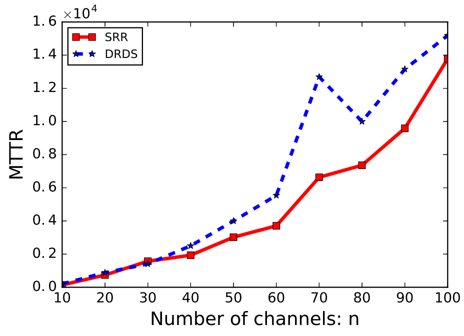


Fig. 8. Comparison between the SRR algorithm and the DRDS algorithm when N increases from 10 to 100 and $R_i = R_j = 0.2$.

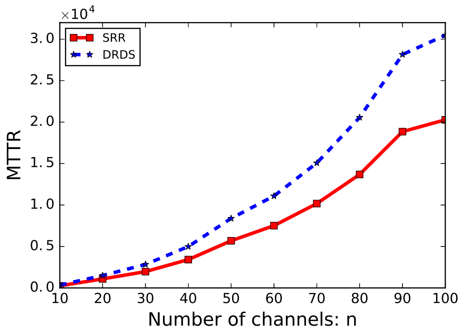


Fig. 9. Comparison between the SRR algorithm and the DRDS algorithm when N increases from 10 to 100 and $R_i = R_j = 0.5$.

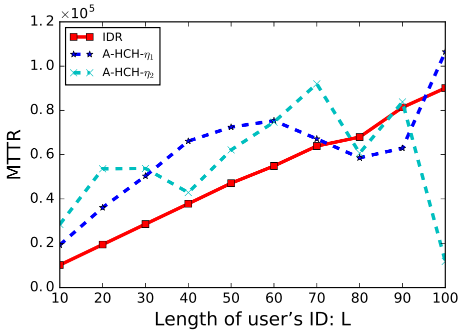


Fig. 10. Comparison between the IDR algorithm and the A-HCH-η_1/η_2 algorithms when l increases from 10 to 100 and $N = 50$.

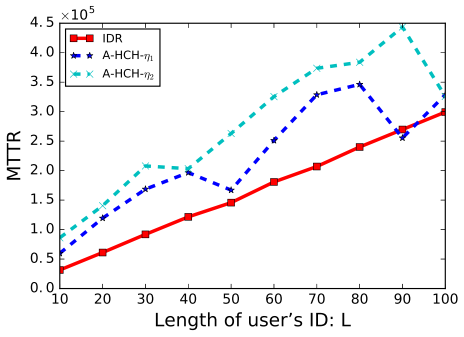


Fig. 11. Comparison between the IDR algorithm and the A-HCH-η_1/η_2 algorithms when l increases from 10 to 100 and $N = 100$.

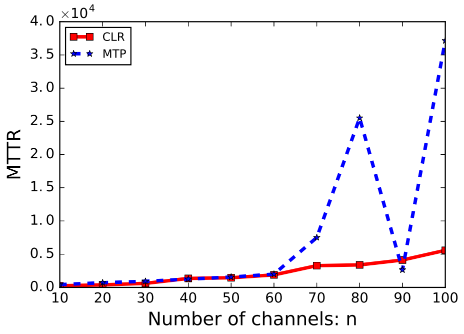


Fig. 12. Comparison between the CLR algorithm and the MTP algorithm when N increases from 10 to 100 and $R_i = R_j = 0.2$.

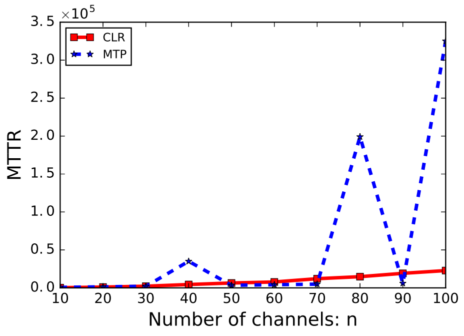


Fig. 13. Comparison between the CLR algorithm and the MTP algorithm when N increases from 10 to 100 and $R_i = R_j = 0.5$.

Second, we conduct experiments to compare the MTTRs of the IDR algorithm and the state-of-the-art asynchronous non-anonymous oblivious local-sequence-based algorithms – A-HCH-η_1 and A-HCH-η_2. For this kind of rendezvous algorithm, the crux is the relationship between the increase of MTTR with the increase of the length of user' ID. Hence, we conduct experiments which fixes R_i, R_j and N but increases the length of ID l step by step. We first fix $R_i = R_j = 0.5$ by setting $V_i = \{1, 2, \cdots, 0.5n\}$ and $V_j = \{0.5n, 0.5n+1, \cdots, n-1\}$. We fix $N = 50$ and increase l from 10 to 100 by 10 steps each time. The result is illustrated in Fig.10. We can clearly see that the MTTR of IDR is less than that of A-HCH-η_1 or A-HCH-η_2 in general. Then we repeats this experiment except for increasing N to 100. The result is illustrated in Fig.11. We can clearly see that the MTTR of IDR is less than that of A-HCH-η_1 or A-HCH-η_2 in general.

Third, we implement experiments to compare our CLR algorithm with the state-of-the-art asynchronous anonymous non-oblivious local-sequence-based algorithm – MTP. MTP has a theoretical MTTR of $O(|V_i||V_j|loglogn)$. We first set $R_i = R_j = 0.2$ by setting $V_i = \{0.3n, 0.3n+1, \cdots, 0.5n\}, V_j = \{0.5n, 0.5n+1, \cdots, 0.7n-1\}$. We increase n from 10 to 100 by 10 steps each time. The result is illustrated in Fig.12. We can clearly see that the MTTR of CLR is less than that of MTP in general, and the MTTR of MTP on some experiment points is very large. This is because the MTP algorithm is designed to be relatively complicated in order to achieve MTTR in $O(|V_i||V_j|loglogn)$ order, however, it results in a large constant coefficient. Hence, the MTTR of MTP algorithm can be very high, result in losing superiority. We implement another experiment by increasing R_i and R_j to 0.5, the result is illustrated in Fig.13. We can see that the situation is similar to that in Fig.12.

VIII. Conclusion

In this paper, we first propose an Sequence-Rotating Rendezvous (SRR) algorithm, which utilizes the label of an arbitrary available channel of a user as local information to generate channel hopping sequence. SRR algorithm uses the thought of number theory and has an MTTR of $(2P^2 + 2P)$ timeslots. Second, we propose an Asynchronous Non-anonymous

Oblivious Local-sequence-based (ANOL) rendezvous algorithm, which utilizes the user's ID to generate channel hopping sequence. ANOL algorithm also uses the thought of number theory and has an MTTR of $(l+1)(P_i+2)(P_j+2)$ timeslots, where l is the length of ID and P_i and P_j are the smallest primes which are not less than the size of available channel sets of user i and j respectively. Third, we propose an Asynchronous Anonymous Non-Oblivious Local-sequence-based (AANL) algorithm, which utilizes the binary representation of an arbitrary available channel of a user as the user's ID. AANL algorithm combines the thought of AANG algorithm and ANOL algorithm and has an MTTR of $((P_i+2)(P_j+2)+P_N)(\lceil log_2 N\rceil+1)$ timeslots. We conducted plenty of experiments comparing our algorithms with state-of-the-art rendezvous algorithms and the results show our algorithms can achieve good performance.

References

[1] Kaigui Bian, Jung Min Park, and Ruiliang Chen. A quorum-based framework for establishing control channels in dynamic spectrum access networks. In *International Conference on Mobile Computing and Networking, MOBICOM 2009, Beijing, China, September*, pages 25–36, 2009.

[2] Kaigui Bian, Jung Min Park, and Ruiliang Chen. Control channel establishment in cognitive radio networks using channel hopping. *IEEE Journal on Selected Areas in Communications*, 29(4):689–703, 2011.

[3] Meenu Chawla, Aishwarya Sagar Anand Ukey, and P. Reshma. Comprehensive asynchronous symmetric rendezvous algorithm in cognitive radio networks. *Sadhana*, pages 1–10, 2017.

[4] Lin Chen, Kaigui Bian, Lin Chen, Cong Liu, Jung Min Jerry Park, and Xiaoming Li. A group-theoretic framework for rendezvous in heterogeneous cognitive radio networks. In *ACM International Symposium on Mobile Ad Hoc NETWORKING and Computing*, pages 165–174, 2014.

[5] Zhaoquan Gu, Qiang Sheng Hua, and Weiguo Dai. Fully distributed algorithms for blind rendezvous in cognitive radio networks. In *ACM International Symposium on Mobile Ad Hoc NETWORKING and Computing*, pages 155–164, 2014.

[6] Zhaoquan Gu, Qiang Sheng Hua, Yuexuan Wang, and Francis C. M. Lau. Nearly optimal asynchronous blind rendezvous algorithm for cognitive radio networks. In *Sensor, Mesh and Ad Hoc Communications and Networks*, pages 371–379, 2013.

[7] Zhaoquan Gu, Haosen Pu, Qiang Sheng Hua, and Francis C M. Lau. Improved rendezvous algorithms for heterogeneous cognitive radio networks. In *Computer Communications*, pages 154–162, 2015.

[8] J. G. Jia, Z. W. He, J. M. Kuang, and H. F. Wang. Analysis of key technologies for cognitive radio based wireless sensor networks. In *International Conference on Wireless Communications NETWORKING and Mobile Computing*, pages 1–5, 2010.

[9] Juncheng Jia, Qian Zhang, and Xuemin Shen. Hc-mac: A hardware-constrained cognitive mac for efficient spectrum management. *Selected Areas in Communications IEEE Journal on*, 26(1):106–117, 2008.

[10] Yogesh R. Kondareddy, Prathima Agrawal, and Krishna Sivalingam. Cognitive radio network setup without a common control channel. In *Military Communications Conference, 2008. Milcom*, pages 1–6, 2009.

[11] Guyue Li, Zhaoquan Gu, Xiao Lin, Haosen Pu, and Qiang Sheng Hua. Deterministic distributed rendezvous algorithms for multi-radio cognitive radio networks. In *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 313–320, 2014.

[12] Ming Li, Pan Li, Miao Pan, and Jinyuan Sun. Economic-robust transmission opportunity auction in multi-hop wireless networks. In *INFOCOM, 2013 Proceedings IEEE*, pages 1842–1850, 2013.

[13] Zhiyong Lin, Hai Liu, Xiaowen Chu, and Yiu Wing Leung. Enhanced jump-stay rendezvous algorithm for cognitive radio networks. *IEEE Communications Letters*, 17(9):1742–1745, 2013.

[14] Hai Liu, Zhiyong Lin, Xiaowen Chu, and Yiu-Wing Leung. Jump-stay rendezvous algorithm for cognitive radio networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(10):1867–1881, 2012.

[15] Liangping Ma, Xiaofeng Han, and Chien Chung Shen. Dynamic open spectrum sharing mac protocol for wireless ad hoc networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, pages 203–213, 2005.

[16] Jia Min, Xinyu Wang, Qing Guo, and Xuemai Gu. A novel multi-bit decision adaptive cooperative spectrum sensing algorithm based on trust valuations in cognitive ofdm system. In *Vehicular Technology Conference*, pages 1–5, 2014.

[17] J Perez-Romero, O Sallent, R Agusti, and L Giupponi. A novel on-demand cognitive pilot channel enabling dynamic spectrum allocation. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, pages 46–54, 2007.

[18] Jongmin Shin, Dongmin Yang, and Cheeha Kim. A channel rendezvous scheme for cognitive radio networks. *IEEE Communications Letters*, 14(10):954–956, 2010.

[19] Ching Chan Wu and Shan Hung Wu. On bridging the gap between homogeneous and heterogeneous rendezvous schemes for cognitive radios. In *Fourteenth ACM International Symposium on Mobile Ad Hoc NETWORKING and Computing*, pages 207–216, 2013.

[20] Shan Hung Wu, Ching Chan Wu, Wing Kai Hon, and Kang G Shin. Rendezvous for heterogeneous spectrum-agile devices. *Proceedings - IEEE INFOCOM*, pages 2247–2255, 2014.

[21] Tsung Ying Wu, Wanjiun Liao, and Cheng Shang Chang. Cach: Cycle-adjustable channel hopping for control channel establishment in cognitive radio networks. In *INFOCOM, 2014 Proceedings IEEE*, pages 2706–2714, 2014.

[22] Bo Yang, Wei Liang, Meng Zheng, and Ying Chang Liang. Fully distributed channel-hopping algorithms for rendezvous setup in cognitive multiradio networks. *IEEE Transactions on Vehicular Technology*, 65(10):8629–8643, 2016.

[23] Bo Yang, Meng Zheng, and Wei Liang. A time-efficient rendezvous algorithm with a full rendezvous degree for heterogeneous cognitive radio networks. In *IEEE INFOCOM 2016 - the IEEE International Conference on Computer Communications*, pages 1–9, 2016.

[24] Lu Yu, Hai Liu, Yiu Wing Leung, and Xiaowen Chu. Adjustable rendezvous in multi-radio cognitive radio networks. In *IEEE Global Communications Conference*, pages 1–7, 2015.

[25] Yifan Zhang, Gexin Yu, Qun Li, Haodong Wang, Xiaojun Zhu, and Baosheng Wang. Channel-hopping-based communication rendezvous in cognitive radio networks. *IEEE/ACM Transactions on Networking*, 22(3):889–902, 2014.