Ring-Walk Rendezvous Algorithms for Cognitive Radio Networks*

ZHIYONG LIN\textsuperscript{2,1}, HAI LIU\textsuperscript{1}, XIAOWEN CHU\textsuperscript{1} AND YIU-WING LEUNG\textsuperscript{1}

\textsuperscript{1}Dept of Computer Science, Hong Kong Baptist University, Hong Kong
\textsuperscript{2}Dept of Computer Science, GuangDong Polytechnic Normal University, GuangZhou, China
E-mail: zylin, hliu, chxw, ywleung@comp.hkbu.edu.hk

Received: June 2, 2010. Accepted: May 11, 2011.

Rendezvous in cognitive radio networks is the process that cognitive users meet and establish a communication link on a commonly available channel, so that consequent information exchange and data communications can be accomplished on the channel. In this study, we propose two ring-walk (RW) rendezvous algorithms to solve the problem of blind rendezvous, i.e., rendezvous without the help of any central controller and dedicated common control channel. Our basic idea is to represent each channel as a vertex in a ring. Cognitive users “walk” on the ring by visiting vertices of channels with different velocities. Rendezvous is achievable since the user with lower velocity will be eventually “caught” by the user with higher velocity. Compared with the existing solutions, our algorithms provide guaranteed rendezvous without the need of time-synchronization and are applicable to rendezvous of the multi-user and multi-hop scenario. We derive upper bounds on the time-to-rendezvous (TTR) and the expected TTR of our algorithms in both 2-user and multi-user scenarios. Extensive simulations are conducted to evaluate the performance of our algorithms.

Keywords: Cognitive radio, Rendezvous, Channel hopping.

1 INTRODUCTION

Due to the enormous deployments of wireless applications, the limited unlicensed spectrum resources (e.g., 2.4 GHz Industrial, Scientific and Medical (ISM) bands) have already become over-crowded. As opposed to this, a large

* A preliminary version of this work was published in [12].
portion of licensed spectrum lies unused at most of the time, which results in many “spectrum holes” or idle channels. To alleviate the severe scarcity in unlicensed spectrum as well as improve the efficiency of licensed spectrum usage, the Federal Communications Commission (FCC) in the US have begun to consider allowing unlicensed users (or secondary users, SUs) to access the spectrum of licensed users (or primary users, PUs) opportunistically [7]. Towards this end, the networking paradigm is driven to shift from the traditional static spectrum access scheme to dynamic spectrum access (DSA) scheme [1, 15]. Cognitive radios, devices which can sense the spectrum for idle channels and further access them by adaptively adjusting their transmission parameters such as modulation type and power, have been envisioned as the promising enabler for DSA [1]. Equipped with cognitive radios, a set of SUs (also known as cognitive users, CUs) can form a cognitive radio network (CRN) and coexist with PUs in the same geographical area by exploring and accessing the vacant spectrum of PUs opportunistically.

Users (unless specified otherwise, the users mentioned hereafter in this paper refer to CUs by default) in a CRN should detect the presence of each other to establish communication links, so that information exchange, spectrum management and data communication can be carried out. The process that two or more radios of users meet and establish a link on a common channel is referred to as “rendezvous” [14]. Rendezvous is a fundamental and essential operation in CRNs and data communication is impossible without rendezvous of users. However, implementation of rendezvous in a CRN is nontrivial, since the users in the network are even not aware of presence of each other before rendezvous and the available channels of each user are usually different and change dynamically.

Most of existing works on rendezvous either utilized centralized controller [2] or employed dedicated common control channel (CCC) [5, 8, 13] to facilitate the rendezvous. Although these strategies simplify the rendezvous considerably, they suffer from new problems which include the poor scalability and flexibility of centralized systems, the difficulty or even infeasibility in establishing a CCC in DSA, the vulnerability of CCC to jamming attack, and so on. Therefore, blind rendezvous systems without any centralized controller and dedicated CCC are preferable in practice [14].

Channel-hopping (CH) is one of the most representative techniques for blind rendezvous. With CH technique, each user of a CRN selects a set of available channels and hops among these channels for rendezvous with its potential neighbors. If all users have the same available channels, we call it symmetric model. We call it asymmetric model otherwise, i.e., different users might have different available channels. Several effective CH algorithms have been proposed in the recent literature [3, 4, 6, 14]. We observe that these algorithms could be further enhanced in following directions: i) providing guaranteed rendezvous without the need of time-synchronization; ii) applicable to asymmetric model; iii) applicable to rendezvous in multi-user multi-hop scenario.
In this study, we focus on the design of CH algorithms for blind rendezvous. We propose two ring-walk rendezvous algorithms, namely RW1 and RW2, which guarantee rendezvous of two or multiple users in time-slotted CRNs without the need of time-synchronization. The most important metric to evaluate CH algorithms is time-to-rendezvous (TTR), which is defined as the number of time slots that it takes for users to achieve rendezvous. Table 1 highlights the performance of our algorithms and other representative algorithms, where MTTR and E(TTR) denote the maximum TTR and the expected TTR, respectively.

Contributions of this work are summarized as follows:

i) New algorithms with guaranteed rendezvous: two ring-walk rendezvous algorithms are proposed to achieve guaranteed rendezvous without the need of time-synchronization under both symmetric and asymmetric models.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>2-user 1-hop</th>
<th>Multi-user multi-hop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Symmetric Model</td>
<td>Asymmetric Model</td>
</tr>
<tr>
<td>RW1</td>
<td>$(M-1)(N^2-N)$</td>
<td>$2(M+1-G)$</td>
</tr>
<tr>
<td></td>
<td>$(M-1)(N^2-N)$</td>
<td>$(2D+(N-K)(1+\ln D))$</td>
</tr>
<tr>
<td>RW2</td>
<td>$(M-1)N$</td>
<td>$2(M+1-G)$</td>
</tr>
<tr>
<td></td>
<td>$(M-1)(lnN+1/2)$</td>
<td>unknown</td>
</tr>
<tr>
<td>Generated</td>
<td>$M(M+1)$</td>
<td>x</td>
</tr>
<tr>
<td>Orthogonal Sequence [6]</td>
<td>$\frac{M^4+2M^2+6M-3}{3M^2+3M}$</td>
<td>x</td>
</tr>
<tr>
<td>Modular</td>
<td>$2P$ (not guaranteed)</td>
<td>x</td>
</tr>
<tr>
<td>Clock [14]</td>
<td>$2P^2/(P-1)$</td>
<td>x</td>
</tr>
<tr>
<td>Modified Modular Clock [14]</td>
<td>x</td>
<td>unknown</td>
</tr>
</tbody>
</table>

TABLE 1
MTTR and E(TTR) of Different Algorithms

Remarks: i) MTTR and E(TTR) are shown in unshadowed rows and shadowed rows, respectively. ii) “x” means that the algorithm cannot be directly applied; “unknown” means that the algorithm is applicable but its performance is unknown. iii) $M$ is the size of the whole potentially available channel set; $P$ is the smallest prime number that is greater than or equal to $M$; $G$ is the number of commonly available channels of users; $D$ is the network diameter in terms of number of hops; $N$ is the network size.
ii) *Generalized rendezvous for multi-user and multi-hop scenarios:* RW1 and RW2 are both applicable to rendezvous of multiple users within 1-hop and even multi-hop neighborhood.

iii) *Solid theoretical analysis:* we derive upper bounds on the MTTR and the expected TTR of new algorithms in 2-user and multi-user multi-hop scenarios.

The rest of this paper is organized as follows. Related work is reviewed in Section 2. The rendezvous problem is formally formulated in Section 3. We propose ring-walk rendezvous algorithms and present the theoretical analysis in Section 4. Numerical simulation is conducted in section 5. We conclude our work in Section 6.

### 2 RELATED WORK

The existing rendezvous systems for CRN can be classified into two categories based on their structures: 1) infrastructure-based or centralized systems and 2) infrastructure-less or decentralized systems.

In a centralized system [2, 5], there is a pre-assigned central controller, which acts like a “broker” to assist the whole rendezvous process of the network. Although employing a central controller can facilitate the rendezvous of users greatly, such centralized systems have many drawbacks, such as the poor scalability and the obvious vulnerability to jamming attack. Therefore, centralized rendezvous systems are not widely adopted in practice.

The decentralized systems without using centralized controllers can be further classified into two subcategories depending on whether employing common control channel (CCC) or not. The decentralized systems using CCC rely either on a global CCC [8, 13] or a local CCC [9, 10, 16] to facilitate the rendezvous. However, it is difficult or even impossible to achieve a global CCC in practice and the overhead to establish and maintain a local CCC is considerable. Thus, the decentralized rendezvous systems without using CCC, i.e., *blind rendezvous* systems, draw more attention of researchers.

In the literature, numbers of blind rendezvous systems have been proposed based on the technique of channel hopping (CH) [3, 4, 6, 14]. With CH technique, each user of a CRN selects a set of available channels and hops among these channels for rendezvous with its potential neighbors. The rendezvous is achieved if the users hop on a commonly-available channel in the same time slot. A trivial CH algorithm is (purely) random algorithm, in which each user, according to the available channels, generates its CH sequence in a completely random way. An improved random algorithm, called AMRCC, was proposed in [4]. The main idea of AMRCC is that the channels with lower interference to PUs have larger chances to be selected into the CH sequence.
However, the random algorithms including AMRCC cannot guarantee the rendezvous of users in finite time.

There are several CH algorithms which achieve guaranteed rendezvous under some specific conditions. Based on quorum systems, two CH algorithms, namely M-QCH and L-QCH, proposed in [3] can guarantee rendezvous between users in time synchronized systems. Another CH algorithm called A-QCH was proposed in [3] for the asynchronous systems. However, A-QCH is applicable to the systems with two channels, which limits applications of the algorithm. A CH algorithm with guaranteed rendezvous for asynchronous systems was proposed in [6] and was later referred to as the generated orthogonal sequence (GOS) algorithm in [14]. The problem with GOS is that it is applicable to the symmetric model only and imbalance of channel availabilities is ignored. Recently, a notable work by Theis et al. [14] presented two CH algorithms: modular clock algorithm (MC) and its modified version MMC. MC and MMC work for symmetric model and asymmetric model, respectively. The basic idea of MC and MMC is that each user picks a proper prime number and randomly selects a rate less than the prime number. Based on the two parameters, the user generates its CH sequence via pre-defined modulo operations. Although MC and MMC are shown to be effective, both algorithms cannot guarantee the rendezvous if the selected rates or the prime numbers of two users are identical.

In conclusion, we observe that the existing CH algorithms could be further enhanced in following directions: i) providing guaranteed rendezvous without the need of time-synchronization; ii) applicable to asymmetric model; iii) applicable to rendezvous in multi-user multi-hop scenario.

3 SYSTEM MODEL AND PROBLEM FORMULATION

We consider a CRN consisting of \( K \), \( K \geq 2 \), users, who coexist with one or more PUs in the same geographical area. We assume that each user has a unique ID and knows the largest possible ID of users (e.g., the network size \( N \)). The PUs are the holders of some licensed spectrum, which can be divided into \( M \), \( M > 1 \), non-overlapping channels. We assume that these channels are indexed uniquely as 1, 2, ..., \( M \) and the indices are known to every user in the network. The whole set of potentially available channels for the users is denoted by \( C = \{ c_1, c_2, \ldots, c_M \} \), in which \( c_i \) denotes the \( i \)th channel and usually is called channel \( i \) for convenience (\( i = 1, 2, \ldots, M \)). Each user in the CRN is equipped with one cognitive radio, which can access the idle channels in \( C \) opportunistically. Let \( C_i \subseteq C \) denote the set of currently available channels of user \( i \), \( i = 1, 2, \ldots, K \), where a channel is said to be available to a user if the user can communicate on the channel without causing interference to PUs. Let \( G \) denote the number of commonly available channels of all users, i.e., \( G = | \bigcap_{i=1}^{K} C_i | \). We consider the following two models.
i) **Symmetric model.** All users have the same available channels if their geographical locations are close. That is, for any $1 \leq i, j \leq K$, we have $C_i = C_j$. For simplicity, we assume $C_i = C_j = C$.

ii) **Asymmetric model.** Different users might have different available channels if their geographical locations are far away from each other. We assume that at least one commonly available channel is shared by the users who are involved in rendezvous, i.e., $G \neq 0$. This is a necessary assumption since there is no feasible solution, otherwise, for rendezvous of the users.

We consider the time-slotted CRNs, where all time slots are with the same fixed length. Our target is to design CH algorithms, which work without the need of time-synchronization of the network. Since time-synchronization is not available, different users may start their CH sequences at different time. The overlap of timeslots of two users should be sufficiently long for completing the rendezvous process. To this end, the duration of the time slot is set to the double of the necessary length for completing the rendezvous process. In this way, CH sequences of different users are equivalent to those in slot-aligned CRNs. Accordingly, the rendezvous of $K, K \geq 2$, users is said to be successful if all the users hop on a commonly available channel in the same time slot [3–4, 14]. The problem of our concern is formulated as follows.

**Rendezvous problem:** Given a multi-hop CRN consisting of $K, K \geq 2$, users, the problem is to design CH algorithms by which the users can generate their CH sequences on the basis of their currently available channels, and all users are guaranteed to hop on a commonly available channel in the same time slot, regardless of different time when the users start their CH sequences.

In reality, the available channel set of each user, $C_i (i = 1, 2, \ldots, K)$, changes dynamically due to mobility of CUs and activities of PUs. Thus, $C_i$ needs to be periodically updated to ensure correct operation of CH algorithms. In this study, we assume that CUs do not move fast and the activities of PUs are not frequently changed. In other words, $C_i$ keeps static during the update period, which is sufficiently long for the users to attempt rendezvous with potential neighbors.

TTR is an important metric to evaluate CH algorithms. It is defined as the number of time slots that it takes for users to achieve rendezvous once all users have begun their hopping. The TTR of a CH algorithm is usually not constant due to random starting time of CH sequences of the users. Thus, maximum TTR (MTTR) and expected TTR ($E(TTR)$) are usually adopted to evaluate the performance of CH algorithms. If a CH algorithm has a finite MTTR, then it is said to be with guaranteed rendezvous. In this study, we seek CH algorithms with guaranteed rendezvous for both symmetric and asymmetric models without the need of time-synchronization.
4 RING-WALK RENDEZVOUS ALGORITHMS

4.1 Basic Idea
Our basic idea is to represent each channel as a vertex in a ring, so that generating a CH sequence is equivalent to visiting vertices in the ring. We assume that the vertices are placed in the ring in clockwise direction according to their indices. For simplicity, we let indices of channels equal the corresponding indices of vertices. Each user “walks” on the ring by visiting vertices with a pre-assigned velocity in either clockwise or anticlockwise direction (clockwise direction is adopted in our algorithms, see Figure 1 for illustration). In each time slot, a user stops at a vertex (channel). In the next time slot, depending on the user’s velocity, the user either stays at the current vertex or jumps to next vertex. A CH sequence is generated when the user continuously walks on the ring. All users walk on the ring in the same direction. Intuitively, any two users with different velocities can eventually meet at a vertex since the user with lower velocity will be eventually “caught” by the user with higher velocity. That is, CH sequences of the two users will meet on a common channel in the same time slot and thus the rendezvous is achieved. Note that each user has a unique ID which could be used to design its distinct velocity. In this way, rendezvous is guaranteed since users walk on the ring with different velocities. In the following, we consider two designs of velocity based on the user ID.

FIGURE 1
Illustration of ring-walk rendezvous algorithms: a ring consists of 4 vertices/channels (vertex $i$ is corresponding to channel $i$, $i = 1, 2, \ldots, 4$); two users are walking on the ring in clockwise direction.
4.1.1 Ring-walk scheme 1

In this scheme, the user with ID $v$, $1 \leq v \leq N$, stays at each vertex for $v$ time slots before it advances by one vertex. It implies that the user with smaller $v$ will walk faster. For instance, suppose a ring consists of 4 vertices (i.e., $M = 4$). A user with $v = 2$ starts at vertex 1 and walks on the ring. Then, the CH sequence generated by the user is:

$$1, 1, 2, 2, 3, 3, 4, 4, 1, 1, 2, 2, 3, 3, 4, 4,...$$

The following CH function generates each channel-index in the CH sequence of the scheme.

```
Function RW1Hopping
1: Input: $M$, $v$, $i_0$, $t$
2: Output: channel $c$
3: $n = \lfloor t/v \rfloor$
4: $i = (i_0 + n - 1) \% M + 1$
5: RETURN $c = c_i$
```

In line 1, variable $i_0$ represents the starting channel-index and $t$ represents the time slot counter which starts from zero.

4.1.2 Ring-walk Scheme 2

In this scheme, each user with ID $v$ advances by $v$ vertices in the ring for every $N$ time slots. Correspondingly, CH sequence of the user is generated in a periodical way and each period lasts for $N$ time slots. Suppose that the user stops at vertex $i$ at the end of a period. The CH sequence in the next period is:

$$i \% M + 1, (i + 1) \% M + 1, \ldots, (i + v - 1) \% M + 1, (i + v - 1) \% M + 1, \ldots, (i + v - 1) \% M + 1.$$

According to the sequence, the user keeps moving forward on the ring in clockwise direction in the first $v$ time slots, and stays at the vertex in the remaining $N - v$ time slots of the period. In contrast to scheme 1, the user with larger $v$ will walk faster in this scheme. An illustration example is given next. Suppose $M = 4$, $N = 5$, and a user with $v = 2$ stops at vertex 1 when the previous period ends. In the next three consecutive periods, the CH sequence that is generated by the user is:

$$2, 3, 3, 3, 3, 4, 1, 1, 1, 2, 3, 3, 3, 3.$$

We can see that the user advances by 2 vertices for every 5 time slots. The CH function of this scheme is formally described as follows.
Function RW2Hopping
1: Input: $M, N, v, i_0, t$
2: Output: channel $c$
3: $n_1 = \lfloor t/N \rfloor; n_2 = t \mod N$
4: IF ($n_2 < v$) 
5: $i = (i_0 + n_1 \times v + n_2 - 1) \mod M + 1$
6: ELSE 
7: $i = (i_0 + n_1 \times v + v - 2) \mod M + 1$
8: RETURN $c = c_i$

In line 1, variable $i_0$ represents the starting channel-index and $t$ represents the time slot counter which starts from zero.

4.2 Algorithms for 2-user 1-hop Scenario
Based on the two hopping functions, i.e., $RW1Hopping$ and $RW2Hopping$, we will design, in this section, the corresponding CH algorithms for 2-user rendezvous under the symmetric and asymmetric model, respectively. Rigorous theoretical analysis on the algorithms’ correctness and performance will be presented as well.

4.2.1 Under the symmetric model
A) Algorithm design
To design CH algorithms for 2-user rendezvous under the symmetric model, we simply let each user in the network call the corresponding hopping function $RW1Hopping/RW2Hopping$ iteratively to generate the CH sequence. The process is continued until the desired rendezvous is achieved.

Based on $RW1Hopping$, the CH algorithm for 2-user rendezvous under the symmetric model, namely $RW1_2_SM$, is formally presented as follows.

Algorithm RW1_2_SM
1: Input: $M, ID$
2: $v = ID$; $i_0 = \text{rand}[1, M]$; $t = 0$
3: WHILE (not rendezvous)
4: $c = RW1Hopping(M, v, i_0, t)$; $t = t + 1$
5: Attempt rendezvous on channel $c$
6: END

In line 2, function “rand” is to randomly generate an integer in the specified interval. Similarly, $RW2_2_SM$ that is based on $RW2Hopping$ is formally presented as follows.

$RW1_2_SM$ and $RW2_2_SM$ have the same framework, but employ different hopping functions.
Algorithm RW2_2_SM
1: Input: M, N, ID
2: \( v = ID; \ i_0 = \text{rand}[1, M]; \ t = 0; \)
3: WHILE (not rendezvous)
4: \( c = \text{RW2Hopping}(M, N, v, i_0, t); \ t = t + 1; \)
5: Attempt rendezvous on channel \( c; \)
6: END

B) Algorithm analysis

In the following, we will analyze the correctness of algorithms (i.e., the algorithms guarantee rendezvous) and the performance of algorithms in terms of both MTTR and E(TTR). Note that generating a CH sequence is equivalent to visiting vertices in the ring. Therefore, successful rendezvous of users means that the users meet at the same vertex of the ring. We will conduct the analysis of algorithms on the basis of this ring-walk model.

The following lemma identifies how many time slots it takes to shorten the distance between two users by 1 vertex if they perform RW1_2_SM.

**Lemma 1.** Assume that user 1 with ID \( v_1 \) and user 2 with ID \( v_2 \) (\( v_1 < v_2 \)) walk on a ring which consists of \( M \) vertices. User 2 is one vertex ahead of user 1 in clockwise direction. It takes at most \( (\lceil (v_1 - 1)/(v_2 - v_1) \rceil + 2)v_1 \) time slots for user 1 to catch up with user 2 if they perform RW1_2_SM.

**Proof.** As mentioned before, one feature of RW1_2_SM is that the user with ID \( v \) should stay for \( v \) time slots at the current vertex before it moves to the next vertex. So, we denote by \( \Gamma_j(t) \) the remaining amount of time slots (including time slot \( t \)) that user \( j \) should still stay at the current vertex before it moves to the next vertex, and let \( \Gamma(t) = \Gamma_1(t) - \Gamma_2(t) \). Obviously, we have \( 1 \leq \Gamma_j(t) \leq v_j \) (\( j = 1, 2 \)) and \( \Gamma(t) \leq v_1 - 1 < v_2 - 1 \) (for \( v_1 < v_2 \)). Let \( t^{(0)} \) denote the time slot in which user 2 is ahead of user 1 by one vertex. We consider the three possible cases of \( \Gamma(t^{(0)}) \) as follows.

**Case 1:** \( \Gamma(t^{(0)}) < 0 \). In this case, after \( \Gamma_1(t^{(0)}) \) time slots, user 1 will move forward to the next vertex while user 2 will keep staying on the current vertex. This means that user 1 can catch up with user 2 after \( \Gamma_1(t^{(0)}) \) time slots.

**Case 2:** \( \Gamma(t^{(0)}) = 0 \). It can be easily verified that user 1 can catch up with user 2 after \( \Gamma_1(t^{(0)}) + v_1 \) time slots.

**Case 3:** \( \Gamma(t^{(0)}) > 0 \). In this case, after \( \Gamma_1(t^{(0)}) \) time slots, i.e., in time slot \( t^{(1)} = t^{(0)} + \Gamma_1(t^{(0)}) \), user 1 will move forward to the next vertex. Notice that \( t^{(1)} \) can be rewritten as \( t^{(1)} = t^{(0)} + \Gamma_2(t^{(0)}) + \Gamma(t^{(0)}) \) and \( \Gamma(t^{(0)}) < v_2 - 1 \). So, in time slot \( t^{(1)} \) user 2 will also move forward to the next vertex. That is, user 2 still keeps one vertex ahead of user 1 in time slot \( t^{(1)} \). However, we have \( \Gamma_1(t^{(1)}) = v_1, \ \Gamma_2(t^{(1)}) = v_2 - \Gamma(t^{(0)}) \) and thereby \( \Gamma(t^{(1)}) = \Gamma(t^{(0)}) - (v_2 - v_1) \) in time...
slot $t^{(1)}$. Inductively, in time slot $t^{(n)} = t^{(n-1)} + \Gamma_1(t^{(n-1)})$, user 2 will still keep one vertex ahead of user 1 if user 1 has not yet caught up with user 2; while we have $\Gamma_1(t^{(0)}) = v_1$, $\Gamma_2(t^{(n)}) = v_2 - \Gamma(t^{(n-1)})$ and $\Gamma(t^{(n)}) = \Gamma(t^{(n-1)}) - (v_2 - v_1) = \Gamma(t^{(n)}) - n(v_2 - v_1)$ ($n = 2, 3, \ldots$). Let $k = |\Gamma(t^{(0)}/(v_2 - v_1)|$, that is, $\Gamma(t^{(k)}) \leq 0$. Then, according to the above discussion in case 1 and case 2, we know that in time slot $(t^{(k)} + \Gamma_1(t^{(k)}))$ (if $\Gamma(t^{(k)}) < 0$, case 1) or in time slot $(t^{(k)} + \Gamma_1(t^{(k)}) + v_1)$ (if $\Gamma(t^{(k)}) = 0$, case 2) user 1 can definitely catch up with user 2. Here, it can be easily found that $t^{(k)} = t^{(0)} + \Gamma_1(t^{(0)}) + (k-1)v_1$ and $\Gamma_1(t^{(k)}) = v_1$.

Combining the above three cases, we conclude that the amount of time slots that user 1 needs to take to catch up with user 2 should not exceed $\Gamma_1(t^{(0)}) + (|\Gamma(t^{(0)}/(v_2 - v_1)| + 1)v_1$, which can be upper-bounded by $(|v_1 - v_2|/2)v_1$ since $\Gamma_1(t^{(0)}) \leq v_1$ and $\Gamma(t^{(0)}) \leq v_1 - 1$.

Note that ID of each user does not exceed $N$. Thus, if we let $v_1 = N - 1$ and $v_2 = N$, $(|v_1 - v_2|/2)v_1$ will reach the maximum value of $N(N - 1)$. Meanwhile, the longest distance between two users in the ring with $M$ vertices is $M - 1$. Combining these observations with Lemma 1, we can prove the correctness of $RW1_2_SM$ and obtain its MTTR in the following theorem.

**Theorem 1.** Under the symmetric model, any two users performing $RW1_2_SM$ achieve rendezvous in at most $(M - 1)(N^2 - N)$ time slots.

**Proof.** Suppose user 1 with ID $v_1$ and user 2 with ID $v_2$ ($v_1 < v_2$) perform $RW1_2_SM$ to rendezvous. Since user 2 is at most $(M - 1)$ vertices ahead of user 1 in clockwise direction, according to Lemma 1, user 1 can catch up with, i.e., meet with user 2 in at most $(M - 1)k$ time slots, where $k = (|v_1 - 1|/2)v_1$. On the other hand, user’s ID cannot exceed $N$, which means that $k \leq (N^2 - N)$ (k can reach the maximum value of $(N^2 - N)$ when $v_1 = N - 1$ and $v_2 = N$). Thus, it can be concluded that any two users performing $RW1_2_SM$ achieve rendezvous in at most $(M - 1)(N^2 - N)$ time slots.

We claim that the MTTR of $RW1_2_SM$ identified in Theorem 1 is tight in the order of $O(MN^2)$. To demonstrate this fact, we present an example as follows (see Figure 2).

![FIGURE 2](image)

An example for the rendezvous of two users by performing $RW1_2_SM$. In this example, $M = 3$, $N = 3$ and user A and B are with IDs equal to 3 and 2, respectively. User B is assumed to start its channel hopping with one time slot later than user A. The actual TTR in this example is 11, which is tightly close to the MTTR identified in Theorem 1, i.e., $(M - 1)(N^2 - N) = 12.$
Next, we analyze the upper bound of $E(TTR)$ of $RW1_2_SM$. We first introduce an inequality, which helps the analysis, in the following lemma.

**Lemma 2.** $(1 + 1/2 + \ldots + 1/(n-k))<1 + \ln n - k/n$, where both $k$ and $n$ are positive integers and $k<n$.

**Proof.** Since $\ln(1 + x) \leq x$ holds when $|x|<1$, and notice that $0<k/n<1$, we have $\ln((n-k)/n) = \ln(1-k/n) \leq -k/n$, that is, $\ln(n-k) \leq \ln n - k/n$. Based on this, we can get the desired inequality as follows:

$$(1 + 1/2 + \ldots + 1/(n-k))<1 + \ln(n-k) \leq 1 + \ln n - k/n.$$ □

**Theorem 2.** Under the symmetric model, the $E(TTR)$ of $RW1_2_SM$ can be upper-bounded by

$$\frac{M - 1}{N^2 - N} \left[\left(-\frac{1}{3} \ln N + \frac{5}{12}\right)N^2 - \left(\frac{1}{6} \ln N + \frac{1}{3} \right)N^2 + \left(\frac{2}{3} \ln N - \frac{5}{12}\right)N + \frac{1}{6}\right],$$

which is in the order of $O(MN\ln N)$.

**Proof.** Suppose user 1 with ID $v_1$ and user 2 with ID $v_2$ ($v_1<v_2$) perform $RW1_2_SM$ to rendezvous. Furthermore, suppose user 2 is $d$ ($1 \leq d \leq M-1$) vertices ahead of user 1 in clockwise direction. For fixed $d$, based on Lemma 1, we have the upper-bound of the conditional expected TTR as follows.

$$E(TTR \mid d) \leq \frac{1}{C^2} \sum_{v_1}^{N-1} \sum_{v_1 < v_2}^{N} (v_1 - 1) v_1 \left(v_2 - v_1 \right) + 2v_1, \quad (1)$$

Let $A = \sum_{v_1}^{N-1} \sum_{v_1 < v_2}^{N} (v_1 - 1) v_1 \left(v_2 - v_1 \right) + 2v_1$, then we have

$$A = 2\sum_{v_1}^{N-1} (N - v_1) v_1 + \sum_{v_1}^{N-1} (v_1 - 1) v_1 \sum_{v_1 < v_2}^{N} \frac{1}{v_2 - v_1}, \quad (2)$$

According to Lemma 2, we have $\sum_{k=1}^{N-1} \frac{1}{k} \leq 1 + \ln N - \frac{v_1}{N}$. Thus, if we let $B = \sum_{v_1}^{N-1} (v_1 - 1) v_1 \sum_{k=1}^{N-v_1} \frac{1}{k}$, then $B$ can be upper-bounded as follows.
\[
B \leq \sum_{v_i=1}^{N-1} (v_i - 1)v_i \left( 1 + \ln N - \frac{v_i}{N} \right)
= \left( 1 + \frac{1}{N} + \ln N \right) \sum_{v_i=1}^{N-1} v_i^2 - \left( 1 + \ln N \right) \sum_{v_i=1}^{N-1} v_i - \frac{1}{N} \sum_{v_i=1}^{N-1} v_i^3
\]

Combining (2) and (3), we can obtain

\[
A \leq (2N - \ln N - 1) \sum_{v_i=1}^{N-1} v_i + \left( \ln N + \frac{1}{N} - 1 \right) \sum_{v_i=1}^{N-1} v_i^2 - \frac{1}{N} \sum_{v_i=1}^{N-1} v_i
\]

Using the following three identity equations:

\[
\sum_{i=1}^{n} i = \frac{1}{2} n^2 + \frac{1}{2} n, \quad \sum_{i=1}^{n} i^2 = \frac{1}{3} n^3 + \frac{1}{2} n^2 + \frac{1}{6} n, \quad \text{and} \quad \sum_{i=1}^{n} i^3 = \frac{1}{4} n^4 + \frac{1}{2} n^3 + \frac{1}{4} n^2
\]

we can further simplify the inequality (4) as follows.

\[
A \leq \left( \frac{1}{3} \ln N + \frac{5}{12} \right) N^3 - \left( \ln N + \frac{1}{6} \right) N^2 + \left( \frac{2}{3} \ln N - \frac{5}{12} \right) N + \frac{1}{6}
\]

Combining (1) and (5), we can upper-bound \( E(TTR) \) as follows.

\[
E(TTR) = \frac{1}{M} \sum_{d=0}^{M-1} E(TTR \mid d) \\
\leq \frac{M-1}{N' - N} \left[ \frac{1}{3} \left( \ln N + \frac{5}{12} \right) N' - \left( \ln N + \frac{1}{6} \right) N' + \left( \frac{2}{3} \ln N - \frac{5}{12} \right) N + \frac{1}{6} \right]
\]

Similarly, we analyze algorithm \( RW2_2_SM \). We first present the following lemma which indicates an important property of \( RW2_2_SM \): a user with ID \( v \) advances by exact \( v \) vertices after any consecutive \( N \) time-slots.

**Lemma 3.** Suppose that a user with ID \( v \) walks on the ring which consists of \( M \) vertices. Let \( S(i) \) denote the channel-index of the \( i \)th element of the CH sequence which is generated by the user performing \( RW2_2_SM \). Equation \( S(t + N) = (S(t) + v - 1) \% M + 1 \) holds for any \( t = 0, 1, 2, \ldots \), where \( t \) is the time slot counter.

**Proof.** As described before, in \( RW2_2_SM \), the user walks on the ring in a periodical way; each period consists of \( N \) time slots. Hence, for the given time slot \( t \), we can assume that it is the \( j \)th time slot of some period \((j = 0, \ldots, N-1)\). Then, time slot \( t + N \) should be certainly the \( j \)th time slot of the next period as well. Recall \( RW2_2_SM \), we know that the vertices visited by the user according to this scheme in two consecutive periods should be indexed as follows:
Accordingly, we divide $j$ into the following two cases.

**Case 1:** $0 \leq j \leq v-1$. In this case, $S(t)$ can be written as $(i + j)\%M + 1$, and $S(t + N)$ should equal $(i + v + j)\%M + 1$. This implies that $S(t + N) = (S(t) + v-1)\%M + 1$ holds.

**Case 2:** $v \leq j < N$. In this case, $S(t)$ can be written as $(i + v-1)\%M + 1$, and $S(t + N)$ should equal $(i + 2v-1)\%M + 1$. It implies that $S(t + N) = (S(t) + v-1)\%M + 1$ holds. ■

Based on Lemma 3, we can derive the following lemma which identifies how many time slots it takes to shorten the distance between two users by $d$ vertices if they perform $RW2_2_SM$.

**Lemma 4.** Assume that user 1 with ID $v_1$ and user 2 with ID $v_2$ ($v_1 < v_2$) walk on a ring which consists of $M$ vertices. User 1 is $d$ vertices ahead of user 2 in clockwise direction. It takes at most $\left\lfloor d/(v_2-v_1)\right\rfloor N$ time slots for user 2 to catch up with user 1 if they perform $RW2_2_SM$.

**Proof.** According to Lemma 3, when a user with ID $v$ performs $RW2_2_SM$ it advances by exact $v$ vertices in the ring after any consecutive $N$ time slots. Let $k = \left\lfloor d/(v_2-v_1)\right\rfloor$, then after $kN$ time slots user 1 and user 2 can advance by $kv_1$ and $kv_2$ time slots respectively. Notice that user 1 is just $d$ vertices ahead of user 2 and $kv_1 + d \leq kv_2$. Hence, we can conclude that user 2 can catch up with user 1 in at most $kN = \left\lfloor d/(v_2-v_1)\right\rfloor N$ time slots. ■

Note that $d \leq (M-1)$ and $(v_2-v_1) \geq 1$. According to Lemma 4, we can easily prove the correctness of $RW2_2_SM$ and obtain its MTTR, which is identified in the following.

**Theorem 3.** Under the symmetric model, any two users performing $RW2_2_SM$ achieve rendezvous in at most $(M-1)N$ time slots.

**Proof.** Suppose user 1 with ID $v_1$ and user 2 with ID $v_2$ ($v_1 < v_2$) perform $RW2_2_SM$ to rendezvous. Since user 1 is at most $(M-1)$ vertices ahead of user 2 in clockwise direction and $(v_2-v_1) \geq 1$, according to Lemma 4, user 2 will catch up with user 1, i.e., they can achieve rendezvous in at most $(M-1) N$ time slots.

We claim that the MTTR of $RW2_2_SM$ identified in Theorem 3 is tight in the order of $O(MN)$, which is illustrated by the following example (see Figure 3).

An upper-bound of the E(TTR) of $RW2_2_SM$ is presented in the following theorem.
Theorem 4. Under the symmetric model, the $E(TTR)$ of $RW2_{-2}_{-SM}$ can be upper-bounded by $(M-1)(\ln N + 1/2)$, which is in the order of $O(M\ln N)$.

Proof. Suppose user 1 with ID $v_1$ and user 2 with ID $v_2$ ($v_1 < v_2$) perform $RW2_{-2}_{-SM}$ to rendezvous. Furthermore, suppose user 1 is $d$ $(1 \leq d \leq M-1)$ vertices ahead of user 2 in clockwise direction. For fixed $d$, based on Lemma 4, we have the upper-bound of the conditional expected TTR as follows.

$$E(TTR \mid d) \leq \frac{1}{C_N^2} \sum_{v_1}^{N} \sum_{v_2 > v_1}^{N} \frac{d}{v_2 - v_1} N$$

$$= \frac{dN}{C_N^2} \sum_{v_1}^{N} \sum_{v_2 > v_1}^{N} \frac{1}{v_2 - v_1} = \frac{dN}{C_N^2} \sum_{v_1}^{N} \sum_{k=1}^{N-v} \frac{1}{k}$$

According to Lemma 2, we have $\sum_{k=1}^{N-v} \frac{1}{k} \leq 1 + \ln N - \frac{v}{N}$. Therefore, we can further upper-bound $E(TTR \mid d)$ as follows.

$$E(TTR \mid d) \leq \frac{dN}{C_N^2} \sum_{v=1}^{N-1} \left(1 + \ln N - \frac{v}{N}\right) = (2\ln N + 1)d$$

And, we also have

$$E(TTR) = \frac{1}{M} \sum_{d=1}^{M-1} E(TTR \mid d) \leq (M-1)(\ln N + 1/2).$$

4.2.2 Under the asymmetric model

A) Algorithm design

Under the asymmetric model, two users might have different available channel sets $C_j$, $j = 1, 2$. Our basic idea is to expand $C_j$ to $C$ and apply the function $RW1Hopping/RW2Hopping$ to generate CH sequences in the similar way of the algorithm $RW1_{-2}_{-SM}/RW2_{-2}_{-SM}$. Since $C_j$ is expanded to $C$, two users may rendezvous on a channel of $C$ which is not available to one or both of
them and the rendezvous is not actually achieved. To tackle the problem, we periodically update the parameter of starting index in the function \textit{RW1Hopping/RW2Hopping}, such that each user restarts its ring-walk at different starting vertex. In this way, every channel of \( C \) has the chance for the rendezvous of users and thus two users can eventually achieve rendezvous on a commonly available channel as long as \( C_1 \cap C_2 \neq \phi \).

Based on \textit{RW1Hopping}, we design the corresponding algorithm for 2-user rendezvous under the asymmetric mode, namely \textit{RW1\_2\_AM}, as follows.

**Algorithm RW1\_2\_AM**

1: \textbf{Input}: \( M, N, ID \)
2: \( v = ID; i_0 = \text{rand}[1, M]; t = 0; \Delta = 2(M-1)(N^2-N); \)
3: \textbf{WHILE} (not rendezvous)
4: \( n = \lfloor t/\Delta \rfloor; i = (i_0 + n-1)\% M + 1; \)
5: \( c = \text{RW1Hopping}(M, v, i, t); t = t + 1; \)
6: Attempt rendezvous on channel \( c \);
7: \textbf{END}

In line 4 of \textit{RW1\_2\_AM}, the starting index (i.e., variable \( i \) of function \textit{RW1Hopping}) is updated every \( \Delta = 2(M-1)(N^2-N) \) time slots. Note that \( \Delta \) is exactly the double of \textit{RW1\_2\_SM}'s MTTR, which is identified in Theorem 1. We set the value of \( \Delta \) in this way to ensure that CH sequences of any two users overlap at least \( (M-1)(N^2-N) \) time slots regardless of different starting time of their CH sequences.

Similarly, based on \textit{RW2Hopping}, we design the corresponding algorithm for 2-user rendezvous under the asymmetric mode, namely \textit{RW2\_2\_AM}, as follows.

**Algorithm RW2\_2\_AM**

1: \textbf{Input}: \( M, N, ID \)
2: \( v = ID; i_0 = \text{rand}[1, M]; t = 0; \Delta = 2(M-1)N; \)
3: \textbf{WHILE} (not rendezvous)
4: \( n = \lfloor t/\Delta \rfloor; i = (i_0 + n-1)\% M + 1; \)
5: \( c = \text{RW2Hopping}(M, N, v, i, t); t = t + 1; \)
6: Attempt rendezvous on channel \( c \);
7: \textbf{END}

In \textit{RW2\_2\_AM}, we set \( \Delta = 2(M-1)N \), i.e., double of \textit{RW2\_2\_SM}'s MTTR which is identified in Theorem 3.

\textbf{B) Algorithm analysis}

The following two theorems present the upper-bounds of MTTR for \textit{RW1\_2\_AM} and \textit{RW2\_2\_AM}, respectively, where \( M \) is the size of the whole channel.
set, \(G\) is the number of commonly available channels of users, and \(N\) is the network size.

**Theorem 5.** Under the asymmetric model, any two users performing \(RW1_2_AM\) achieve rendezvous in at most \(2(M + 1 - G)(M - 1)(N^2 - N)\) time slots.

**Proof.** Without loss of generality, we assume that user 1 begins performing \(RW1_2_AM\) not later than user 2. According to the running scheme of \(RW1_2_AM\) (see the example in Figure 4 for reference), each user performs its ring-walk in rounds. When a round of ring-walk ends, the user will restart a new round of ring-walk by just switching the starting vertex to the next one (in clockwise direction). Since each round lasts for \(\Delta = 2(M - 1)(N^2 - N)\) time slots, any two users should overlap at least \((M - 1)(N^2 - N)\) time slots before either one restarts its ring-walk. In other words, there should be at least \((M - 1)(N^2 - N)\) overlapping time slots between the first round of user 2 and some round of user 1. According to Theorem 1, among these overlapping time slots there should exist one in which the two users will achieve their first “meeting” at some vertex, say \(i\). Similarly, after \(\Delta\) time slots they will achieve the second meeting at another vertex, say \(j\). Since each user switches its starting vertex to the next one in clockwise direction when it begins a new round of ring-walk, we argue that vertex \(j\) must be different with vertex \(i\). More precisely, we actually have \(j = i \% M + 1\), i.e., vertex \(j\) is just the next vertex to vertex \(i\) (in clockwise direction). Inductively, we conclude that the two users can meet at all the \(M\) different vertices during the first \(M\) rounds (i.e., \(M\Delta\) time slots) of user 2. Since there are \(G\) channels commonly available to both

![FIGURE 4](image)

An illustrative example for the running scheme of \(RW1_2_AM\). In this example, \(M = 3, N = 3\) and user A and B are with IDs equal to 3 and 2, respectively. We will show that two users achieve rendezvous on every channel of \(C\). Users A and B start their ring-walk at vertex 3 and vertex 2, respectively. User B starts its ring-walk 3 time slots later than user A. The underlined numbers denote those starting indices in different rounds, and each round consists of exactly \(\Delta = 2(M - 1)(N^2 - N) = 24\) time slots. Each pair of shadowed cells in the figure represents a meeting/rendezvous between users at/on a vertex/channel. It is clear that the rendezvous is achieved on every channel of \(C\).
users, they can achieve rendezvous in at most \((M + 1 - G)\Delta = 2(M + 1 - G)(M - 1)(N^2 - N)\) time slots.

**Theorem 6.** Under the asymmetric model, any two users performing RW2_2_AM achieve rendezvous in at most \(2(M + 1 - G)(M - 1)N\) time slots.

**Proof.** Similar to the proof of Theorem 5.

### 4.3 Algorithms for Multi-user Multi-hop Scenario

All the rendezvous algorithms described in the previous section can be smoothly extended to the multi-user and multi-hop scenario. Our basic idea is to apply the pair-wise rendezvous of two users to achieve the global rendezvous of all users. Note that two users can exchange information over a commonly available channel if the rendezvous is successfully achieved on this channel. Once two users achieve rendezvous, parameter setting of their CH functions (i.e., RW1Hopping or RW2Hopping) can be synchronized, so that the two users will generate the identical CH sequence afterward.

Take RW1Hopping for example, there are three parameters involved: \(v, i_0\) and \(t\) (parameter \(M\) is identical to all users in the network). The three parameters can be represented as a 3-tuple \((v, i_0, t)\). Note that users have distinct IDs. We define the partial order of these 3-tuples as follows. \((v_2, i_{02}, t_2)\) is said to be greater than \((v_1, i_{01}, t_1)\) if and only if \(v_2 < v_1\). Once rendezvous of two users is successful, they exchange their 3-tuples. The user who holds the smaller 3-tuple will follow its neighbor by updating its 3-tuple with the larger one. Afterwards, the two users use the identical and synchronized CH sequence for rendezvous with other users in similar way. This process is continued until rendezvous of all users is achieved (all users eventually use the same and the largest 3-tuple). Recall that in RW1_2_SM/RW1_2_AM, the user with smaller ID \(v\) walks faster than the user with larger ID \(v\). Therefore, the user that is walking faster will “lead” other users once they meet at some vertex in the ring.

Based on RW1Hopping, we design the CH algorithm for rendezvous of multiple users under the symmetric model, RW1_Multi_SM, as follows.

---

**Algorithm RW1_Multi_SM**

1: **Input**: \(M, ID\)
2: \(v = ID; i_0 = \text{rand}[1, M]; t = 0;\)
3: **WHILE** (not terminated)
4: \(c = \text{RW1Hopping}(M, v, i_0, t); t = t + 1;\)
5: Attempt rendezvous on channel \(c; //\text{Comment 1}\)
6: **IF** (success in rendezvous) //**Comment 2**
7: \(q = \text{argmax}\{(v_q, i_{0q}, t_q)\}; //\text{Comment 3}\)
8: **IF** \((v, i_0, t) < (v_{q}, i_{0q}, t_q)) (v, i_0, t) = (v_{q}, i_{0q}, t_q); END\)
9: **END**
10: **END**
The CH algorithm for rendezvous of multiple users under the asymmetric model, \( RW1\_Multi\_AM \), is given as follows.

Algorithm RW1_Multi_AM
1: **Input**: \( M, N, ID \)
2: \( v = ID; i_0 = \text{rand}[1, M]; t = 0; \Delta = 2(M-1)(N^2-N); \)
3: **WHILE** (not terminated)
4: \( n = \lfloor t/\Delta \rfloor; i = (i_0 + n-1)\%M + 1; \)
5: \( c = \text{RW1Hopping}(M, v, i, t); t = t + 1; \)
6: Attempt rendezvous on channel \( c \); //Comment 1
7: **IF** (success in rendezvous) //Comment 2
8: \( q = \arg \max \{ (v_q, i_{0q}, t_q) \}; //Comment 3 \)
9: **IF** \( (v, i_0, t) < (v_q, i_{0q}, t_q) \) \( \text{ END } \)
10: **END**
11: **END**

Remark: Comment 1 — exchange 3-tuple \( (v, i_0, t) \) with its potential neighbors; Comment 2 — receive 3-tuples \( (v_q, i_{0q}, t_q) \) from its neighbors; Comment 3 — select the largest 3-tuple among all received 3-tuples.

Based on \( RW2\_Hopping \), we can similarly design the corresponding CH algorithms \( RW2\_Multi\_SM \) and \( RW2\_Multi\_AM \), which work for rendezvous of multiple users under the symmetric model and the asymmetric model, respectively. The only difference is the definition on the partial order of 3-tuples. Specifically, since the user with larger ID \( v \) walks faster in \( RW2 \), the partial order of 3-tuples is defined as follows: \( (v_2, i_{02}, t_2) \) is said to be greater than \( (v_1, i_{01}, t_1) \) if and only if \( v_2 > v_1 \). To avoid repetition of similar algorithms, we omit the details of \( RW2\_Multi\_SM \) and \( RW2\_Multi\_AM \). Readers can easily implement the algorithms based on \( RW2\_Hopping \).

The following two theorems identify the upper-bounds of MTTR for \( RW1\_Multi\_SM, \ RW2\_Multi\_SM, \ RW1\_Multi\_AM \) and \( RW2\_Multi\_AM \), where \( M \) is the size of the whole channel set, \( G \) is the number of commonly available channels of users, \( N \) is the network size, \( K \) is the number of active users attempt to rendezvous in the network, and \( D \) is the network diameter (in terms of the number of hops).

**Theorem 7.** Under the symmetric model, \( K, K \geq 2 \), users performing \( RW1\_Multi\_SM \) and \( RW2\_Multi\_SM \) achieve rendezvous in at most \( (M-1)(N+1-K)[2D + (N-K)(1 + \ln D)] \) and \( (M-1)N(1 + \ln D) \) time slots, respectively.

**Proof.** Among the \( K \) users involved in the rendezvous process, we assume user \( i \) has ID \( v_i \) \((i = 1, 2, \ldots, K)\). Here, we analyze \( RW1\_Multi\_SM \) first. For the convenience of analysis, we assume \( v_1 \) is the smallest ID, that is, user 1 holds the largest 3-tuple among the \( K \) users. According to the running scheme
of RW1_Multi_SM, once rendezvous with user 1 successfully, other user will update its 3-tuple with that of user 1 and then generate the CH sequence which is identical to that of user 1. Suppose user $j (j \neq 1)$ is a 1-hop neighbor of user 1, then according to Lemma 1 user $j$ and user 1 can achieve rendezvous in at most $(M-1)((v_j-1)/(v_j-v_1)+2)v_1$ time slots. Afterwards, user $j$ and user 1 will hold the same 3-tuple and use the identical and synchronized CH sequence to rendezvous with other users. In other words, when all 1-hop neighbors of user 1 have achieved rendezvous with it, they can afterwards behave the same as user 1 in channel-hopping and achieve rendezvous with 2-hop neighbors of user 1 (if any) in the similar way. Based on this, we conclude that the number of time slots for global rendezvous of all users can be upper-bounded by $(M-1)A$, where $A$ is defined as follows.

$$A = \max \left\{ \sum_{k=2}^{D+1} v_i \left( \frac{v_1-1}{v_k-v_i} + 2 \right) \mid v_k < v_{k+1}, 1 \leq v_k \leq N, k = 1, \ldots, D \right\}$$

For fixed $v_1$, we have the following inequality (note that we have the constraints $v_k < v_{k+1}, k = 1, \ldots, D$).

$$\sum_{k=2}^{D+1} v_i \left( \frac{v_1-1}{v_k-v_i} + 2 \right) \leq \sum_{k=1}^{D} v_i \left( \frac{v_1-1}{v_1+k-v_i} + 2 \right) = 2Dv_1 + v_i(v_1-1)\sum_{k=1}^{D} \frac{1}{k} \leq 2Dv_1 + v_i(v_1-1)(1+\ln D) \quad (8)$$

On the other hand, since there are $K$ users and $v_1$ is the smallest ID, we must have $v_1 \geq (N+1-K)$. Combining this with (8), we obtain that $A \leq (N+1-K) [2D + (N-K)(1 + \ln D)]$. Hence, we can further conclude that, when $K$ users perform RW1_Multi_SM, they can achieve rendezvous in at most $(M-1)(N+1-K)[2D + (N-K)(1 + \ln D)]$ time slots.

RW2_Multi_SM can be analyzed in the similar way if we assume $v_1$ is the largest ID instead so as to make user 1 hold the largest 3-tuple among the $K$ users. In this case, based on Lemma 4, we can conclude that the number of time slots for global rendezvous of all users can be upper-bounded by $(M-1)B$, where $B$ is defined as follows.

$$B = \max \left\{ \sum_{k=2}^{D+1} \frac{N}{v_i-v_k} \mid v_k > v_{k+1}, 1 \leq v_k \leq N, k = 1, \ldots, D \right\}$$

For fixed $v_1$, we have the following inequality (note that we have the constraints $v_k < v_{k+1}, k = 1, \ldots, D$).

$$\sum_{k=2}^{D+1} \frac{N}{v_i-v_k} \leq \sum_{k=1}^{D} \frac{N}{v_i-(v_i-k)} = N\sum_{k=1}^{D} \frac{1}{k} \leq N(1+\ln D) \quad (9)$$

Therefore, we can conclude that when $K$ users perform RW2_Multi_SM they can achieve rendezvous in at most $(M-1)N(1 + \ln D)$ time slots. ■
Theorem 8. Under the asymmetric model, \( K, K \geq 2 \), users performing \( \text{RW1\_Multi\_AM} \) and \( \text{RW2\_Multi\_AM} \) achieve rendezvous in at most \( 2(M+1-G)(M-1)(N^2-N)D \) and \( 2(M+1-G)(M-1)ND \) time slots, respectively.

Proof. Here, we only analyze \( \text{RW1\_Multi\_AM} \); \( \text{RW2\_Multi\_AM} \) can be analyzed in the similar way. First, we notice that among the \( K \) users involved in the rendezvous process there should be a user, say \( u \), which holds the largest 3-tuple. When all users have begun the rendezvous process, according to Theorem 5, we know that \( u \) will meet all its 1-hop neighboring users after at most \( 2(M+1-G)(M-1)(N^2-N) \) time slots. Once rendezvous with \( u \), the users will update their 3-tuples with the largest one and continue the rendezvous process. After at most \( 2(M+1-G)(M-1)(N^2-N) \) time slots, rendezvous of all the 2-hop neighboring users of \( u \) will be achieved. Notice that the network diameter is \( D \). Hence, inductively, we can conclude that when \( K \) users perform \( \text{RW2\_Multi\_AM} \) they can achieve rendezvous in at most \( 2(M+1-G)(M-1)(N^2-N)D \) time slots. Similarly, based on Theorem 6 it can be proofed that when \( K \) users perform \( \text{RW2\_Multi\_AM} \) they can achieve rendezvous in at most \( 2(M+1-G)(M-1)ND \) time slots.

5 SIMULATION

Matlab 7.9 is adopted to implement our RW1 and RW2 algorithms. Three representative algorithms, i.e., GOS, MC and MMC [6, 14], are selected for comparison since they have been shown to achieve the overall best performance in the literature [14]. GOS and MC work for two users under the symmetric model while MMC works for two users under the asymmetric model. MTTR and E(TTR) are for theoretical analysis which has been studied in Section 4. In simulation, the average TTR is used to evaluate performance of the algorithms. For simplicity, we set \( G=1 \) and \( D=1 \) in the simulation. That is, there is only one commonly available channel of users and the network diameter is 1 (our algorithms are applicable to the case of \( G>1 \) and \( D>1 \)). Under the asymmetric model, we randomly select channels from \( C \) to \( C_i \) by using a ratio \( \theta \) (\( 0<\theta<1 \)) to control the size of \( C_i \). That is, each user has \( \theta M \) available channels on average.

To fully utilize chances for rendezvous of users under the asymmetric model, we integrate a random-replacement operation into the corresponding RW1/RW2 algorithms, in which each user randomly replaces unavailable channels with its available ones in the CH sequence. In fact, the similar strategy was adopted in MMC [14]. In each simulation run, we randomly generate the staring time for each user to start its CH sequence. The results presented in the following figures are the means of 50 separate runs.
5.1 2-user Rendezvous

Figure 5 shows the average TTR of our algorithms and GOS and MC under the symmetric model. To show how tight the MTTR upper-bounds of different algorithms are (see Table 1 for a summary), we first try to plot them with the average TTRs in the same figures, i.e., in Figure 5(a) and 5(b), where the legend marked with “MTTR-UpperBound” is used to denote the MTTR upper-bound of the corresponding algorithm. Notice that we have not plotted the MTTR upper-bound of MC in the figures since MC has been proved without finite MTTR [14]. As we can see in Figure 5(a) and 5(b), the MTTR upper-bounds of all algorithms, including GOS, are far greater than the corresponding average TTRs. Actually, these upper-bounds on MTTR are so loose that the average TTRs cannot be shown clearly if we plot them in the same figure. Therefore, to highlight the comparison of different algorithms in terms of average TTR as well as to save the space, we decide to not any more plot the MTTR upper-bounds and only plot the average TTRs, which are shown in Figure 5(c) and 5(d).

From Figure 5(c) where \( N \) is set to 10, we can see that the average TTRs of the four algorithms all increase with \( M \), especially for that of RW1 and GOS. Overall speaking, RW2 and MC outperform RW1 and GOS, while RW2 is competitive with MC. This observation can be further confirmed by the results shown in Figure 5(d) where \( N \) is set to 100. As shown in the previous theoretical analy-
sis, for 2-user under the symmetric model, the $E(TTR)$ of RW1 and RW2 can be upper-bounded by a value within the order of $O(MN\ln N)$ (see Theorem 2) and $O(M\ln N)$ (see Theorem 4), respectively. On the other hand, according to Table 1, we know that the $E(TTR)$ of GOS can be upper-bounded by a value within the order of $O(M^2)$, while that of MC can be upper-bounded by $2P^2/(P–1)$, where $P$ is the smallest prime number greater than or equal to $M$. Thus, theoretically speaking, in this scenario both RW2 and MC should outperform RW1 and GOS, and our simulation results verify this fact. It is worth pointing out that, though MC has no finite MTTR, it shows almost the best performance in terms of average TTR. Furthermore, when $N$ is relatively small, RW1 should be superior to GOS, which is verified by the results in Figure 5(c) where $N=10$; however, when $N$ become large, say $N=100$, as illustrated in Figure 5(d), RW1 is obviously inferior to GOS.

Figure 6 shows the success rate of our algorithms and GOS and MC under the symmetric model, where *success rate* is defined as the rate that two users achieve rendezvous successfully in one trial when they perform the algorithms. It should be pointed out that MC cannot provide guaranteed rendezvous. Actually, the users running MC need to randomly select different rates in some interval, which may fail in a trial if the randomly selected rates are identical [14]. In our simulation, we allow MC to repeatedly attempt rendezvous until it is achieved. Since our algorithms and GOS have guaranteed rendezvous, their success rates in a trial are always 100%. The success rate of MC is around 70% to 80%, as shown in Figure 6. Note that success rate of MC is affected by only parameter $M$.

![FIGURE 6](image)

Success rate of RW1, RW2, GOS and MC for 2-user rendezvous under the symmetric model.
Figure 7 shows the average TTR of our algorithms and MMC under the asymmetric model. As demonstrated in Figure 7(a) and 7(b), by fixing $\theta (= 0.5)$, we can find that the average TTRs of the three algorithms all increase with $M$, and MMC is overall superior to both RW1 and RW2 for varying $M$, especially for big $M$ (e.g., exceeding 40). In addition, we can find that RW2 outperforms RW1 slightly in these cases. If fixing $M$ but varying $\theta$, we can see that, according to Figure 7(c) and 7(d), the average TTRs of the three algorithms increase with $\theta$ as well. However, according to Figure 7(c), it seems that for fixed $M$ both RW1 and RW2 can compete with and sometimes even outperform MMC slightly when $N$ is relatively small (say $N=20$). The explanation for this phenomena maybe lie in this fact: regarding TTR, both RW1 and RW2 depend on parameter $N$, while MMC doesn’t. More specifically, when $M$ is fixed, the smaller $N$ is, the better performance (in terms of average TTR) RW1 and RW2 can achieve. Therefore, as illustrated in Figure 7(c), in simulation we may observe both RW1 and RW2 outperform MMC when $N$ is relatively small.

Comparing Figure 5 with Figure 7, it seems that the average TTRs of the algorithms (especially for that of RW1) under the asymmetric model have lower order of magnitude than that under the symmetric model. The main reason for this is that the integrated random-replacement operation significantly increases the chances for users to rendezvous. The lower $\theta$ is, i.e., the
fewer channels are available for the users, more frequently the random-replacement operation takes place. In the extreme case, when \( \theta \) approaches to zero, the algorithm integrated with the random-replacement operation would perform alike a pure random algorithm.

Figure 8 shows the success rate of our algorithms and MMC under the asymmetric model. Similar to MC, though MMC is efficient in average TTR, MMC actually cannot guarantee rendezvous of users in a trial, which is verified by the success rates shown in Figure 8. Note that success rate of MMC is affected by only parameter \( M \).

5.2 Multi-user Rendezvous

Figure 9 shows the average TTR of our algorithms and GOS for multi-user rendezvous under the symmetric model. For the purpose of comparison with our algorithms, we extend GOS to the multi-user scenario by using the similar strategy which is adopted in our algorithms. Specifically, the users running GOS exchange information of their time slot counters after rendezvous; each user updates its time slot counter with the largest one among all received counters. According to Figure 9(a) and 9(b), when the number of active users \( K \) is fixed, the average TTRs of the three algorithms all increase with \( M \). When \( K \) is small, as shown in Figure 9(a), both RW1 and RW2 outperform GOS and performance of RW2 is better than that of RW1. However, when \( K \) is large, as shown in Figure 9(b), RW1 is obviously superior to the other two algorithms. This fact can be further confirmed by Figure 9(c) and 9(d), in which we fix \( M \) but vary \( K \); it is clear that RW1 performs best among the three algorithms when \( K \) is large, particularly near to the network size \( N \). This
Z. Lin et al. implies that RW1 is scalable to deal with the rendezvous of large-scale networks under the symmetric model. We argue that this experimental observation on the characteristic of RW1 is highly consistent with the theoretical result obtained previously. Actually, according to Theorem 7, we know that for multi-user rendezvous under the symmetric model the $E(TTR)$ of RW1 can be upper-bounded by $(M-1)(N+1-K)[2D + (N-K)(1 + \ln D)]$, where $D$ is network diameter and it is set to 1 in our simulation. Clearly, this bound will decrease as $K$ increases. Extremely, when $K$ approaches $N$, i.e., when the number of active users is near to the network size, the $E(TTR)$ of RW1 is in the order of $O(M)$ and RW1 will perform very well.

Only our algorithms can work for multi-user rendezvous under the asymmetric model. Figure 10 shows the average TTR of RW1 and RW2. It is shown in Figure 10(a) and 10(b) that RW2 outperforms RW1 when $K$ is fixed and $M$ is varied. Nevertheless, RW1 is competitive with RW2 when $M$ is fixed and $K$ is varied, especially when $K$ is relatively large, as demonstrated in Figure 10(c) and 10(d). In all the above cases, we fix $\theta$ as 0.5. However, if $\theta$ is varied, as shown in Figure 10(e) and 10(f), the similar observation can be found, that is, RW1 is competitive with RW2 for large $K$. And, similar to the case under the symmetric model, it implies again that, compared with RW2, RW1 is more suitable to the rendezvous of users with large-scale.
6 CONCLUSIONS AND FUTURE WORK

In this paper, we have studied the channel-hopping algorithms for blind rendezvous in CRNs. We consider rendezvous of 2-user and multi-user scenarios under both symmetric and asymmetric models. We have proposed two ring-walk rendezvous algorithms which guarantee rendezvous of $K, K \geq 2$, users in all scenarios without the need of time-synchronization. We have shown the efficiency of our algorithms by solid theoretical analysis and extensive simulations.
According to the simulation results as well as the theoretical analysis, we have the following conclusions:

1. RW1 and RW2 provide guaranteed rendezvous for both 2-user and multi-user rendezvous. The success rates of MC and MMC are around 65%-85% for 2-user rendezvous.

2. Though without guaranteed rendezvous, MC and MMC are efficient for 2-user rendezvous in terms of average TTR. When the network size $N$ is small, RW2 is competitive with MC for 2-user rendezvous under the symmetric model.

3. Employing random-replacement operations can significantly enhance the performance of both RW1 and RW2 for 2-user rendezvous under the asymmetric model.

4. For multi-user rendezvous, especially under the symmetric model, RW1 performs better than RW2 when the number of users involved in rendezvous, i.e. $K$, is large and close to the network size $N$.

In our ongoing work, we use a different approach, called jump-stay, to achieve rendezvous [11]. There are several open problems in rendezvous. First, what are the smallest MTTR under the symmetric and asymmetric models? Trivial lower-bounds of MTTR under the symmetric and asymmetric models are 1 and $M$, respectively. We conjecture that the tight lower-bound of MTTR under the symmetric model is $M$ and the tight lower-bound of MTTR under the asymmetric model is $P^2$ where $P$ is the smallest prime number not less than $M$. Second, study of lower-bound of $E(TTR)$ is missing. Note that MTTR is the achievable upper-bound of TTR while $E(TTR)$ is the expected value of TTR. A rendezvous algorithm with a small MTTR does not necessarily guarantee a small $E(TTR)$. In practice, $E(TTR)$ is more important for evaluating performance of rendezvous algorithms.

ACKNOWLEDGEMENT

This work is supported in part by Research Grants of Hong Kong Baptist University [Project No. FRG2/09-10/088 and FRG1/10-11/036]. Zhiyong Lin is supported by the Natural Science Foundation of Guangdong Province in China (Grant No. S2011040002890).

REFERENCES


