Chapter 16
Oblivious Blind Rendezvous for Multi-user Multihop CRN

Abstract In this chapter, we propose the distributed oblivious blind rendezvous algorithm for multiple users in a multi-hop distributed system. As described in Problem 11.2, the system consists of M users with distinct identifers (IDs) in $[1, \hat{M}]$, where $\hat{M} \geq M$ and $\hat{M} \leq N^c$ for a constant value c (for simplicity, we re-use notation M to mean $\hat{M}$). Any two users in the system are connected within D hops, which implies the network diameter is D. In Sect. 16.1, we describe the algorithm for multiple users in a multi-hop system, and the correctness is presented in Sect. 16.2. We summarize the chapter in Sect. 16.3.

16.1 Algorithm Description

We adopt the intuitive idea in [1–3] to extend the algorithms for OBR-2 to multiple users: once every two users achieve rendezvous on a common port successfully, they can exchange their local information over the established communication link such that the input of the OBR-2 algorithms can be synchronized; then they would generate the same hopping sequence afterwards. Assume there are four parameters (I, M, N, C) (I is the user's ID, M is the maximum value of the ID, N is the number of all ports, and C is the set of available ports) used for each user and we extend the MSH algorithm (Algorithm 13.5) in Chap. 13 to the multi-user multi-hop scenario.

Algorithm 16.1 Rendezvous Algorithm for Multiuser Multihop Scenario

1: *Input:* I, M, N, C;
2: **while** Not terminated **do**
3: Run the MSH algorithm (Alg. 13.5) with input (I, M, N, C);
4: **if** Rendezvous with user - (I', M, N, C') **then**
5: $I := \min(I, I')$;
6: $C := C \bigcap C'$;
7: Synchronize labels of the ports in C according to the user with smaller I;
8: **end if**
9: **end while**

Z. Gu et al., *Rendezvous in Distributed Systems*,
DOI 10.1007/978-981-10-3680-4_16

As described in Algorithm 16.1, the user runs the MSH algorithm with local parameters (I, M, N, C). Once rendezvous is achieved with another user with parameters (I', M, N, C'), they exchange their information and three operations are executed:

(1) Change I to be the smaller value between I, I'[1];
(2) change C to be the intersection of C and C';
(3) synchronize the labels for the available ports in C according to the user with smaller I value such that $\forall i \in [1, |C|], c(i) = c'(i)$;

After these three steps, the four parameters of the two users are the same and they access the ports with the same hopping sequence until the next rendezvous happens. We derive the correctness and time complexity in Theorem 16.1.

16.2 Correctness and Complexity

Theorem 16.1 *Algorithm 16.1 guarantees that all users can achieve rendezvous in* $MTTR = O(N^2 D \log_N M)$ *time slots, where* D *is the diameter of the system.*[2]

Proof The theorem can be concluded through induction, similar to Theorem 10.1. We show that all users can update to the same I value, the same set of available ports $\bigcap_{i=1}^{m} C_i$ and the same labels of the ports in $O(N^2 D \log_N M)$ time slots.

By adopting the same analysis in Theorem 10.1, all users can generate the same set of available ports $\bigcap_{i=1}^{m} C_i$ after $4lNP * D = O(N^2 D \log_N M)$ time slots when all users have begun the rendezvous process. We show that all users can update to the same I value through induction. Denote the user with the smallest ID value as r, and we show that any user r_i can update the I value as user r's in $4lNP * d(r, r_i)$ time slots where $d(r, r_i)$ represents the minimum number of hops separating them.

(1) When $r_i = r$, $d(r, r_i) = 0$, it is satisfied obviously.
(2) Suppose for any user r_k with $d(r, r_k) \leq k$, it updates I as the ID value of user r in $4lNP * d(r, r_k)$ time slots. Consider user r_{k+1} with $d(r, r_{k+1}) = k + 1$ and suppose it is connected to some user r_k with $d(r, r_k) = k$ (it is easy to see that such user must exist); user r_k has already updated the I value to be the same as user r in $4lNP * k$ time slots and user r_{k+1} can update the value as that of user r_k in $4lNP$ time slots when they rendezvous on a some common available port in $4lNP$ time slots from Theorem 13.3. Therefore, user r_{k+1} can also update I to have the ID value of user r in $4lNP * (k + 1)$ time slots.

Combining these two aspects, where D is the network diameter, all users in the network can update the I value as that of user r in $4lNPD = O(N^2 D \log_N M)$ time slots (the process can be thought of as user r sends its ID value to all users within D hops). After another $4lNPD$ time slots, all users can synchronize the labels of

[1] It does not mean the user really changes the ID value, but it only changes the input of the rendezvous algorithm.

[2] l is defined as in Theorem 13.3.

the available ports as user r which has the smallest ID value. Therefore, the users in the network would hop through the ports according to the same sequence after $O(N^2 D \log_N M)$ time slots. So the theorem holds.

16.3 Chapter Summary

In this chapter, we extend oblivious blind rendezvous algorithms between two users to multiple users in a multihop distributed system. Similar to the non-oblivious blind rendezvous problem, every two neighboring users can rendezvous on a common available port and their local information can be synchronized. Then, they can repeat the rendezvous attempt until all users finally access the same available port.

Similar to the discussion about rendezvous process among multiple users in a multihop distributed system (Chap. 10), we assume all users in the network share some common available port, which is impractical. In the future, we will design efficient algorithms to implement the system based on the rendezvous process between every pair of neighboring users.

References

1. Chuang, I., Wu, H.-Y., Lee, K.-R.,& Kuo, Y.-H. (2013). alternate hop-and-wait channel rendezvous method for cognitive radio networks. In *INFOCOM*.
2. Gu, Z., Hua, Q.-S., Wang, Y., & Lau, F. C. M. (2013). Nearly optimal asynchronous blind rendezvous algorithm for cognitive radio networks. In *SECON*.
3. Liu, H., Lin, Z., Chu, X., & Leung, Y.-W. (2012). Jump-stay rendezvous algorithm for cognitive radio networks. *IEEE Transactions on Parallel and Distributed Systems*, *23*(10), 1867–1881.