Computer Communications 79 (2016) 66–77

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

A periodic jump-based rendezvous algorithm in cognitive radio networks

Saber Salehkaleybar*, Mohammad Reza Pakravan

Data Network Research Lab, Faculty of Electrical Engineering, Sharif University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 29 January 2015
Revised 18 July 2015
Accepted 16 January 2016
Available online 22 January 2016

Keywords:
Opportunistic Spectrum Access (OSA)
Blind rendezvous
Multichannel Medium Access Control (MAC) protocols

ABSTRACT

An important issue in designing multichannel MAC protocols for Opportunistic Spectrum Access (OSA) is the synchronization between Secondary Users (SUs). Synchronization can be performed in two phases: the initial handshaking, and then the synchronous hopping across available channels. In this paper, we address the problem of initial handshaking through the approach called "blind rendezvous". We first introduce a role-based solution by constructing two channel hopping sequences. The secondary transmitter and receiver jump across channels according to these two sequences. The proposed algorithm guarantees rendezvous in at most $(C+1)^2$ time slots (where C is the number of channels) and two SUs have a chance to rendezvous in each successive period of C time slots. We show that this property of constructed sequences leads to a lower average time-to-rendezvous (TTR) for the proposed algorithm in comparison to other related works. We also devise a solution to ensure rendezvous in the presence of jitter. Next, we extend the proposed algorithm to the non-role-based scenario. Simulation results show that it has the best performance with respect to existing works in various traffic patterns of primary users. Besides, it is the most predictable solution due to its low variance of TTR.

1. Introduction

Traditional spectrum management faces with scarcity in spectral resources due to inefficient utilization of operating frequencies. As a solution, DARPA introduced NeXt Generation networks in which there are some cognitive radios (CRs) who seek "spectral holes" in time and space domain [1]. This communication technique is commonly called Opportunistic Spectrum Access (OSA). In OSA technique, a hierarchical access structure with primary and secondary users is employed. Secondary users (SUs), equipped with CRs, can opportunistically utilize the spectrum when it is not used by primary users (PUs). Since there might be multiple SUs in a CR network, MAC protocol design has significant implications on the performance of CR networks. There are many multichannel MAC protocols which have been proposed for OSA approach in the literature [2,3]. Most of these works use channel hopping (CH) schemes [3]. The main issue in OSA MAC design through CH schemes is that two SUs should be synchronized [2]. That means they should be on the same channel at the same time to exchange their information. The synchronization can be divided into two phases [2]: the initial handshaking between the transmitter and the receiver, and then the synchronous hopping across the channels to exchange data.

In the initial handshaking phase, two SUs should meet each other for the first time in a common channel to exchange control messages and establish a communication link. This process is referred to as "rendezvous" [4]. Most of existing works assumed a common control channel (CCC) to facilitate the rendezvous. Although the dedicated CCC simplifies the rendezvous process, it may become a bottleneck for a CR network because of PUs' activities or jamming attacks [5]. Therefore, we focus on distributed approaches which are preferable in practical scenarios and called "blind rendezvous" in the literature [4]. The common approach in blind rendezvous is based on Channel Hopping (CH) technique [6,7]. In this technique, two SUs hop across PUs' channels until they find each other in a common channel.

One of the important research directions in blind rendezvous is to design algorithms that work properly in difficult situations. The blind rendezvous algorithms are usually evaluated by the following criteria:

- Asynchronous condition: As we mentioned before, in CH technique, two SUs try different channels in each time slot and jump across them according to CH sequences until they meet

* Corresponding author. Tel.: +98 2166165980.
E-mail addresses: saber.salehk@gmail.com, saber_saleh@ee.sharif.edu (S. Salehkaleybar), pakravan@sharif.edu (M.R. Pakravan).

http://dx.doi.org/10.1016/j.comcom.2016.01.004

Table 1
CH-based blind rendezvous algorithms.

Algorithms	MTTR	ETTR
JS [11]	$O(P^3)$	$O(P^3)$
CRSEQ [10]	$O(P^2)$	$O(P^2)$
S-ACH [14]	$O(\log(n)C^2)$	$O(C^2)$
E-AHW [9]	$O(CP)$	$O(CP)$
EJS [12]	$O(P^2)$	$O(P^2)$
mPJR	$O(C^2)$	$O(C^2)$
SARCH [16]	$O(P^2)$	–
MMC [4]	–	–

each other in a common available channel[1]. In real scenarios, there might be a timing offset between slot boundaries of CH sequences in two SUs. Therefore, it is desirable to design algorithms which work in the asynchronous condition.

- Asymmetric model: SUs attempt to rendezvous in the presence of PUs and they cannot meet each other if the visited channel is occupied by a PU in either side. In the asymmetric model, SUs may have different available channels because of their different location with respect to PUs in a wireless network and a blind rendezvous solution should facilitate the meeting of two SUs if they have at least one common available channel.
- Time-to-Rendezvous: Time-to-Rendezvous (TTR) shows the number of time slots that it takes to achieve rendezvous. The maximum TTR (MTTR) and expected TTR (ETTR) are two main benchmarks for comparing the performance of blind rendezvous algorithms. Another important criteria is variance of TTR [8], i.e. $\text{Var(TTR)} = \mathbb{E}\{(\text{TTR} - \text{ETTR})^2\}$, which indicates how samples of TTRs are close to ETTR.
- Heterogeneity of roles: In some blind rendezvous solutions, it is necessary that two SUs play different roles as a transmitter and receiver to guarantee rendezvous. The benefit of role-based solution is that they have less MTTR and ETTR than non-role-based solutions. However, it may be difficult to assign roles to SUs in some scenarios. For instance, SU A may search for SU B which it also tries to meet a third SU C. Then, SU B should be a transmitter and receiver at the same time.

1.1. Previous works

In the literature, different methods have been proposed in designing blind rendezvous algorithms. A comprehensive summary of previous works is given in [9]. Despite efforts, there are a few works which satisfy all criteria mentioned earlier. Hence, we consider algorithms which fulfill all requirements and they may have comparable performance in terms of MTTR and ETTR with respect to our proposed algorithm. The list of these algorithms are given in Table 1.

In [10], the CRSEQ algorithm has been proposed in which SUs sweep through all channels according to a CH sequence. The MTTR of this algorithm is $P(3P-1)$ time slots where P is the smallest prime number greater than or equal to the total number of channels, C. Lin et al. [11] suggested the JS algorithm which also supports multi-hop scenarios for multiuser rendezvous. The MTTR for this algorithm is equal to $6CP(P-G)$ time slots, where G is the number of common available channels between two SUs. Later, authors in [12] improved the JS algorithm to the enhanced JS (EJS) which reduces the MTTR by a factor of P. In [13], the ACH algorithm has been proposed which is a role-based solution. In this algorithm, the secondary transmitter and receiver try to rendezvous according to two different sequences. The MTTR for this algorithm is C^2. The extended version of it (S-ACH) for non-role-based case has been proposed in [14]. In this solution, it is assumed that each SU has a unique identifier (UID) and constructs its own CH sequence based on its UID. The MTTR of S-ACH algorithm is $\log(n)C^2$ where n is the number of SUs in the network. In [4], Theis et. al suggested the MC and MMC algorithms based on the number theory. In these algorithms, CH sequences are constructed by choosing a proper prime number and a rate that is selected randomly from numbers less than the chosen prime number. It can be shown that the MC algorithm does not guarantee rendezvous when two SUs select the same rate [11]. For the MMC algorithm, a similar problem arises when the chosen prime number of two SUs are the same [11]. Nevertheless, simulation results show that the MMC algorithm has a comparable ETTR to other algorithms in Table 1 and we consider it in our list.

Recently, the E-AHW algorithm has been proposed in which each SU generates a unique CH sequence based on its UID [9]. The CH sequences is composed of two hopping and waiting modes. It can be shown that the MTTR of this algorithm is $O(CP)$. Another approach in constructing CH sequences is based on ring walk method in which CH sequences are constructed from a base sequence and its circular shifts [15,16]. As we will see later, our proposed algorithm is a special case of ring walk-based sequences which guarantees periodic attempts to rendezvous in all available channels.

1.2. Our contributions

In this paper, we first develop a blind rendezvous algorithm for role-based scenario through construction of two CH sequences: Seq_{tx} and Seq_{rx}. In particular, transmitters and receivers hop across the channels according to Seq_{tx} and Seq_{rx}, respectively. All SUs which have no data to send, stay in the receiver mode and jump through channels according to Seq_{rx}. Whenever a SU wants talk to another node, it switches to transmitter mode and jumps based on Seq_{tx}. We show that the MTTR for the proposed algorithm is C^2 time slots if C is odd and it is equal to $(C+1)^2$ if C is even. In addition, the secondary transmitter and receiver choose the same channel to rendezvous once in each successive period of C time slots. Due to this property of constructed sequences, we call our proposed algorithm "Periodic Jump Rendezvous" (PJR). We show that this property of the constructed sequences improves the performance of the PJR algorithm by a linear factor reduction in ETTR compared to existing work [13] which may become a large value when the number of channels increases. We also study the case that there is a mismatch between the duration of a time slot in two SUs. This may frequently occur in practical implementations, due to presence of jitter[2] in SUs' clocks used for measuring time intervals. Hence, it is crucial to investigate the impact of this undesirable effect on blind rendezvous algorithms. We devise a solution for the PJR algorithm to ensure rendezvous in at most $(C+1)^2$ time slots in the presence of jitter.

Next, we extend the PJR algorithm to non-role-based case by a randomized solution. The modified version of the PJR algorithm is denoted by modified PJR (mPJR) algorithm. We prove that the MTTR of the mPJR algorithm is $O(C^2)$ similar to the PJR algorithm[3]. In addition, simulation results show that the mPJR algorithm has the lowest ETTR with respect to previous works in various PUs' traffic patterns. Furthermore, it is also the most predictable solution due to its very low Var(TTR) compared to other algorithms.

[1] In our context, a time slot is defined as a time interval that each SU stays in each channel before next jump.

[2] Here, we define the jitter as the difference between a measured clock period and the ideal period.

[3] In Section 2, we will show that the probability of not achieving rendezvous in $O(C^2)$ can be made arbitrarily small.

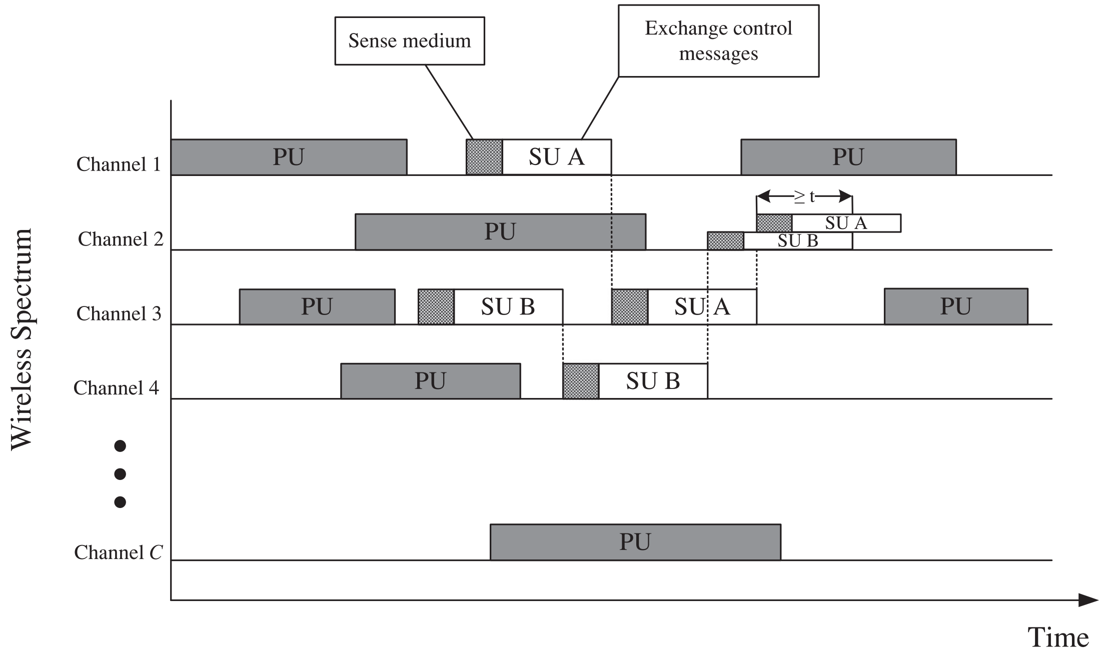


Fig. 1. Rendezvous process: SU A and SU B stay in each channel for a fixed length time slot. Each SU senses the current channel for the presence of PU. If the channel is idle, it will try to exchange control messages. In this example, two SUs meet each other on channel 2.

This paper is organized as follows: In Section 2, we first introduce the PJR algorithm and study some of its properties. Then, we analyze its performance and give a solution in the presence of jitter. In Section 3, we propose the mPJR algorithm for non-role-based scenario. In Section 4, we provide the simulation results for the performance of the PJR and mPJR algorithms and compare them with other related works. Finally, the conclusions are presented.

2. The periodic jump-based rendezvous algorithm

In this section, we first construct two sequences Seq'_{tx} and Seq'_{rx} with some specific features. Then, we explain how to generate Seq_{tx} and Seq_{rx} using these two sequences. We show that the MTTR for the PJR algorithm is C^2 time slots if C is odd and it is equal to $(C+1)^2$ if C is even. Afterwards, we obtain a tight upper bound for the ETTR of the PJR algorithm and compare it to a recent role-based solution, the ACH algorithm. At the end, we propose a solution to tolerate jitter of clocks at SUs.

Fig. 1 illustrates how the rendezvous process is executed similar to previous works in [4,11]. Each SU stays in a channel for a fixed length time slot and it will only change channel between time slots. SUs sense the medium for the presence of PU at the beginning of a time slot. If the channel is idle, it will try to exchange control message. A pair of SUs can establish a link successfully if they stay in a common channel for at least a time t. Note that t is the minimum time required to establish the link. After initial handshaking, two SUs decides how and when to exchange data.

2.1. Constructing channel hopping sequences Seq'_{tx} and Seq'_{rx}

Our goal in this part is to construct two CH sequences Seq'_{tx} and Seq'_{rx} with the largest achievable correlation between their circular shifts. Theorem 1 and Theorem 2 as follows, give a solution for constructing such sequences. First, we need some definitions.

Definition 1. Sequences Seq'_{tx} and Seq'_{rx} with length C are permutations of set $\{1, \ldots, C\}$. The value j $(1 \le j \le C)$ in the CH sequences represents jth channel. We define the correlation between two sequences Seq'_{tx} and Seq'_{rx} as follows:

Seq_{tx}	C	C-1	...	2	1
Seq_{rx}	1	2	...	C-1	C

Fig. 2. Two sequences with the correlation of one for any number of circular shifts.

$$Corr(Seq'_{tx}, Seq'_{rx}) = \sum_{i=1}^{C} I(Seq'_{tx}(i) = Seq'_{rx}(i)), \tag{1}$$

where $I(A = B)$ stands for indicator function[4], and $Seq'_{tx}(i)$ and $Seq'_{rx}(i)$ denote the integer value of ith entry in sequence Seq'_{tx} and Seq'_{rx}, respectively. For instance, in Fig. 2, we have $Seq'_{tx}(2) = C - 1$ and $Seq'_{rx}(2) = 2$.

Definition 2. A k-time circular shift of Seq'_{tx} (with length C) to the right, $Circ(Seq'_{tx}, k)$, is defined as follows:

$$Circ(Seq'_{tx}, k)(i) = Seq'_{tx}((i-k)_{\bmod C}), 1 \le i \le C, \tag{2}$$

where $()_{\bmod C}$ returns the remainder of argument to the integer value C. We also define $Seq'_{tx}(0) \triangleq Seq'_{tx}(C)$.[5] $Circ(Seq'_{rx}, k)$ is defined in the same way.

Theorem 1. *Consider any two sequences Seq'_{tx} and Seq'_{rx}. If any k-time $(1 \le k \le C)$ circular shift of Seq'_{tx} has at least one common value with Seq'_{rx}:*

$$Corr(Circ(Seq'_{tx}, k), Seq'_{rx}) \ge 1 \ , \forall k \in \{1, \ldots, C\}, \tag{3}$$

then,

$$Corr(Circ(Seq'_{tx}, k), Seq'_{rx}) = 1, \ \forall k \in \{1, \ldots, C\}. \tag{4}$$

[4] $I(A = B)$ is equal to one if $A = B$, otherwise it is equal to zero.
[5] Since we index the entries from 1 to C, we hold this assumption to coincide with our labeling.

Proof. Since each value j ($1 \le j \le C$) comes once in each sequence, this value contributes in correlation for only one of the C circular shifts. Therefore, for the sum of all C correlations between two sequences, we have:

$$\sum_{k=1}^{C} Corr(Circ(Seq'_{tx}, k), Seq'_{rx}) = C. \tag{5}$$

From (3) and (5), we can derive (4) and the proof is complete. □

According to Theorem 1, we should seek two sequences Seq'_{tx} and Seq'_{rx} with $Corr(Circ(Seq'_{tx}, k), Seq'_{rx}) = 1$ for any k-time ($1 \le k \le C$) circular shift. Existence of such sequences for odd values of C is ensured through the next theorem.

Theorem 2. *If C is odd, then there exist two sequences which satisfy Eq. (4). When C is even, such two sequences do not exist.*

Proof. First, we prove that there exist two sequences with the property stated in Eq. (4) (when C is odd) by giving an example (see Fig. 2). The integer value k occurs in the kth entry of Seq'_{rx} and $(C-k+1)$th entry of Seq'_{tx}. Therefore, the integer value k in two sequences meet each other after $(2k-1)_{\bmod C}$-time circular shift of Seq'_{tx} to the right. In addition, the required circular shifts are not equal for any different values of k and k'. To prove this, suppose that they are equal, then we have: $2k-1 \equiv 2k'-1 \mod C$. Therefore, $2(k-k') \equiv 0 \mod C$. Since C is odd, $C|k-k'$ (since C is odd and two is not factor of C. Hence, C should be a divisor of $k-k'$)[6]. Without loss of generality, assume that $k > k'$. We know that k and k' are lower than C. Hence, $k-k'$ is not divisible by C. Therefore, there are C different values for $(2k-1)_{\bmod C}$ ($1 \le k \le C$). In conclusion, the proposed sequences in Fig. 2 have $Corr(Circ(Seq'_{tx}, k), Seq'_{rx}) = 1$ for any number of circular shifts.

Now, we show that there are not such two sequences when C is even. Consider two sequences Seq'_{tx} and Seq'_{rx} with length C (C is even). We denote the entry that equals to i ($1 \le i \le C$) in Seq'_{tx} by x_i and in Seq'_{rx} by y_i. Hence, $(y_i - x_i)_{\bmod C}$-time circular shift to the right is needed to match the value i in Seq'_{tx} with the one in Seq'_{rx}. If Seq'_{tx} and Seq'_{rx} satisfy (3), $(y_i - x_i)_{\bmod C}$ ($1 \le i \le C$) should be different integers from the set $\{0, \ldots, C-1\}$. Therefore, we have:

$$\sum_{i=1}^{C} x_i - \sum_{i=1}^{C} y_i \equiv \sum_{i=1}^{C} k_i \mod C, \tag{6}$$

$$\equiv \frac{(C-1)C}{2} \mod C, \tag{7}$$

where $(y_i - x_i)_{\bmod C} = k_i$. In addition, we have: $\sum_{i=1}^{C} x_i = \sum_{i=1}^{C} y_i$. Therefore, $C|\frac{C(C-1)}{2}$. Since C and $C-1$ are relatively prime to each other, we have: $C|\frac{C}{2}$ which is a contradiction. Thus, there are not such two sequences when C is even. □

Remark 1. We have for any circular shift of Seq'_{rx} in Fig. 2, i.e. $Circ(Seq'_{rx}, k)$, $0 \le k \le C-1$:

$$Corr(Circ(Seq'_{tx}, k'), Circ(Seq'_{rx}, k)) = 1, \quad \forall k' \in \{0, \ldots, C-1\}. \tag{8}$$

when C is odd. Moreover, each channel contributes exactly one time in one of the above correlations.

Proof. One could easily follow the same method in Theorem 2 to prove the Remark 1. □

[6] For any two integer numbers a and b, we define the relation $a|b$ if there is an integer number k such that $b = a \times k$.

2.2. Constructing sequences Seq_{tx} and Seq_{rx}

From Theorem 2, the number of the channels must be odd in order to construct Seq'_{tx} and Seq'_{rx}. If the number of available channels is even, we add a virtual channel. This virtual channel can be any of the C channels (for instance, the first channel) and it is known by all SUs. Therefore, it is assumed that C is odd in the remainder of this section. We construct Seq_{tx} and Seq_{rx} with length C^2 using this procedure: to generate Seq_{tx}, we use the $Circ(Seq'_{tx}, k)$ ($0 \le k \le C-1$) for entries $kC+1$ to $(k+1)C$, where Seq'_{tx} is the sequence shown in Fig. 2. Besides, we copy Seq'_{rx} in Fig. 2, C times to generate Seq_{rx}. The transmitter and the receiver hop across the channels according to Seq_{tx} and Seq_{rx}, respectively. If any of them reaches at the end of its sequence, it starts hopping from the beginning.

Theorem 3. *Assume that the available channel sets of transmitter and receiver are S_1 and S_2, respectively. Two SUs performing the PJR algorithm achieve rendezvous at most in C^2 time slots if $S_1 \cap S_2 \ne \emptyset$.*

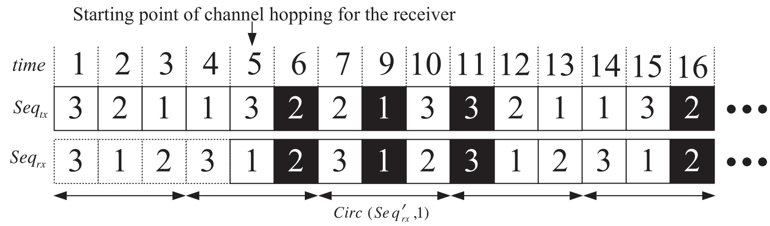
Proof. First, we assume two SUs are slot-synchronized. We remove this assumption later. Two cases can be considered here. In the first case, the receiver begins hopping i time slots (without loss of generality, we assume that $i < C^2$) later than the transmitter. Therefore, it seems that Seq_{rx} is shifted circularly $(i)_{\bmod C}$ times to the right. From another point of view, we can imagine that the new CH sequence for the receiver is constructed from $Circ(Seq'_{rx}, (i)_{\bmod C})$ and the receiver starts hopping from $i+1$th entry of this sequence. An example of this case is depicted in Fig. 3(a). If C is 3 and the receiver starts hopping 4 time slots later than the transmitter, it seems that the receiver uses a sequence which is constructed by $Circ(Seq'_{rx}, 1)$ and it starts from 5th entry of this sequence. According to Remark 1, $Circ(Seq'_{rx}, (i)_{\bmod C})$ has the same property as Seq'_{rx} and there is a common channel in two sequences $Circ(Seq'_{rx}, (i)_{\bmod C})$ and $Circ(Seq'_{tx}, k)$ for any $0 \le k \le C-1$. Besides, this common channel is different for various values of k. Hence, two SUs achieve rendezvous at most in C^2 time slots after the receiver starts hopping (see Fig. 3(b)). In the case that the transmitter begins hopping i time slots later than the receiver, it seems that the Seq_{rx} is constructed by copies of $Circ(Seq'_{rx}, (C-i)_{\bmod C})$. According to Remark 1, this sequence has a common channel with any circular shift of Seq'_{tx}, each one in a different channel. Therefore, two SUs achieve rendezvous in C^2 time slots after the transmitter starts hopping if there is a common available channel between two SUs.

So far, we proved that the rendezvous can be achieved for any offset i between two sequences Seq_{tx} and Seq_{rx} at most in C^2 time slots. Now, we extend our solution to the asynchronous condition. Without slot synchronization, the slot boundaries for two SUs may not be aligned. Suppose that t is the minimum time required to exchange control messages between two SUs. If the time slot duration is doubled to $T_{slot} = 2t$ as suggested in [18], then it is guaranteed that the two SUs stay in a common channel for at least time t. In order to show how the rendezvous is achieved when t is doubled, suppose that the misalignment difference between the transmitter and the receiver is $\delta = kT_{slot} + \sigma$, where $0 \le \sigma < T_{slot}$ (see Fig. 4). Then, we have two cases:

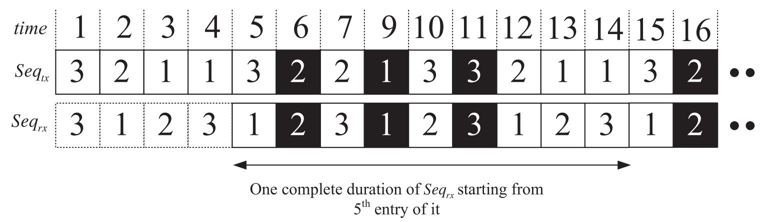
(1) $0 \le \sigma < t$: In this case, every slot from the beginning for the receiver overlaps for time duration $(T_{slot} - \sigma)$ with the corresponding one for the transmitter which is enough to exchange control messages.

(2) $t \le \sigma < T_{slot}$: In this case, every slot from the beginning for the receiver overlaps with the next slot for the transmitter for a time duration σ which is also enough to establish a link.

It should be noted that the same argument can be used if the receiver starts hopping first and the proof is complete. □



(a) The receiver starts hopping 4 slots later than the transmitter. It seems that the receiver uses a sequence which is constructed by $Circ(Seq'_{rx}, 1)$ and it starts from 5-th entry of this sequence. It should be noted that the first four entries of Seq_{rx} are shown in dash line. The reason is that the receiver starts hopping after these entries and they are just added to complete the first two sequences $Circ(Seq'_{rx}, 1)$ in Seq_{rx}.



(b) At time slot 14, the receiver completes the first round of hopping if it starts hopping from the 5-th entry of Seq_{rx}. Therefore, rendezvous will be achieved in $14 - 5 = 3^2$ time slots.

Fig. 3. Schematic of rendezvous process.

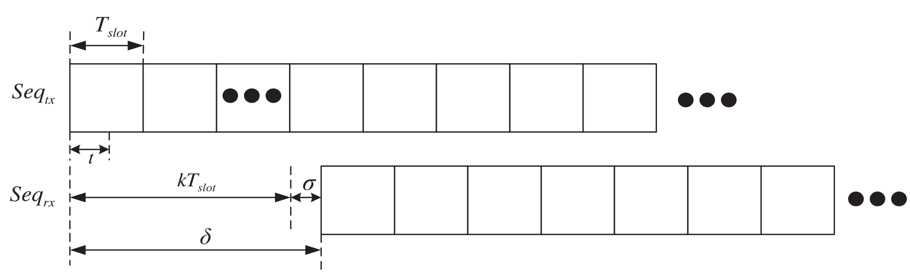


Fig. 4. Solution for the asynchronous condition by doubling the duration of the time slot.

In [13], it was shown that the minimum achievable MTTR is equal to C^2 if two SUs have a common available channel. Hence, the PJR algorithm can achieve this lower bound when C is odd and its MTTR is close to this bound when C is even. Next remark asserts a specific property of the constructed sequences which help SUs to achieve rendezvous in lower ETTR with respect to the related work, the ACH algorithm [13].

Remark 2. In PJR algorithm, the transmitter and the receiver choose the same channel for rendezvous once in each successive period of C time slots.

Proof. Without loss of generality, assume that the receiver starts hopping i time slots later than the transmitter. Therefore, Seq_{rx} is constructed by C copies of $Circ(Seq'_{rx}, (i) \bmod C)$. According to Remark 1, the correlation of the sequence $Circ(Seq'_{rx}, (i) \bmod C)$ with any sequence $Circ(Seq'_{tx}, k)$ $(0 \leq k \leq C-1)$ is equal to one. Hence, sequences Seq_{tx} and Seq_{rx} have a common value in each successive period of C time slots and the proof is complete. □

We explain in details what it means the successive period of C time slots by an example. Assume that the number of channels is three and the receiver starts hopping two time slots later than the transmitter (see Fig. 5). As it can be seen, in the block period [4, 6] at time slot 5, in the block period [7, 9] at time slot 7, and in the block period [10, 12] at time slot 12, the transmitter and the receiver meet each other and this pattern repeats in next periods. Therefore, the transmitter and the receiver have a chance to choose the same channel in each successive period of C time slots.

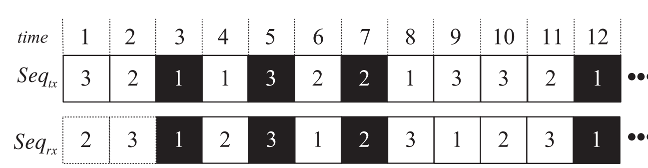
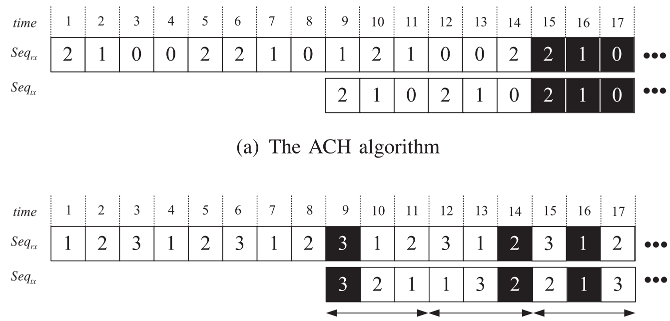


Fig. 5. In the PJR algorithm, sequences Seq_{tx} and Seq_{rx} have a common value in each successive period of C time slots. It should be noted that the two first entries of Seq_{rx} are shown in dash line. The reason is that the receiver starts hopping after these entries and they are just added to complete the first sequence $Circ(Seq'_{tx}, 2)$ in Seq_{rx}.

2.3. Average TTR of the PJR algorithm

As we stated before, the ACH and PJR algorithms have the minimum achievable MTTR. In this part, we compare two algorithms in terms of ETTR. Average TTR is one of the key performance metrics in designing a rendezvous algorithm for cognitive radio networks [11,13]. In fact, it is necessary to know how long it takes that two users meet each other on average. From an intuitive point of view, it can be said that the periodic jump property of the PJR algorithm may reduce the ETTR with respect to the ACH algorithm. To show



(a) The ACH algorithm

(b) The PJR algorithm

Fig. 6. Periodic jump feature of the PJR algorithm: the transmitter starts jumping 8 time slots later than the receiver. As it can be seen, for the sample sequences of the ACH algorithm, all attempts occur in the last three time slots in a block of nine time slots. However, in the PJR algorithm, the rendezvous attempts are distributed in time. This property yields a lower ETTR.

this, consider the scenario in Fig. 6. In fact, there are events that all attempts occur in the last C time slots of a block of C^2 time slots for the ACH algorithm. However, in the PJR algorithm, the rendezvous attempts are distributed in time. On average, the periodic jump property of the PJR gives lower ETTR for this algorithm.

To compare the ETTR of two algorithms analytically, we obtain an approximate ETTR for the ACH algorithm, $\mathbb{E}\{\text{TTR}_{ACH}\}$, and an upper bound for the ETTR of the PJR algorithm, $\mathbb{E}\{\text{TTR}_{PJR}\}$. We consider a simple model for channels occupation states to get closed-form formula for average TTRs. In this model, we assume that each channel is idle with probability p at the transmitter and the receiver side. Hence, they can exchange control messages in each channel with probability p^2 if they choose the same channel for rendezvous.

We first consider the ACH algorithm. In [13], it is argued that the CH pair sequences are constructed in a way that they have C distinct rendezvous channels in C^2 time slots. Then, they concluded that the ETTR for the ACH algorithm has an order of C time slots to rendezvous. Here, we show that the ETTR for this algorithm is approximately C/p^2. Considering the fact that each two CH sequences in the ACH algorithm have C distinct rendezvous channel in C^2 time slots, then the probability of rendezvous in each time slot is $1/C$. Therefore, the probability of a successful rendezvous in each time slot is $1/C \times p^2$ and we can approximately model the probability mass function (pmf) of TTR in the ACH algorithm as a geometric distribution with mean C/p^2. Thus, it can be concluded: $\mathbb{E}\{\text{TTR}_{ACH}\} \simeq C/p^2$. In the next section, we will see that simulation results verify the geometric distribution assumption for the pmf of TTR of the ACH algorithm.

Now, we derive an upper bound on the ETTR of the PJR algorithm. If the receiver starts hopping first, it can be shown by the property of the proposed sequences stated in Remark 2 that the probability of achieving rendezvous in kth time slot is approximately equal to: $\Pr\{\text{TTR} = k|\text{The recv. starts hopping first}\} = p^2(1-p^2)^i/C,$ for $Ci+1 \le k \le C(i+1)$ where $0 \le i \le C-1$. If the transmitter starts hopping first, then the probability of achieving rendezvous in kth time slot is approximately equal to the following equation:

$$\Pr\{\text{TTR} = k|\text{The trans. starts hopping first}\} = \begin{cases} \dfrac{p^2(1-p^2)^{2i}}{C} & \text{If } 2Ci < k \le 2Ci + \dfrac{C-1}{2}, \\ \dfrac{1}{2}\left(\dfrac{p^2(1-p^2)^{2i}}{C} + \dfrac{p^2(1-p^2)^{2i+1}}{C}\right) & \text{If } 2Ci + \dfrac{C-1}{2} < k \le 2Ci + \dfrac{3C+1}{2}, \\ \dfrac{p^2(1-p^2)^{2i+1}}{C} & \text{If } 2Ci + \dfrac{3C+1}{2} < k \le 2Ci + 2C. \end{cases} \quad (9)$$

where $0 \le i \le \frac{C-1}{2}$. The proof is given in Appendix A.

Now, we can calculate an upper bound for the ETTR of the PJR algorithm:

$$\begin{aligned} \mathbb{E}\{\text{TTR}_{PJR}\} \le \frac{1}{2}\sum_{i=0}^{\infty}\Bigg(&\left[\frac{1}{2}\left(\frac{C-1}{2}\right)\left(\frac{C+1}{2}\right) + (2iC)\frac{C}{2}\right] \times \frac{p^2(1-p^2)^{2i}}{C} + \\ &+\left[\left(2Ci + \frac{C+1}{2}\right)C + \frac{C(C-1)}{2}\right] \\ &\times\left(\frac{p^2(1-p^2)^{2i}}{2C} + \frac{p^2(1-p^2)^{2i+1}}{2C}\right) + \\ &+\left[\left(2Ci + \frac{3C+1}{2}\right)\frac{C}{2} + \frac{1}{2}\frac{(C-1)(C+1)}{2}\right] \\ &\times\frac{p^2(1-p^2)^{2i+1}}{C}\Bigg) \\ &+\frac{1}{2}\sum_{j=0}^{\infty}\frac{p^2(1-p^2)^j}{C}\left(jC^2 + \frac{C(C+1)}{2}\right). \end{aligned} \quad (10)$$

After some manipulations, we have:

$$\mathbb{E}\{\text{TTR}_{PJR}\} \le C\left(\frac{1}{p^2} - \frac{16-9p^2}{16(2-p^2)}\right). \quad (11)$$

Since the term $\frac{16-9p^2}{16(2-p^2)}$ is positive for any value of p, the upper bound of ETTR for the PJR algorithm is always less than the ETTR for the ACH algorithm. In addition, it can be seen that the gap between the PJR and ACH algorithm increases at least linearly with the number of channels. In the next part, we give a solution to ensure rendezvous in the presence of jitter.

2.4. Rendezvous in the presence of jitter

In most existing works, it is assumed that all SUs stay in each channel with the same time slot duration [4,6,10,13,17]. However, the duration of time slots in SUs may have small differences in practical networks due to inaccuracy of the SU clocks (clock jitter). Therefore, it is important to present a solution for this problem. In the next theorem, we present an idea by extending the duration of time slot.

Theorem 4. *Let ε be the time difference between the duration of one time slot in the secondary transmitter and receiver. Suppose that there is a common available channel at both side. Then, rendezvous occurs in C^2 time slots if the duration of one time slot, T_{slot}, is greater than the following term in both SUs:*

$$T_{slot} \ge 2(C^2-1)\varepsilon + 2t, \quad (12)$$

where t is the minimum time required to exchange control messages between two SUs.

Proof. Without loss of generality, assume that the duration of one time slot in the secondary transmitter and the receiver is T_{slot} and $T_{slot} + \varepsilon$, respectively. Furthermore, consider that the receiver starts hopping after the transmitter. The other case that the receiver starts hopping first can be analyzed in the same way. Now, assume that the time difference between the starting point of the receiver and the nearest time slot boundary of the transmitter is δ

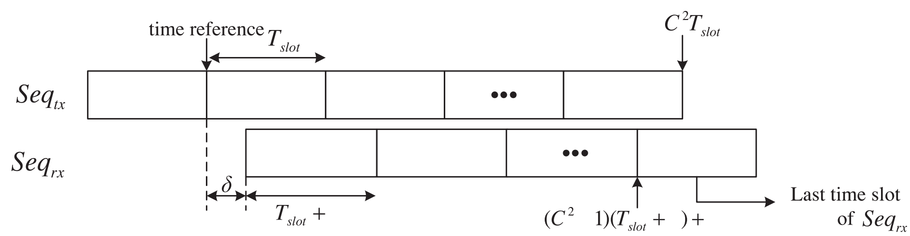


Fig. 7. Time diagram of two sequences Seq_{tx} and Seq_{rx} when the duration of one time slot in the transmitter and receiver is T_{slot} and $T_{slot}+\varepsilon$, respectively.

where $-\frac{T_{slot}}{2} \leq \delta \leq \frac{T_{slot}}{2}$ (see Fig. 7). In the worst case, assume that the rendezvous is achieved in the last time slot of Seq_{rx}. Hence, it is sufficient to guarantee that the transmitter and receiver have enough time to exchange control messages in this time slot. Since, by satisfying this condition, they have also enough time to meet each other in all previous time slots. Therefore, we can write the following inequalities to guarantee rendezvous according to Fig. 7:

$$\begin{cases} C^2T_{slot} - [(C^2-1)(T_{slot}+\varepsilon)+\delta] & \\ \quad \geq t \rightarrow T_{slot} \geq 2(C^2-1)\varepsilon + 2t & \text{for } 0 \leq \delta \leq \dfrac{T_{slot}}{2}, \\ C^2(T_{slot}+\varepsilon) - ((C^2-1)T_{slot}-\delta) & \\ \quad \geq t \rightarrow T_{slot} \geq 2t - 2C^2\varepsilon & \text{for } -\dfrac{T_{slot}}{2} \leq \delta < 0. \end{cases} \tag{13}$$

If the duration of a time slot in two SUs is greater than $2(C^2-1)\varepsilon+2t$, then they have enough time to exchange control messages. At last, we should add that if $\varepsilon \ll \frac{t}{C^2}$, the extension in the duration of one time slot is negligible and it is reasonable to employ this solution. □

3. The mPJR algorithm

In this section, we explain how the PJR algorithm can be extended to mPJR algorithm for non-role-based scenario. We will show that rendezvous is achieved after at most $2kC^2$ time slots with probability greater than $1-Cq^k$ where $0<q<1$. Hence, at least $2\log(C/\epsilon)/\log(q)C^2$ time slots are required to guarantee rendezvous with probability $1-\epsilon$. In the end, we give a simple analysis for the ETTR of mPJR algorithm.

3.1. Description of mPJR algorithm

Consider two SU A and SU B. The idea behind the mPJR algorithm is that each SU switches between two alternative modes I and II by using both Seq'_{tx} and Seq'_{rx} sequences. To do so, suppose each SU has a shift register of length C which represents the channel indices to be visited in upcoming C time slots. We denote this register by REG_i, $i \in \{A, B\}$. At the beginning, the REG_i is set to sequence Seq'_{tx}.

For each $2C$ subsequent time slots, SU i ($i \in$ {A, B}) chooses to be in one of two modes I and II with probability, $P_{i,I}$ and $1-P_{i,I}$, respectively. According to selected mode, SU i takes action in two different ways:

- Mode I: In this case, it uses REG_i for the first C time slots. Then, it shifts the REG_i circularly to the right by one and visits channels based on the updated REG for the next C time slots. After these $2C$ time slots, it shifts the REG_i to the right by one again and gets ready for the next $2C$ time slots.
- Mode II: In this mode, SU i sweeps through channels according to Seq'_{rx} twice. It also shifts the REG_i to the right by two.

The description of mPJR algorithm for SU i is given in Algorithm 1. The next theorem shows that if SUs follow the mPJR algorithm, they will have a rendezvous within $O(C^2)$ slots with high probability.

Algorithm 1 The mPJR algorithm.

```
Initialization: REG_i ← Seq'_tx
repeat
    Mode ← choose randomly from mode I or mode II with probability {P_{i,I}, 1 − P_{i,I}}.
    if Mode = mode I then
        visit channels according to REG_i for the first C time slots.
        REG_i ← Circ(REG_i, 1)
        visit channels according to REG_i for the next C time slots.
        REG_i ← Circ(REG_i, 1)
    else
        visit channels based on Seq'_rx twice for 2C time slots.
        REG_i ← Circ(REG_i, 2)
    end if
until Rendezvous is achieved.
```

Theorem 5. *In mPJR algorithm, the rendezvous is achieved after at most $2kC^2$ time slots with probability greater than $1-Cq^k$ if there is a common available channel between two SUs ($q=\max\{P_{A,I}(1-P_{B,I}), P_{B,I}(1-P_{A,I})\}$).*

Proof. Suppose that SU A and SU B try to rendezvous by executing mPJR algorithm. Without loss of generality, assume that SU B starts to visit channels j ($0 \leq j < C$) time slots later than SU A (see Fig. 8). Consider blocks $[2kC^2+2Cj+1, 2kC^2+2C(j+1)]$ of SU A where $k=0,1,2,\ldots,$ and $j \in \{0,\ldots,C-1\}$. We denote the set of these blocks by B_j^{SUA}. If SU A is in mode I, then it uses the same sequence in all blocks in the set B_j^{SUA}. Now, consider blocks $[2kC^2+2Cj+1, 2kC^2+2C(j+1)]$ of SU B where $k=0,1,2,\ldots,$ and $j \in \{0,\ldots,C-1\}$ (see Fig. 8). We represent the set of these blocks by B_j^{SUB}. If SU B is in mode II in one of the blocks in the set B_j^{SUB} and SU A chooses to be in mode I in one of blocks in the set B_j^{SUA}, the rendezvous is achieved if the common channel is available at both side. The mentioned event occurs in the first k blocks in the set B_i^{SUB} with probability $1-(q')^k$ where $q'=1-P_{A,I}(1-P_{B,I})$. Thus, the probability of SU A being in mode I and SU B in mode II at least once in the first k blocks of each set B_i^{SUB}, $0 \leq i < C$, is greater than $1-C(q')^k$. With the same arguments, it can be shown that the probability of SU A being in mode II and SU B in mode I at least once in the first k block of each set B_i^{SUB}, $0 \leq i < C$, is greater than $1-C(q'')^k$ where $q''=1-P_{B,I}(1-P_{A,I})$. Consequently, the rendezvous is achieved in $2kC^2$ time slots with probability greater than $1-Cq^k$. □

3.2. Performance analysis

Consider two SU A and SU B. There are three possible situations for initial handshaking between two SUs:

- SU A → SU B: SU A tries to establish a connection with SU B.
- SU B → SU A: SU B tries to get in contact with SU A.
- SU A ⇆ SU B: Both SUs attempt to rendezvous.

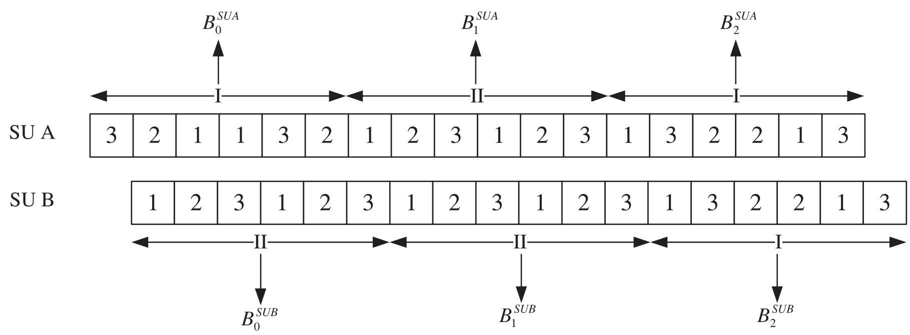


Fig. 8. SU B starts the mPJR algorithm one time slot later than SU A. In this instance of mPJR algorithm, SU A and SU B choose modes I, II, I and modes II, II, I for the first three of their blocks, respectively. For each block, the set which it belongs to, is also depicted.

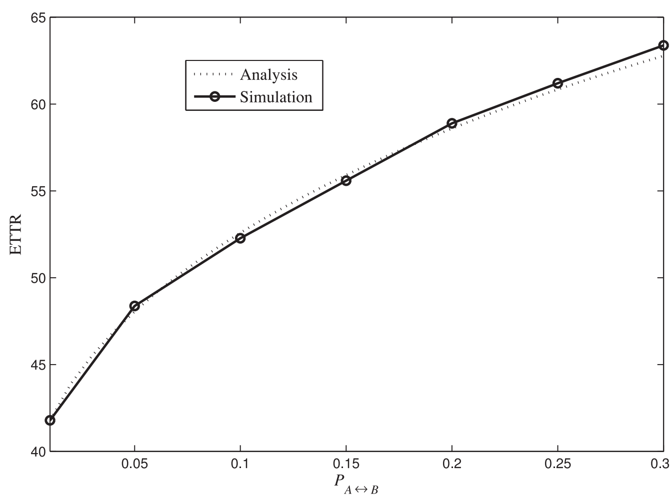


Fig. 9. ETTR of mPJR algorithm versus $P_{A\leftrightarrows B}$ in asymmetric model, $C = 11$.

Suppose that the third situation occurs with probability $P_{A\leftrightarrows B}$. We assume that if each SU i does not initiate to rendezvous, it just stays in mode II, i.e. $P_{i,I} = 0$, $i \in \{\text{A, B}\}$. Otherwise, it runs the mPJR algorithm by choosing modes I and II with probabilities $\{P_I, 1 - P_I\}$. In order to simplify the analysis, we assume that ETTR in each situation, is inversely related to the probability that one of SUs is in mode I and the other in mode II: ETTR$\propto 1/(P_{I,A}(1 - P_{I,B}) + P_{I,B}(1 - P_{I,A}))$.[7] Thus, the ETTR can be given as follows:

$$\text{ETTR} \approx K\left(P_{A\leftrightarrows B} \times \frac{1}{2P_I(1 - P_I)} + (1 - P_{A\leftrightarrows B}) \times \frac{1}{P_I}\right) \tag{14}$$

where K is a scale factor. Furthermore, the optimal $P_I^\star$ solution which minimizes the above equation can be obtained as follows:

$$P_I^\star = \frac{(P_{A\leftrightarrows B} - 2) + \sqrt{(2 - P_{A\leftrightarrows B})P_{A\leftrightarrows B}}}{2(P_{A\leftrightarrows B} - 1)} \tag{15}$$

In Fig. 9, the ETTR of mPJR algorithm for $P_{A\leftrightarrows B}$ in the range [0.01, 0.3], is depicted. As it can be seen, the mPJR algorithm inherits the low ETTR from role-based solutions. In simulation section, we will show that it has the lowest ETTR in different PU's traffic patterns.

In order to obtain an upper bound on $P_{A\leftrightarrows B}$, consider a CR network with $n \geq 10$ number of SUs co-located in the same area. Suppose that incoming traffic of each SU is ON-OFF process [19]. Thus, new session is started after a time interval according to Poisson process with rate λ. We assume that the mean inter-event time is greater than 10s, i.e. $\lambda \leq 0.1$, and the receiver of each session is randomly chosen from all SUs in the network. From simulation, it can be seen that $\mathbb{E}\{TTR\} \leq 5C$ in moderate PU's traffic. If we consider that the duration of a time slot (T_{slot}) is equal to $T_{slot} = 625\ \mu$s like in the rendezvous layer protocol of Bluetooth technology [20], then we have: $P_{A\leftrightarrows B} = 1 - e^{-\frac{\lambda \mathbb{E}\{TTR\} T_{slot}}{n-1}} \leq 0.01$. In simulation section, we set $P_{A\leftrightarrows B}$ to 0.01.

4. Simulations

In this section, we evaluate the performance of PJR and mPJR algorithms in comparison with other related works. Results are averaged over 10,000 runs. First, we compare ETTR of PJR algorithm with the existing role-based solution, i.e. the ACH algorithm, in the simple model for PUs' activities described in Section 2. Let us look at the pmf of TTR for two algorithms in an example. It can be seen from Fig. 10 that the simulation results are close to the pmfs from analysis. The lower ETTR for the PJR algorithm can be inferred from this figure. The reason is that the tail of distribution for the PJR algorithm is under the one of the ACH algorithm. Consequently, it is less probable for the PJR algorithm to achieve rendezvous in large TTRs. Furthermore, the main reason of stair-shaped form of the pmf for the PJR algorithm is that two SUs have a chance to rendezvous once in each successive period of C time slots according to Remark 2. Therefore, if in a C-time slot period, two SUs cannot meet because the mutually coinciding channel chosen by the two parties is occupied by PU, then the next opportunity will not be within this C-time slot period. The next chance will surely be in the next C-time slot period.

In order to quantify how the pmfs obtained by simulation and analysis are close to each other, we compute the Mean Square Error (MSE) as a goodness of fit between the simulation results and analysis. The results are given in Table 2 for different number of channels and channels' idle probabilities. It can be observed that the pmf given for the PJR algorithm is a valid approximation and it is close to the pmf obtained by simulation. In addition, the two pmfs become close as the number of channels increases.

Now, we compare the performance of two algorithms for different values of p in Fig. 11. The figure shows that the PJR algorithm has a lower ETTR with respect to the ACH algorithm for

Table 2
The MSE of pmfs of TTR between simulation and analysis.

$p\backslash C$	11	15	19	23
0.5	6.61×10^{-8}	3.18×10^{-9}	1.74×10^{-10}	4.48×10^{-11}
0.6	1.12×10^{-7}	5.51×10^{-9}	2.15×10^{-9}	9.17×10^{-11}
0.7	3.13×10^{-7}	7.29×10^{-9}	4.28×10^{-9}	2.49×10^{-10}
0.8	7.81×10^{-7}	2.57×10^{-8}	8.49×10^{-9}	4.31×10^{-10}

[7] Let E_k^j be the event that one of SUs in mode I and the other in mode II for the first time in kth block of the set B_j^{SUB}, $j \in \{0, \ldots, C-1\}$. This event has a geometric distribution, i.e. $\Pr\{E_k^j\} = r(1-r)^{k-1}$ where $r = P_{I,A}(1 - P_{I,B}) + P_{I,B}(1 - P_{I,A})$ and its average is $1/r$. Hence, the ETTR scales approximately with the parameter $1/r$.

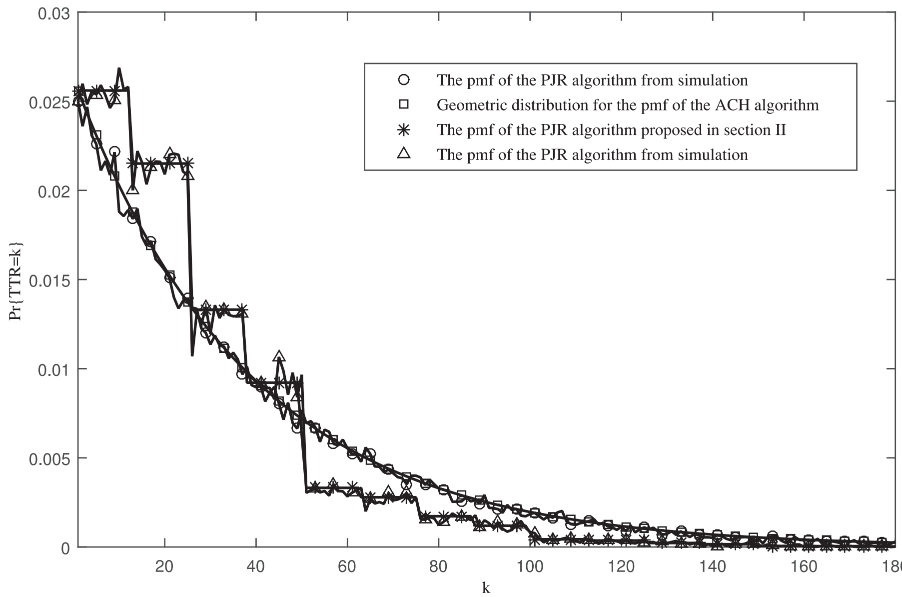


Fig. 10. Comparison of the pmf of the ETTR for PJR and ACH algorithms ($C = 25,\ p = 0.8$).

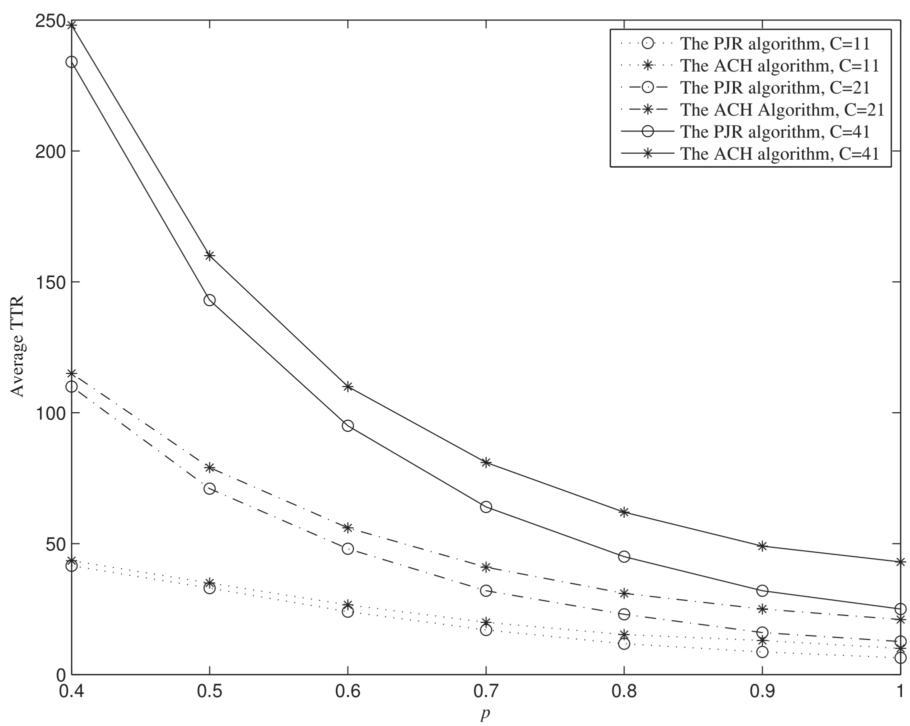


Fig. 11. Comparison of ETTR for PJR and ACH algorithms for different values of p.

different values of p. Besides, the difference between the average TTRs of two algorithms increases linearly with the number of channels. For instance, when $C = 21$ and $p = 0.7$, the gap is 10 and for $C = 41$ and $p = 0.7$, the gap is 30. This can be justified by our analysis of ETTR in previous section where we showed that $\mathbb{E}\{\mathrm{TTR}_{ACH}\} - \mathbb{E}\{\mathrm{TTR}_{PJR}\} \geq \frac{16-9p^2}{16(2-p^2)}C$.

In Fig. 12, the average TTR of the PJR algorithm is plotted versus number of channels for the jitter $\varepsilon = 0.05T_{slot}, 0.1T_{slot}$. We set the time slot $T_{slot} = 2t = 2$ where t is the minimum time to exchange control messages. It can be seen that average TTR increases just by 10% percent if we double value of jitter from $0.05T_{slot}$ to $0.1T_{slot}$.

Although the model considered for PUs' activities in Section 2 is useful to simplify the analysis, it may not remain valid in practical scenarios. Therefore, we consider a more realistic model based on Markov chains in the rest of simulations. Empirical studies show that PU traffics can be modeled approximately by Markov (or Semi-Markov) chains with two idle and busy states [21,22]. In Markov chain model, one can set average time duration of idle and busy periods by adjusting transition probabilities in the model. In Fig. 13, the relation between average time of idle and busy periods (T_{idle} and T_{busy}) of PU traffics and time slot (T_{slot}) of SUs, are depicted. We denote the sum of T_{idle} and T_{busy} by T. The parameters α and β are the ratio T_{idle}/T, and T_{slot}/T, respectively.

By increasing the value of α from zero to one, we can model heavy to light traffic load in PU connections. Besides, we can have different traffic patterns for running services at PUs by changing

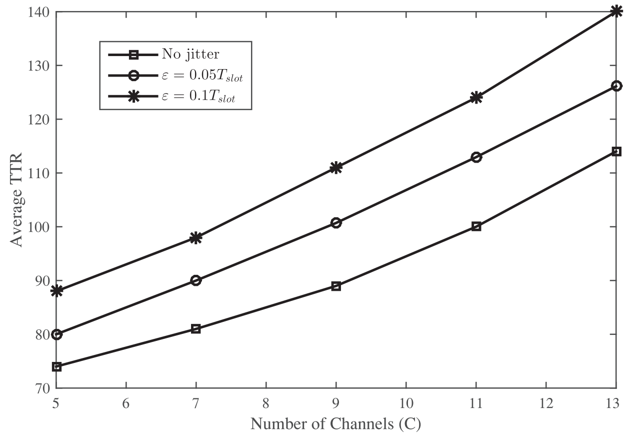


Fig. 12. Average TTR versus number of channels in the presence of jitter, $p = 0.5$.

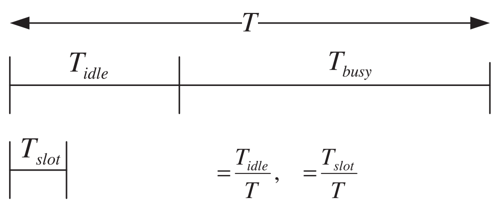


Fig. 13. Illustration of idle time, busy time, and time slot.

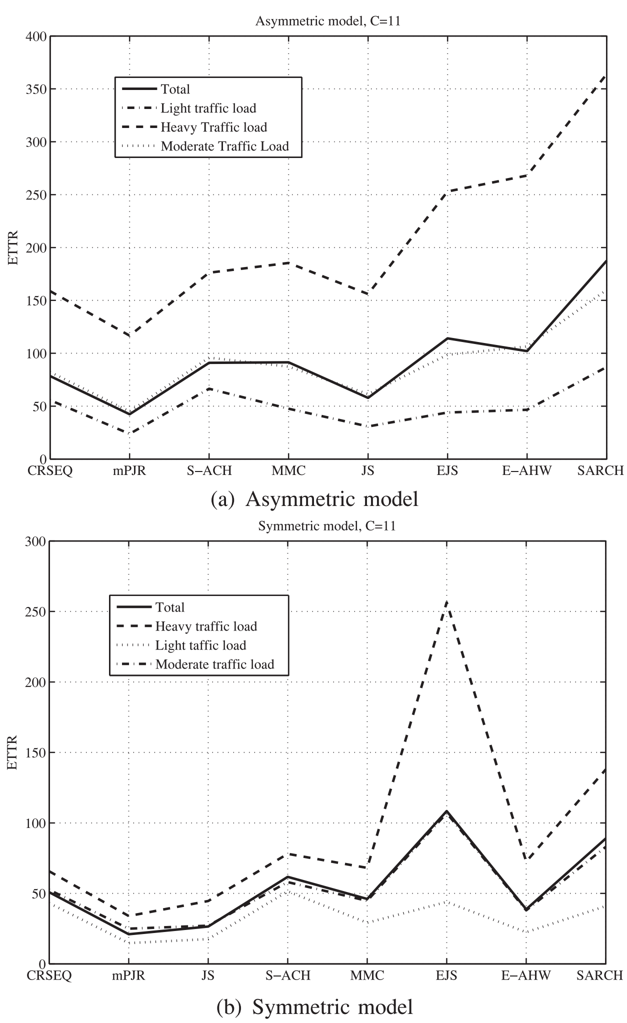


Fig. 14. Comparison of ETTR for non-role-based solutions.

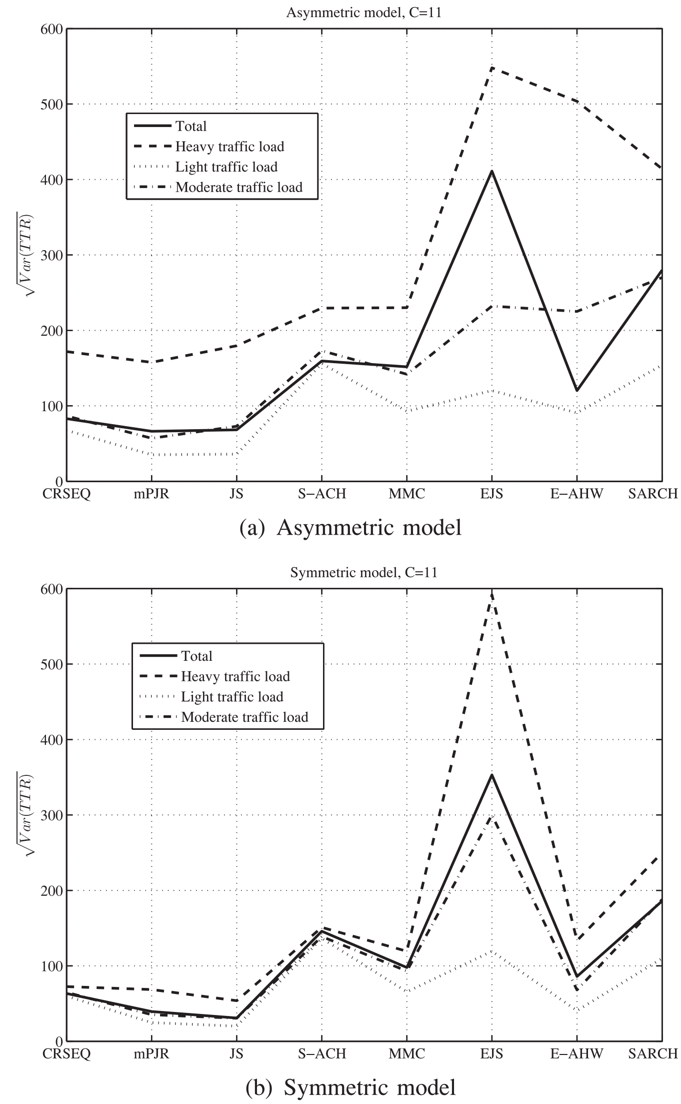


Fig. 15. Comparison of variance of TTR for non-role-based solutions.

the value of β. For instance, large values of β can be used for voice traffics which have relatively small idle and busy periods and small value of β can model some traffic patterns like web traffics. In each run of simulation, the value of β for each channel is randomly chosen from the set {1/20, 1/100, 1/200, 1/400, 1/600, 1/800, 1/1200, 1/1600}.

Now, we compare the performance of mPJR algorithm with other related works in Table 1. We run simulations in two symmetric and asymmetric model. In each model, we consider four cases: (1) light traffic load: $\alpha = 2/3$; (2) moderate traffic load: $\alpha = 1/2$; (3) Heavy traffic load: $\alpha = 1/3$; (4) Total: in each run, the value of α is randomly chosen from the set {1/3, 1/2, 2/3} for each channel.

Simulation results for four cases in asymmetric and symmetric models are given in Fig. 14. As it can be seen, the mPJR algorithm has the lowest ETTR between other algorithms for all cases in both models. More specifically, we have at least 20% and 25% improvement in symmetric and asymmetric models, respectively. In Fig. 15, variance of TTR of non-role-based algorithms are given. From this figure, it can be inferred that the mPJR algorithm is more predictable solution compared to other algorithms because of having the lowest variance under almost all the cases.

Table 3
The value of $\Pr\{\text{TTR}=k|A_j\}$.

$\Pr\{\text{TTR}=k|A_j\},\ 1\le k\le 12$													
$j\backslash$TTR	1	2	3	4	5	6	7	8	9	10	11	12	
1	1	1	0.4	0.7	0.7	0.2	0.2	0	0	0	0	0	
2	0	0	0.6	0.3	0.3	0.4	0.7	0.9	0.9	0.9	0	0	
3	0	0	0	0	0	0.4	0.1	0.1	0.1	0.1	1	1	
4	0	0	0	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	0	0	0	0	0	0	0	0	
$\Pr\{\text{TTR}=k|A_j\},\ 13\le k\le 25$													
$j\backslash$TTR	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	0.3	0.6	0.9	0.1	0.1	0	0	0	0	0	0	0
4	0	0.7	0.4	0.1	0.4	0.7	0.8	0.8	0.3	0	0	0	0
5	0	0	0	0	0.5	0.2	0.2	0.2	0.7	1	1	1	1

5. Conclusions

In this paper, we proposed the PJR algorithm for rendezvous in CR networks. We first considered a role-based solution and showed that it ensures rendezvous in at most $(C+1)^2$ time slots. Furthermore, the PJR algorithm has better performance than the other existing work, the ACH algorithm, in terms of ETTR due to the specific property of constructed sequences. We also studied the case in which there is a mismatch between the duration of time slots in two SUs and give a solution to guarantee rendezvous in this condition. Afterwards, we proposed the mPJR algorithm for non-role-based case. Simulation results showed that the proposed algorithm has the lowest ETTR in various PUs' traffic patterns with respect to other related works. Besides, its variance of TTR is the lowest among other algorithms under almost all cases which makes it a promising solution for rendezvous in CR networks.

Appendix A

In this appendix, we derive the proposed pmf of TTR_{PJR} which is stated in Section 2. We should mention that we do not follow the exact analysis. Actually, obtaining the exact pmf is intractable in analysis due to many possible events yielding the same TTR. Our aim here is to give reasonable justification for the proposed pmf of TTR. We start by obtaining an approximate pmf for the case that the receiver starts hopping first. Consider that the transmitter starts hopping d_0 time slots later than the receiver. Then, the channel k $(1\le k\le C)$, appears for the jth times in $(Cj-(k-j)_{\bmod C})$th entry of Seq_{rx} and $(Cj-(d_0-k)_{\bmod C})$th entry of Seq_{tx} due to structure of constructed sequences. By solving the equation $Cj-(k-j)_{\bmod C}=Cj'-(d_0-k)_{\bmod C}$ to find k and replacing it in $Cj-(k-j)_{\bmod C}$, we get the following equation:

$$A(j,d_0)=\begin{cases} Cj-\left(\dfrac{d_0-j}{2}\right)_{\bmod C} & \text{If } d_0-j \text{ is even,}\\ Cj-\left(\dfrac{C+d_0-j}{2}\right)_{\bmod C} & \text{Otherwise.}\end{cases} \tag{16}$$

where $A(j,\ d_0)$ denotes the time slot which jth attempt for rendezvous occurs and the transmitter starts hopping d_0 time slots later than the receiver. Since the range of the function $()_{\bmod C}$ is $[0,C-1]$, then the value of $A(j,\ d_0)$ is between $C(j-1)+1$ and Cj. Besides, we can get any possible integer value for $A(j,\ d_0)$ in the range $[C(j-1)+1,Cj]$ by changing the value of d_0. Therefore, if the rendezvous is achieved in time slot $C(j-1)+1\le \text{TTR}=k\le Cj$, it would be the jth attempt and the transmitter and the receiver have not exchanged control messages successfully in any previous attempts. Now, note that the probability of meeting in each time slot is $\frac{1}{C}$ on average and the probability of the rendezvous channel being idle in the transmitter and the receiver's side is p^2. Hence, the probability of successfully exchanging control message is $\frac{p^2}{C}$. In addition, the probability of unsuccessful rendezvous in $(j-1)$ number of previous attempts is $(1-p^2)^{(j-1)}$. Hence, $\Pr\{\text{TTR}=k|\text{The recv. starts hopping first}\}=\frac{p^2(1-p^2)^{j-1}}{C}$ for $C(j-1)+1\le k\le Cj$. For the case that the transmitter starts hopping first, the exact analysis cannot be followed similar to the previous case. Here, we should first compute the probability $\Pr\{\text{TTR}=k|$ the current meeting of the transmitter and the receiver is jth attempt to exchange control messages$\}$ which is intractable in analysis as the number of channels increases due to many possible events yielding the same TTR. An example of this probability, $\Pr\{\text{TTR}=k|A_j\}$, when $C=5$ is given in Table 3. Assume that the transmitter and receiver meet each other 6 time slots later than the receiver starts hopping (TTR = 6). From this table, we can state that there is a probability of 0.2 that this is their first rendezvous attempt and there is a probability of 0.4 that this is their second or third rendezvous attempt. After observing the value of this probability for different number of channels, we considered the following equation as a good approximation of it in terms of MSE:

$$\Pr\{\text{TTR}=k|A_j\}=\begin{cases} 1 & \text{If } C(j-1)<k\le C(j-1)+\dfrac{C-1}{2},\\ \dfrac{1}{2} & \text{If } C(j-1)+\dfrac{C-1}{2}<k\le C(j-1)+\dfrac{3C+1}{2},\\ \dfrac{1}{2} & \text{If } C(j-2)+\dfrac{C-1}{2}<k\le C(j-2)+\dfrac{3C+1}{2},\\ 1 & \text{If } C(j-2)+\dfrac{3C+1}{2}<k\le C(j-2)+2C.\end{cases} \tag{17}$$

Therefore, the probability $\Pr\{\text{TTR}=k|\text{the transmitter starts hopping first}\}$ is given by Eq. (9) and we have for $\Pr\{\text{TTR}=k\}$:

$$\Pr\{\text{TTR}{=}k\}=\frac{1}{2}(\Pr\{\text{TTR}=k|\text{the transmitter starts hopping first}\} + \Pr\{\text{TTR}=k|\text{the receiver starts hopping first}\}). \tag{18}$$

References

[1] I.F. Akyildiz, W. Lee, M.C. Vuran, S. Mohanty, Next generation dynamic spectrum access/cognitive radio wireless networks: A survey, Comput. Netw. 50 (13) (2006) 2127–2159.

[2] Q. Zhao, L. Tong, A. Swami, Y. Chen, Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: A POMDP framework, IEEE J. Select. Areas Commun. 25 (3) (2007) 589–600.

[3] P. Pawelczak, S. Pollin, H.S. So, A. Bahai, R.V. Prasad, R. Hekmat, Performance analysis of multichannel medium access control algorithms for opportunistic spectrum access, IEEE Trans. Veh. Technol. 58 (6) (2009) 3014–3031.

[4] N.C. Theis, R.W. Thomas, L.A. DaSilva, Rendezvous for cognitive radios, IEEE Trans. Mob. Comput. 10 (2) (2010) 216–227.

[5] B. Horine, D. Turgut, Link rendezvous protocol for cognitive radio networks, in: Proceedings of the Second IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2007, pp. 444–447.

[6] K. Bian, J.-M. Park, R. Chen, A quorum-based framework for establishing control channels in dynamic spectrum access networks, in: Proceedings of the Fifteenth Annual International Conference on Mobile Computing and Networking (MobiCom'09), September 2009.

[7] K. Bian, J.-M. Park, R. Chen, Control channel establishment in cognitive radio networks using channel hopping, IEEE J. Sel. Areas Commun. 29 (4) (2011) 689–703.

[8] Q. Liu, P. Deming, H. Gang, W. Xiaodong, Z. Xingming, A neighbor cooperation framework for time-efficient asynchronous channel hopping rendezvous in cognitive radio networks, in: Proceedings of the IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN), 2012, pp. 529–539.

[9] I. Chuang, H. Wu, Y. Kuo, A fast blind rendezvous method by alternate hop-and-wait channel hopping in cognitive radio networks, IEEE Trans. Mob. Comput. 13 (10) (2014) 2171–2184.

[10] J. Shin, D. Yang, C. Kim, A channel rendezvous scheme for cognitive radio networks, IEEE Comm. Lett. 14 (10) (2010) 954–956.

[11] Z. Lin, H. Liu, X. Chu, Y.W. Leung, Jump-stay based channel-hopping algorithm with guaranteed rendezvous for cognitive radio networks, in: Proceedings of the Thirtieth IEEE International Conference on Computer Communications (INFOCOM), Shanghai, China, 2011.

[12] Z. Lin, H. Liu, X. Chu, Y. Leung, Enhanced jump-stay rendezvous algorithm for cognitive radio networks, IEEE Communications Letters vol.17 (no.9) (2013) 1742–1745.

[13] K. Bian, J. Park, Asynchronous channel hopping for establishing rendezvous in cognitive radio networks, in: Proceedings of the Thirtieth IEEE International Conference on Computer Communications (INFOCOM), Shanghai, China, 2011.

[14] K. Bian, J. Park, Maximizing rendezvous diversity in rendezvous protocols for decentralized cognitive radio networks, IEEE Trans. Mob. Comput. 12 (7) (2013) 1294–1307.

[15] H. Liu, Z. Lin, X. Chu, Y.-W. Leung, Ring-walk based channel hopping algorithms with guaranteed rendezvous for cognitive radio networks, in: Proceedings of the 2010 IEEE/ACM International Conference on Green Computing and Communications, 2010, pp. 755–760.

[16] G.-Y. Chang, W.-H. Teng, H.-Y. Chen, J.-P. Sheu, Novel channel- hopping schemes for cognitive radio networks, IEEE Trans. Mob. Comput. 13 (2) (2014) 407–421.

[17] C. Cormio, K.R. Chowdhury, Common control channel design for cognitive radio wireless ad hoc networks using adaptive frequency hopping, Ad Hoc Netw. 8 (2010) 430–438.

[18] D.M. Yang, J.M. Shin, C.H. Kim, Deterministic rendezvous scheme in multichannel access networks, IET Electron. Lett. 46 (2010) 1402.

[19] A. Adas, Traffic models in broadband networks, IEEE Commun. Mag. 35 (7) (1997) 82–89.

[20] F. Siegemund, M. Rohs, Rendezvous layer protocols for Bluetooth-enabled smart devices, Trends in Network and Pervasive Computing ARCS 2002, Springer, Berlin Heidelberg, 2002, pp. 256–273.

[21] S. Geirhofer, L. Tong, B.M. Sadler, Dynamic spectrum access in the time domain: Modeling and exploiting white space, IEEE Commun. Mag. 45 (5) (2007) 66–72.

[22] A. Motamedi, A. Bahai, MAC protocol design for spectrum-agile wireless networks: Stochastic control approach, in: Proceedings of the Second IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), April 2007, pp. 448–451.