Deterministic Rendezvous in Restricted Graphs *

Ashley Farrugia[1], Leszek Gąsieniec[1], Łukasz Kuszner[2], and Eduardo Pacheco[3]

[1] Department of Computer Science, University of Liverpool, UK.
[2] Dept. of Algorithms & System Modeling, Gdańsk University of Technology, Poland.
[3] School of Computer Science, Carleton University, Canada.

Abstract. In this paper we consider the problem of synchronous rendezvous in which two anonymous mobile entities (robots) A and B are expected to meet at the same time and point in a graph $G = (V, E)$. Most of the work devoted to rendezvous in graphs assumes that robots have access to the same sets of nodes and edges, where the topology of connections may be initially known or unknown. In our work we assume the movement of robots is restricted by the topological properties of the graph space coupled with the intrinsic characteristics of robots preventing them from visiting certain edges in E.
We consider three rendezvous models reflecting on restricted maneuverability of robots A and B. In *Edge Monotonic Model* each robot $X \in \{A, B\}$ has weight w_X and each edge in E has a weight restriction. Consequently, a robot X is only allowed to traverse edges with weight restrictions greater that w_X. In the remaining two models graph G is unweighted and the restrictions refer to more arbitrary subsets of traversable nodes and edges. In particular, in *Node Inclusive Model* the set of nodes V_X available to robot X, for $X \in \{A, B\}$ satisfies the condition $V_A \subseteq V_B$ or vice versa, and in *Blind Rendezvous Model* the relation between V_A and V_B is arbitrary. In each model we design and analyze efficient rendezvous algorithms. We conclude with a short discussion on the asynchronous case and related open problems.

1 Introduction

In this paper we consider rendezvous problem, a challenge in which two or more mobile entities, called later *robots* have the goal to meet at the same point and time in provided space. This space can be either a network of discrete nodes between which robots can move along existing connections, or a geometric environment in which the movement of robots is restricted by the topological properties of the space. As indicated in [19] *symmetry* plays a key role in determining the feasibility and efficiency of solutions in the rendezvous problem. It is quite often that *anonymous* (indistinguishable) players find themselves in a situation where the tools and advice given to each robot are identical and rendezvous may not

* Research partially supported by the Polish National Science Center grant DEC-2011/02/A/ST6/00201 and by Network Sciences and Technologies initiative at University of Liverpool.

be feasible [5]. In this context, determining even small pieces of information that would help to distinguish between participating robots often prove to be vital in achieving rendezvous.

Rendezvous problems have been studied in a number of different settings. A vast literature includes several exhaustive surveys on the topic and other searching games, see, e.g., [4,5,22]. The work on rendezvous includes both deterministic algorithms surveyed recently in [22] as well as randomized approaches including already classical work in [2,3,8,9]. Another group of algorithms focus on geometric setting including earlier work on the line [9,10] and the plane [6,7] as well as more recent work on fat (with non-zero radius) robots [1,14]. Another interesting group of rendezvous algorithms is designed for infinite (Euclidean) spaces for both synchronized and asynchronous solutions [12,13,16]. An important group of rendezvous algorithms have been considered for graph based environments, see, e.g., [15,18,21]. However, all previous work is devoted to the case when both robots have access to the same part of the network. An interesting version of rendezvous in which robots face different costs associated with traversed edges was considered recently in [17] where the authors consider scenarios with and without communication between participating robots.

Our work refers to the extreme communicationless case of [17] in which the costs imposed on edges are either unit or infinite. We also make reference to *blind rendezvous* considered recently in the context of cognitive radio networks [11,20].

1.1 Model of Computation

We consider rendezvous of *anonymous* (indistinguishable also with respect to the control mechanism) robots in networks modeled by graphs. The network $G = (V, E)$ where the two robots are expected to rendezvous is a simple connected graph in which two nodes $s_A, s_B \in V$ are selected as the starting points for robots A and B respectively. Moreover, for each $X \in \{A, B\}$ we define its *reachability graph* also referred to as *the map* $G_X = (V_X, E_X)$, a subgraph of G in which V_X and E_X are respectively the sets of nodes and edges accessible from s_X. Moreover, agent X is only able to see its own map. Let $k_X = |V_X|$ be *the size* of map G_X and assume w.l.o.g. that $k_A \leq k_B$. While the robots are anonymous, we use extra assumptions with respect to the network nodes (and in some cases edges too). In particular, we assume that all nodes of the input network graph $G = (V = \{v_1, v_2, \ldots, v_n\}, E)$ are ordered, s.t., $v_i < v_{i+1}$ for all $i = 1, 2, \ldots n-1$ and this order is consistent with the order of nodes in G_X, for $X \in \{A, B\}$. In particular, if $V_X = \{v_1^{(X)}, v_2^{(X)}, \ldots, v_{k_X}^{(X)}\}$, $v_a^{(X)} = v_i$, and $v_b^{(X)} = v_j$, where $v_i, v_j \in V$ and $i \leq j$, we also get $v_a^{(X)} \leq v_b^{(X)}$. Finally, let $T(V_X)$ be a rooted tree that spans all nodes in V_X in which the starting point s_X is placed in the root of $T(V_X)$ and the order on children is consistent with the order of nodes in V_X.

The actions of the two robots are synchronized. I.e., the two robots A and B have access to the global clock ticking in discrete *time steps* $0, 1, 2, \ldots$. Our algorithms start with the global clock set to time 0. During a single time step each robot assesses the node in which it resides in (including check for co-

location/rendezvous with the other robot). Then it decides whether to stay at the same node or to move to one of its neighbors via an available (edge) connection. During the traversal between two connected nodes the "eyes" of the robot are closed. Consequently, since the robots cannot meet on edges rendezvous has to take place at some node. The running time of all algorithms is bounded, i.e., the robots stop within the time given to the respective rendezvous algorithms.

We consider three **models of computation** with restrictions on maps given to robots A and B.

1. **Edge Monotonic Model** This model is motivated by the case in which each robot $X \in \{A, B\}$ has weight w_X and each edge in E has weight restriction. This setting imposes an order on edges in $E = \{e_1, e_2, \ldots e_m\}$, in which for any $1 \leq i < j \leq m$ edge e_j tolerates weights non-smaller than e_i. Let i_X be the smallest integer, s.t., e_{i_X} tolerates weight w_X. One can conclude that robot X is only allowed to traverse edges with index $\geq i_X$. Consequently in this model if rendezvous is possible $E_A \subseteq E_B$ and $V_A \subseteq V_B$ (i.e., G_A is a subgraph of G_B), see section 2.1.
2. **Node Inclusive Model** In this model we only assume that $V_A \subseteq V_B$, i.e., the relationship between edges spanning nodes in E_A and E_B remains unspecified.
3. **Blind Rendezvous Model** In this model we only assume that $V_A \cap V_B \neq \emptyset$. Also here the relationship between E_A and E_B is unspecified.

1.2 Our results

In this paper we study synchronized rendezvous in three different restriction models. In section 2.1 we present optimal $O(k_A+k_B)-$time rendezvous algorithm **RV1** in the Edge Monotonic Model. In section 2.2 we present rendezvous algorithm **RV2** that meets two robots in the Node Inclusive Model in almost linear time $O((k_A+k_B)\log(k_A+k_B))$. In the Blind Rendezvous Model, see section 2.3 we show that rendezvous is not feasible. We introduce explicit labels to make rendezvous feasible and present two rendezvous algorithms **RV3** and **RV4** whose superposition allows robots to meet in time $\min\{O((k_A+k_B)^3 \log\log n, O((k_A+k_B)^2 \log n)\}$. We conclude with the final comment and a short discussion on asynchronous models in section 3.

2 Rendezvous Algorithms

In this section we design and analyze several rendezvous algorithms in the considered restriction models.

2.1 Rendezvous in Edge Monotonic Model

Recall that in this model, we adopt the order of edges in $E = \{e_1, e_2, ..., e_m\}$ where $e_i < e_{i+1}$. For any $l \in \{1, \ldots, m\}$, we define a sequence of subgraphs

$G(l) = (V(l), E(l))$, where $E(l) = \{e_l, e_{l+1}, ..., e_m\}$ and $V(l)$ is the set of nodes in V induced by the edges of $E(l)$, and $E(l+1) \subset E(l)$. In this model each robot X is associated with the threshold index $i_X \in \{1, \ldots, m\}$ determining the set of edges $E(i_X)$ traversable by X. In other words, robot X can walk only along edges from $E(i_X)$. We also define a sequence of connected components $G_X(l) = \{V_X(l), E_X(l)\}$, for $l \in \{i_X, \ldots, m\}$, where $V_X(l)$ is the set of nodes reachable from s_X via edges in $E(i_X)$, and $E_X(l) \subseteq E(l)$ is the maximal set of edges spanning nodes in $V_X(l)$. So in this case $V_X = V_X(i_X)$, $E_X = E_X(i_X)$, and $k_X = |V_X(i_X)|$. The following Lemma holds.

Lemma 1. *In Edge Monotonic Model either* $(V_A \subseteq V_B)$ *or* $(V_B \subseteq V_A)$, *or* $V_A \cap V_B = \emptyset$.

Proof. The lemma (statement) would be false if all of the terms $(V_A \subseteq V_B), (V_B \subseteq V_A)$, and $V_A \cap V_B = \emptyset$ were false too. Assume w.l.o.g. that $V_A \cap V_B \neq \emptyset$, where $V_A = V_A(i_A)$ and $V_B = V_B(i_B)$, and $i_A \geq i_B$. Since $i_B \leq i_A$ (edges traversable by A are also traversable by B) and $V_A \cap V_B \neq \emptyset$ (the reachability graphs G_A and G_B coincide) all edges and points in $G_A(i_A)$ are also available to robot B, meaning $V_A \subseteq V_B$.

We define the concept of a *sleeve of graphs* with respect to X denoted by $SL(X)$.

Definition 1. *The sleeve of graphs* $SL(X)$ *with respect to robot* X *is the maximal sequence of connected components* $G_X(i_X)$, $G_X(i_X+1), \ldots, G_X(l^*)$, *in which* $|V_X(l+1)| > |V_X(l)|/2$, *for all* $i_X \leq l^* < m$. *A subsequence* $G_X(i_X+j)$, $G_X(i_X+j+1), \ldots, G_X(l^*)$, *for any* $j \in \{0, 1, \ldots, l^* - i_X\}$, *is called a tail of* $SL(X)$ *and the smallest (in the adopted order) node* $v^* \in V_X(l^*)$ *is called the target in* $SL(X)$.

In what follows we present a pseudo-code of the proposed rendezvous algorithm in the monotonic model. If at any time step the two robots A and B meet, the rendezvous is achieved and the two robots *halt*.

Algorithm RV1$(X \in \{A, B\})$
Step 1 Walk from s_X to the target node v^* in $SL(X)$
Step 2 Wait in v^* until conclusion of time step $2k_X$;
Step 3 Walk along the Euler tour of $T(V_X)$ and *Halt*.

Theorem 1. *If rendezvous is feasible, Algorithm* **RV1** *admits meeting in optimal time* $O(k_A + k_B)$.

Proof. Recall that $k_A \leq k_B$. According to Lemma 1 if rendezvous is feasible, i.e., $V_A \cap V_B \neq \emptyset$ we conclude that $V_A \subseteq V_B$. We consider two complementary cases:

Case 1 $[2k_A > k_B]$ Since $2k_A > k_B$ according to Definition 1 sleeve $SL(A)$ is a tail of $SL(B)$ and the two sleeves share the same target v^*. The robots A and B are initially placed in their own sleeves at distance at most $k_B < 2k_A$ from the joint target v^*. This admits rendezvous in **Step 1** in time at most k_B.

Case 2 $[2k_A \leq k_B]$ In this case, robot A halts at the latest at time step $4k_A$ on the conclusion of **Step 3**, i.e., after $2k_A$ time steps devoted to **Step 1** and **Step 2**, followed by additional $2k_A - 2$ time steps devoted to the Euler tour traversal in $T(V_A)$) in **Step 3**. Note, however, that robot B enters **Step 3** in time step $2k_B + 1 > 4k_A$, when robot A is already immobilized. Since during **Step 3** robot B visits all nodes in V_B (that include also all nodes in V_A) rendezvous must occur. □

2.2 Rendezvous in Node Inclusion Model

Recall that in this model we assume that all nodes are ordered and $k_A \leq k_B$, where $V_A \subseteq V_B$. In this model we have no order on edges and in turn the concept of sleeve of graphs cannot be applied here. Instead, one can focus on a different mechanism that will allow to distinguish between two robots and with this in mind we focus on the values of k_A and k_B. Note that if $k_A = k_B$ due to the inclusion assumption we also have $V_A = V_B$. In this case, since orders of nodes in V_A and V_B are consistent the robots can meet at the smallest (in order) node v^* in V_A and V_B that must coincide. Otherwise, the values of k_A and k_B differ and each robot X, for $X \in \{A, B\}$ can adopt k_X as its unique identifier. Furthermore, apart from unique identities there needs to be a synchronization mechanism (sizes of k_A and k_B can be dramatically different) that will allow robots to coordinate their individual moves. The rendezvous mechanism for any robot X is based on synchronized awaiting of the first stage that is long enough to accommodate actions reflecting the size k_X. In particular, we identify the power of two j_X, s.t., $2^{j_X - 1} \leq k_X < 2^{j_X}$ that provide a constant estimation and the upper bound on the size of k_X. The algorithm applied to robot X operates in stages $j = 1, 2, 3, ..., j_X$, where during stages 1 through $j_X - 1$ the robot remains immobilized and in the last stage j_X it actively participates (visiting all nodes in V_X) in the rendezvous process. Note that if $j_A < j_B$ (and $V_A \subset V_B$) in stage j_B, when robot A is already immobilized, B by visiting all nodes in V_B (that is a superset of V_A) must conclude rendezvous. In the complementary case, i.e., when the estimates j_A and j_B are the same we use binary expansions $b_A[0, \ldots, j_A]$ and $b_B[0, \ldots, j_B]$ (where positions j_A, j_B are the most significant) of k_A and k_B respectively to differentiate between the robots.

Lemma 2. *If $j_A = j_B$ and $k_A < k_B$ there exists $i \in \{0, 1, \ldots, j_A = j_B\}$, s.t., $b_A[i] = 0$ and $b_B[i] = 1$.*

Proof. If for all $i \in \{0, 1, \ldots, j_A = j_B\}$, $(b_A[i] = 0) => (b_B[i] = 0)$ would imply $k_A \geq k_B$.

A pseudo-code of the rendezvous algorithm **RV2** in the inclusion model follows. If at any time step the two robots A and B meet, the rendezvous is achieved and the two robots *halt*.

```
1. Algorithm RV2(X ∈ {A, B})
2. Step 1 Compute j_X and b_X[0, ..., j_B].
3. Step 2 for j = 1, 2, ..., j_X do
4.         if (j = j_X) {active stage}
5.             use 2^{j_X} time steps to walk to and wait in v*. {smallest node}
6.             (i) for i = 0, 1, ..., j_X do
7.                 if (b_X[i] = 1)
8.                     (a) use 2·2^{j_X} time steps to visit Euler tour in T(V_X)
9.                         and return to v*
10.                else (b) wait 2 · 2^{j_X} time steps in v*
11.        else (ii) wait 2^j · (2j + 1) time steps where you are.
```

We prove the following theorem.

Theorem 2. *If rendezvous is feasible Algorithm* **RV2** *admits meeting in time* $O((k_A + k_B)\log(k_A + k_B))$.

Proof. The rendezvous algorithm runs in j_X stages controlled by the loop **for** in line 3. There are two cases. In the first case, where $j_A < j_B$, when robot B is in the active stage robot A is already immobilized, and B meets A during traversal of the Euler tour in $T(V_B)$, see line 8 of the code. Otherwise, when $j_A = j_B$ we have two subcases. In the first subcase when $k_A = k_B$ the robots meet in the shared smallest node v^*, see line 5. In the second subcase, where $k_A < k_B$, according to Lemma 2 there exists i, s.t., $b_A[i] = 0$ and $b_B[i] = 1$ when robot B visits the Euler tour in $T(V_B)$ and robot A is immobilized. Thus this subcase admits rendezvous too.

With respect to the time complexity we first observe that the execution time of algorithm **RV2** is bounded and it depends on the parameter j_X. The time complexity of each stage $j = 1, ..., j_X$ is bounded by $3 \cdot 2^j$ resulting in the total complexity $\sum_{j=1}^{j_X}(2^j \cdot (2j+1)) \leq \sum_{j=0}^{j_X}(2^j \cdot (2j_X + 1))$. This is equivalent to $(2j_X + 1)\sum_{j=1}^{j_X}(2^j) = (2j_X + 1) \cdot (2^{j_X+1} - 1) = O(k_X \cdot \log k_X)$, since $2^{j_X} - 1 \leq k_X < 2^{j_X}$. This admits the time complexity $O((k_A + k_B)\log(k_A + k_B))$. □

2.3 Blind Rendezvous Model

In this section we consider rendezvous where the relationship between the maps of robots is more arbitrary. We first show that without any additional information, even if $V_A \cap V_B \neq \emptyset$, rendezvous cannot be reached.

Theorem 3. *Blind rendezvous is not feasible.*

Proof. Assume that for any $X \in \{A, B\}$ we have $V_X = \{v_1^{(X)}, v_2^{(X)}\}$ and $E_X = \{(v_1^{(X)}, v_2^{(X)})\}$, where node $v_2^{(A)}$ coincides with $v_1^{(B)}$ and where for each robot X the starting node s_X coincides with v_1^X on its own map. It is enough to observe that without any additional information the symmetry tie cannot be broken. And indeed, since the robots are anonymous (indistinguishable) whenever robot A visits $v_2^{(A)}$ robot B visits $v_2^{(B)}$, i.e., the two robots never visit the shared node simultaneously. □

One can adopt a natural assumption that the nodes in V_X apart from being ordered they also have *explicit labels*. In consequence, if a node $v_a^{(A)} \in V_A$ coincides with $v_b^{(B)} \in V_B$ they both possess the same explicit label. We assume that the labels are drawn from the set of integers $\{1, 2, \ldots, n\}$, and we use notation $b_i^{(X)}$ (or $b_i^{(X)}[0..\log n]$) to denote the binary expansion of the explicit label of $v_i^{(X)} \in V_X$.

We also assume that n is known to both robots. Otherwise no rendezvous algorithm would stop and report infeasibility of rendezvous when $V_A \cap V_B = \emptyset$, as robots are not aware of sizes of each others maps.

Before we present two rendezvous algorithms we show that the symmetry tie problem, see Theorem 3, can be overcome if the explicit labels are available. W.l.o.g. we also assume that the order of labels is consistent with the order imposed on nodes on each map. If this is not the case a new (consistent) order for nodes in V_A and V_B can be computed on the basis of explicit labels (we only care about nodes in $V_A \cap V_B$). The following result has been shown in [11]. Our proof, however, is much simpler and based on binary representation of explicit labels.

Lemma 3. *Assume that the map of any robot $X \in \{A, B\}$ is an ordered pair of nodes $(v_1^{(X)}, v_2^{(X)})$ connected by a symmetric edge, where nodes $v_2^{(A)}$ and $v_1^{(B)}$ physically coincide and nodes $v_1^{(A)}$ and $v_2^{(B)}$ don't. In such network one can break the symmetry tie to reach rendezvous in time $O(\log \log n)$.*

Proof. We first observe that according to the imposed order $b_1^{(A)} < b_2^{(A)} = b_1^{(B)} < b_2^{(B)}$. The case where $s_A = v_2^{(A)}$ and $s_B = v_1^{(B)}$ is trivial and another case where $s_A = v_1^{(A)}$ and $s_B = v_2^{(B)}$ can be easily resolved by an algorithm that alternates between the two nodes (e.g., in every other time step). Let $1 \leq r_A \leq \log n$ be the largest integer position, s.t., $b_1^{(A)}[r_A] \neq b_2^{(A)}[r_A]$. Since $b_1^{(A)} < b_2^{(A)}$ one can conclude that $b_1^{(A)}[r_A] = 0$ and $b_2^{(A)}[r_A] = 1$. Similarly let $1 \leq r_B \leq \log n$ be the largest integer position, s.t., $b_1^{(B)}[r_B] \neq b_2^{(B)}[r_B]$. Since $b_1^{(B)} < b_2^{(B)}$ one can also conclude that $b_1^{(B)}[r_B] = 0$ and $b_2^{(B)}[r_B] = 1$. We observe that since $b_2^{(A)} = b_1^{(B)}$ one can conclude that $r_A \neq r_B$ as the respective positions cannot contain 0 and 1 at the same time. Moreover binary expansions br_A and br_B of r_A and r_B respectively are limited to $\log \log n + 1$ bits.

We consider a symmetry breaking algorithm in which in time step i each robot $X \in \{A, B\}$ moves to the other node only if $i = 2 \cdot l$ (i is even) or if

$i = 2 \cdot l - 1$ (i is odd) and $br_X[l] = 1$, for $l = 1, \ldots, \log\log n + 1$. Note that since $r_A \neq r_B$ for some $1 \leq l \leq \log\log n + 1$ we must have $br_A[l] \neq br_B[l]$ and if until now the rendezvous is not reached (all previous moves were symmetric and in the last odd time step, when the symmetry was broken robots occupy different nodes) in the next even step the rendezvous is accomplished.

Corollary 1. *Note that the lemma above applies to pairs of nodes at distance 1. In a more general case, where the distance between nodes in the pair is $d \geq 1$, the symmetry breaking rendezvous takes time $O(d \log\log n)$.*

In the remaining part of this section we present two rendezvous algorithms followed by their superposition. The first algorithm **RV3** has the time complexity $O((k_A + k_B)^3 \log\log n)$ and its idea is based on the blind rendezvous algorithm from [11] where the problem was studied in complete graphs. The second algorithm **RV4** has the time complexity $O((k_A + k_B)^2 \log n)$ making it superior to **RV3** when $k_A + k_B > \frac{\log n}{\log\log n} = \tau$, where τ is the threshold value. This rendezvous algorithm resembles algorithm **RV2** however here the symmetry tie is broken with the help of explicit labels.

Blind rendezvous in time $O((k_A + k_B)^3 \log\log n)$ Similarly to its predecessor **RV2** also the first blind rendezvous algorithms **RV3** operates in stages accommodating geometrically increasing estimates on sizes of the input maps. This is needed as the size of the map of one robot is not known to the other. The robot starts using active stages only when the current estimate is large enough to accommodate its map. The rendezvous process terminates in time $O((k_A + k_B)^3 \log\log n)$ if the maps of both agents are smaller than the threshold value τ. Otherwise, algorithm **RV3** is followed by execution of algorithm **RV4**. If at any time step the two robots A and B meet, the rendezvous is achieved and the two robots *halt*.

1. **Algorithm RV3**($X \in \{A, B\}$)
2. **Step 1** Compute j_X and the threshold $\tau = \frac{\log n}{\log\log n}$.
3. **Step 2 for** $j = 1, 2, \ldots, \lceil \log\tau \rceil$ **do**
4. **if** $(j \geq j_X)$ {*active stage*}
5. (i) **for** all pairs $(a, b) \in \{1, \ldots, 2^j\} \times \{1, \ldots, 2^j\}$
6. ordered lexicographically **do**
7. **if** (either of $v_a^{(X)}, v_b^{(X)}$ exists)
8. (a) **run** blind rendezvous in pair $(v_a^{(X)}, v_b^{(X)})$ **or**
9. **wait** appropriate $O(2^j \log\log n)$ time steps
10. in the only existing node;
11. **else** (b) **wait** appropriate $O(2^j \log\log n)$ time steps
12. where you currently are;
13. **else** (ii) **wait** suitable $O(2^{3j} \cdot \log\log n)$ time steps where you are.

Theorem 4. *If $k_A + k_B < \tau = \frac{\log n}{\log \log n}$ and rendezvous is feasible, algorithm* **RV3** *admits rendezvous in time $O((k_A + k_B)^3 \log \log n)$.*

Proof. The rendezvous algorithm runs in $\lceil \log \tau \rceil$ stages controlled by the loop **for** in line 3. Robot X starts executing active stages as soon as the stages can accommodate the size of X's map. If the size of the map is too big, robot X awaits execution of the second rendezvous algorithms **RV4**, see line 9. During an active round all pairs (a, b) from the Cartesian product $\{1, \ldots, 2^j\} \times \{1, \ldots, 2^j\}$ are drawn in the lexicographic order. Only certain pairs are valid, i.e., when either of $v_a^{(X)}$ and $v_b^{(X)}$ exists. In each valid pair if only one node exists robot X remains in this node for the duration of the symmetry breaking procedure. Otherwise, if both nodes exist the breaking symmetry procedure is executed with the distance between the two nodes bounded by 2^j.

If rendezvous is feasible we must have nodes $v_a^{(A)} \in V_A$ and $v_b^{(B)} \in V_B$ that coincide by sharing the same label. If the pair $(v_a^{(X)}, v_b^{(X)})$ exists in both maps thanks to the symmetry breaking procedure eventually robot A will visit $v_a^{(A)}$ at the same time when entity B visits $v_b^{(B)}$ and the rendezvous is reached. If only one element of the pair $(v_a^{(X)}, v_b^{(X)})$ exists, i.e., either $v_a^{(A)}$ for A or $v_b^{(B)}$ for B the respective robot is asked to wait in the existing node of the pair resulting in rendezvous too. Otherwise the robots await the next pair from the Cartesian Product without movement for the period corresponding to execution of the symmetry breaking procedure. Thus the actions performed by robots A and B remain fully synchronized.

With respect to the time complexity we first observe that the execution time of algorithm **RV2** is bounded and it depends on the parameter j_X. The time complexity of each stage $j = 1, ..., j_X$ is bounded by $3 \cdot 2^j$ resulting in the total complexity $\sum_{j=1}^{j_X}(2^j \cdot (2j+1)) \leq \sum_{j=0}^{j_X}(2^j \cdot (2j_X + 1))$. This is equivalent to $(2j_X + 1)\sum_{j=1}^{j_X}(2^j) = (2j_X + 1) \cdot (2^{j_X+1} - 1) = O(k_X \cdot \log k_X)$, since $2^{j_X - 1} \leq k_X < 2^{j_X}$. This admits the time complexity $O((k_A + k_B)\log(k_A + k_B))$. □

Blind rendezvous in time $O((k_A + k_B)^2 \log n)$ We start with the proof of the following fact.

Lemma 4. *One can impose a periodic order $\pi(X)$ on nodes of a spanning tree $T(V_X)$, s.t., the walking distance (the number of edges to be visited) between two consecutive nodes in order $\pi(X)$ is at most* 3.

Proof. We say that the nodes located at an even distance from the root s_X are on an even level and all the remaining nodes are on an odd level. The ordering of nodes π is created according to the following principle. Starting from the root s_X we visit all nodes in $T(V_X)$ using depth-first search algorithm. The root gets label 0. When we arrive (from the parent) to an even level the currently visited node gets the next available label. In other words at even levels we use *pre-order numbering principle*. And when we arrive (from the last child) to an odd level

the currently visited node gets the next available label. I.e., at odd levels we follow *post-order numbering principle*

We need to show that the labeling (ordering) procedure proposed above generates at least one new label in three consecutive steps. And indeed, if we follow the route determined by the depth-first search algorithm and we visit for the first time a node v at an even level (when the new label is generated): (case 1) if the first child of v has a child w then w (which is at distance 2 from v) gets the new label; (case 2) if the first child of v is a leaf this child (which is at distance 1 from v) gets the new label; (case 3) if the node v is a leaf but not the last child of its parent the next label goes to the (next) sibling of v (which is at distance 2); and (case 4) if v is the last child the next label goes to its parent (which is at distance 1).

Similarly, if v is visited for the last time on an odd level it gets a new label. Now (case 5) if v is the last child and its parent w is not the last child the next sibling of the parent (which is at distance 3 from v) gets the new label; (case 6) if v is the last child and its parent w is also the list child then the parent of w (at distance 2 from v) gets the new label; (case 7) and if v is the last child and its parent is the root, the periodic order is established (and the next label is at distance 1). In the remaining cases when v is not the last child (case 8) if its next sibling (at distance 2) is a leaf it gets the new label; and (case 9) if the next sibling of v has children the next label go to the first child (at distance 3 from v) of this sibling. □

1. **Algorithm RV4(**$X \in \{A, B\}$**)**
2. **Step 1** Determine j_X, the threshold $\tau = \frac{\log n}{\log \log n}$, and the label $b_i^{(X)}$ of s_X;
3. **Step 2 for** $j = \lceil \log \tau \rceil, 2, \ldots, \log n$ **do**
4. **if** $(j \geq j_X)$ {*active stage*}
5. (**walk** to and **wait** in s_X) in 2^j time steps;
6. **for** $l = 0, 1, \ldots, \log n$ **do** {test all bits}
7. **if** $(b_i^{(X)}[l] = 1)$ {walk all the time}
8. **for** $2^{2j} \times 3$ time steps **do**
9. **walk** to the next node in order $\pi(X)$;
10. **else repeat** 2^j times {walk and wait for another}
11. (**walk** to the next node in order $\pi(X)$
12. and **wait** there) in $2^j \times 3$ time steps;
13. **else wait** appropriate $O(2^{2j} \cdot \log n)$ time steps where you are.

The last rendezvous algorithm **RV4** operates on the following principle. At the start of each active stage robot X returns (if moved before) to the starting point s_X. If the two starting points in V_A and in V_B coincide rendezvous is accomplished. Otherwise the algorithm controls further movement of robots, s.t., during long enough ($\geq 2^j \times 3$ time steps) interval of an active stage j one of the robots, say w.l.o.g. A, visits all nodes in V_A in the periodic order $\pi(A)$ with frequency of one visit per three time steps. While the other robot B visits

consecutive nodes with frequency of $2^j \times 3$ time steps. So when eventually robot B resides in the node that belongs to $V_A \cap V_B$ there is enough time for robot A to arrive in this node before B moves away. If at any time step the two robots A and B meet, the rendezvous is achieved and the two robots *halt*.

Theorem 5. *If $k_A + k_B \geq \tau = \frac{\log n}{\log \log n}$ and rendezvous is feasible, algorithm* **RV4** *admits rendezvous in time $O((k_A + k_B)^2 \log n)$.*

Proof. Lets consider the first stage that is active for both robots A and B, i.e., when $j = j_B$. Note that line 13 of the pseudo-code accommodates for the waiting time needed for two robots to stay synchronized prior to this stage. In this active stage loop for in line 6 compares consecutive bits of labels $b_i^{(A)}$ adopted by A and $b_{i'}^{(B)}$ adopted by B. There must be at least one position l on which the two labels differ. In consequence, there is a spell of $2^{2j} \times 3$ time steps during which one of the robots, say w.l.o.g. A with the bit $b_i^{(A)}[l] = 1$, visits periodically all nodes in V_A with frequency of 3 time steps per node. During the same times spell the other robot B with the bit $b_{i'}^{(B)}[l] = 0$ waits long ($\geq 2^j \times 3$ time steps) periods of time in every node of V_B. So when eventually robot B visits the node that belongs to $V_A \cap V_B$ the other robot A has enough time to arrive in this node before B moves on.

The time complexity of this first active stage is $O(2^{2j_B} \cdot \log n) = O(k_B^2 \log n)$. Since the duration of stages grows exponentially we conclude that the total time complexity is also $O(k_B^2 \log n) = O((k_A + k_B)^2 \log n)$. □

Corollary 2. *In the Blind Rendezvous Model two robots can rendezvous in time* $\min\{O((k_A + k_B)^3 \log \log n, O((k_A + k_B)^2 \log n)\}$.

Proof. The result follows directly from the superposition of **RV3** and **RV4**.

3 Conclusion

In this paper we studied deterministic synchronized rendezvous of two robots in the network environment with restrictions imposed on network edges. The restrictions prevent robots from visiting certain parts of the network. We considered three restriction models and we provided four efficient solutions in section 2. One of the open problems is to establish the exact complexity of rendezvous in considered models and to answer whether the use of randomisation helps. One can also consider models in which maps are not known to the robots. Another interesting question refers to better understanding (including time complexity) of gathering more than two robots. In this setting while robots could meet in pairs, one mutually accessible location for gathering may not be available. It would be also good to understand the case when robots are asked to meet asynchronously. Initial studies indicate that in Edge Monotonic Model there exist rendezvous algorithms that allows robots to meet after adopting trajectories of length polynomial in $k_A + k_B$. In Node Inclusive Model the lengths of respective trajectories become exponential. Finally in Blind Rendezvous Model rendezvous is not feasible even if explicit labels are provided.

References

1. C. Agathangelou, C. Georgiou, and M. Mavronicolas, A distributed algorithm for gathering many fat mobile robots in the plane, In Proc. PODC 2013, pp. 250-259.
2. S. Alpern, The rendezvous search problem, SIAM J. Control and Optimization (33), pp. 673–683, 1995.
3. S. Alpern, Rendezvous search on labeled networks, Naval Reaserch Logistics (49), pp. 256–274, 2002.
4. S. Alpern, R. Fokkink, L. Gąsieniec, R. Lindelauf, and V.S. Subrahmanian, Search Theory, A Game Theoretic Perspective, Springer 2013.
5. S. Alpern and S. Gal, The Theory of Search Games and Rendezvous, Kluwer Academic Publisher, 2002.
6. E. Anderson and S. Fekete, Asymmetric rendezvous on the plane, In Proc. Symp. on Computational Geometry 1998, pp. 365-373.
7. E. Anderson and S. Fekete, Two-dimensional rendezvous search, Operations Research (49), pp. 107-118, 2001.
8. E. Anderson and R. Weber, The rendezvous problem on discrete locations, Journal of Applied Probability (28), pp. 839–851, 1990.
9. V. Baston and S. Gal, Rendezvous on the line when the players' initial distance is given by an unknown probability distribution, SIAM J. Control and Optimization (36), pp. 1880–1889, 1998.
10. V. Baston and S. Gal, Rendezvous search when marks are left at the starting points. Naval Reaserch Logistics (48), pp. 722–731, 2001.
11. S. Chen, A. Russell, A. Samanta, and R. Sundaram, Deterministic Blind Rendezvous in Cognitive Radio Networks, In Proc. ICDCS 2014.
12. A. Collins, J. Czyzowicz, L. Gąsieniec, and A. Labourel, Tell Me Where I Am So I Can Meet You Sooner, In Proc. ICALP 2010, pp. 502-514.
13. A. Collins, J. Czyzowicz, L. Gąsieniec, A. Kosowski, R.A. Martin, Synchronous Rendezvous for Location-Aware Agents, In Proc. DISC 2011, pp. 447-459.
14. J. Czyzowicz, L. Gąsieniec, and A. Pelc, Gathering few fat mobile robots in the plane, Theoretical Computer Science, (410:6âĂŞ7), pp. 481–499, 2009.
15. J. Czyzowicz, A. Kosowski, and A. Pelc, How to meet when you forget: Log-space rendezvous in arbitrary graphs, Distributed Computing (25), pp. 165-178, 2012.
16. J. Czyzowicz, A. Labourel, and A. Pelc, How to meet asynchronously (almost) everywhere, ACM Transactions on Algorithms (8), article 37, 2012.
17. D. Dereniowski, R. Klasing, A. Kosowski, Ł. Kuszner, Rendezvous of Heterogeneous Mobile Agents in Edge-Weighted Networks, In Proc. SIROCCO 2014, pp. 311-326.
18. D. Kowalski, A. Malinowski, How to meet in anonymous network, Theoretical Computer Science (399), pp. 141-156, 2008.
19. E. Kranakis, D. Krizanc, S. Rajsbaum, Mobile Agent Rendezvous: A Survey, In Proc. SIROCCO 2006, pp. 1–9.
20. Z. Lin, H. Liu, X. Chu, and Y-W. Leung, Jump-stay based channel-hopping algorithm with guaranteed rendezvous for cognitive radio networks, In Proc. INFOCOM 2011, pp. 2444–2452.
21. A. Miller and A. Pelc, Time Versus Cost Tradeoffs for Deterministic Rendezvous in Networks In Proc. PODC 2014, pp. 282–290.
22. A. Pelc, Deterministic rendezvous in networks: A comprehensive survey, Networks (59), pp. 331–347, 2012.
23. T. Schelling, The Strategy of Conflict, Harvard Univ. Press, Cambridge, 1960.