Tell Me Where I Am So I Can Meet You Sooner
(Asynchronous Rendezvous with Location Information)

Andrew Collins[1], Jurek Czyzowicz[2], Leszek Gąsieniec[1], and Arnaud Labourel[3]

[1] University of Liverpool
[2] Université du Québec
[3] Université de Bordeaux

Abstract. In this paper we study efficient rendezvous of two mobile agents moving asynchronously in the Euclidean 2D-space. Each agent has limited visibility, permitting it to see its neighborhood at unit range from its current location. Moreover, it is assumed that each agent knows its own initial position in the plane given by its coordinates. The agents, however, are not aware of each others position. The agents possess coherent compasses and the same unit of length, which permit them to consider their current positions within the same system of coordinates. The cost of the rendezvous algorithm is the sum of lengths of the trajectories of both agents. This cost is taken as the maximum over all possible asynchronous movements of the agents, controlled by the adversary.

We propose an algorithm that allows the agents to meet in a local neighborhood of diameter $O(d)$, where d is the original distance between the agents. This seems rather surprising since each agent is unaware of the possible location of the other agent. In fact, the cost of our algorithm is $O(d^{2+\varepsilon})$, for any constant $\varepsilon > 0$. This is almost optimal, since a lower bound of $\Omega(d^2)$ is straightforward. The only up to date paper [12] on asynchronous rendezvous of bounded-visibility agents in the plane provides the feasibility proof for rendezvous, proposing a solution exponential in the distance d and in the labels of the agents. In contrast, we show here that, when the identity of the agent is based solely on its original location, an almost optimal solution is possible.

An integral component of our solution is the construction of a novel type of non-simple *space-filling curves* that preserve locality. An infinite curve of this type visits specific grid points in the plane and provides a route that can be adopted by the mobile agents in search for one another. This new concept may also appear counter-intuitive in view of the result from [22] stating that for any simple space-filling curve, there always exists a pair of close points in the plane, such that their distance along the space-filling curve is arbitrarily large.

1 Introduction

1.1 The Problem and the Model

A pair of identical mobile agents is located at two points in the plane. Each agent has limited visibility, permitting it to see its neighborhood at unit range from its current location. We assume that each agent knows its own *initial position* in the

S. Abramsky et al. (Eds.): ICALP 2010, Part II, LNCS 6199, pp. 502–514, 2010.

plane given by its coordinates, i.e., it is *location aware*. However, the agents do not know each others position. We also assume that the agents possess coherent compasses and the same unit of length, which permit them to consider their current positions within the same system of coordinates. Therefore each agent may consider its initial location as its unique ID.

The *route* of each agent is a sequence of segments which are subsequently traversed during its movement. The entire route of the agent depends uniquely on its initial position. The actual *walk* of each agent along every segment is *asynchronous*, i.e., it is controlled by an adversary. The agents meet if they eventually get within the visibility range of each other, i.e., at some point in time the distance between their current positions will not be greater than one.

We now define more precisely the power of the adversary. The adversary initially places both agents at any two points in the plane. Given its initial location a_0, the route chosen by the agent is a sequence of segments $(e_1, e_2, \ldots)$, such that in stage i the agent traverses segment $e_i = [a_{i-1}, a_i]$, starting at a_{i-1} and ending at a_i. Stages are repeated indefinitely (until rendezvous). We assume that each agent may start its walk at any time, but both agents are placed by the adversary at their respective initial positions at the same moment, and since that time any moving agent may find the other agent, even if the other agent didn't start its walk yet.

We describe the walk f of an agent on its route, similarly as in [12]: let $R = (e_1, e_2, \ldots)$ be the route of an agent. Let $(t_1, t_2, \ldots)$, where $t_1 = 0$, be an increasing sequence of reals, chosen by the adversary, that represent points in time. Let $f_i : [t_i, t_{i+1}] \to [a_i, a_{i+1}]$ be any continuous function, chosen by the adversary, such that $f_i(t_i) = a_i$ and $f_i(t_{i+1}) = a_{i+1}$. For any $t \in [t_i, t_{i+1}]$, we define $f(t) = f_i(t)$. The interpretation of the walk f is as follows: at time t the agent is at the point $f(t)$ of its route. The adversary may arbitrarily vary the speed of the agent, as long as the walk of the agent in each segment is continuous, eventually bringing the agent from the starting endpoint to the other endpoint of the corresponding segment of its route[1].

Agents with routes R_1 and R_2 and with walks $f^{(1)}$ and $f^{(2)}$ meet at time t, if points $f^{(1)}(t)$ and $f^{(2)}(t)$ are identical. A rendezvous is guaranteed for routes R_1 and R_2, if the agents using these routes meet at some time t, regardless of the walks chosen by the adversary. The cost of the rendezvous algorithm is measured by the sum of the lengths of the trajectories of both agents from their starting locations until the time t of the rendezvous. Since the actual portions of these trajectories may vary depending on the adversary, we consider the maximum of this sum, i.e., the worst-case over all possible walks chosen for both agents by the adversary. In this paper we are looking for the rendezvous algorithm of the smallest possible cost with respect to the original distance d between the agents.

[1] This defines a very powerful adversary. Notice that the presented algorithm is valid even with this powerful adversary, and the lower bound argument works also for a weaker adversary that can only speed up or slow down the robot, without moving it back (corresponding to a walk function f which must be additionally monotonous). Hence our results also hold in this (perhaps more realistic) model.

1.2 Related Work

The rendezvous problem is often presented as a search game involving two players (or agents) which cooperate in order to meet as quickly as possible. An extensive presentation of the large literature on rendezvous can be found in the excellent book [2]. The majority of research concern deterministic algorithms, which is also the model adopted in this paper. Most papers on rendezvous assume either the graph environment, e.g., [2,3] or the geometric environment. In the geometric scenario the agents move in some geometric terrain, e.g. line [6,7,20], plane [4,5,12], or unknown bounded environment [11,21].

One of the fundamental issues in the deterministic rendezvous algorithms is the problem of *symmetry breaking*, since identical agents starting at some symmetric positions may never meet, indefinitely performing symmetric moves. One possible way to break symmetry is to have agents identified with different *labels*. For the case of agents which are unlabeled, i.e., *anonymous*, [19] studied rendezvous in trees, assuming that agents have bounded memory. However, trees are a special case in which rendezvous is often feasible, supposing that neither nodes nor agents are labeled or marked. The rendezvous in a graph is feasible, see [13], if and only if the starting positions of the anonymous agents are asymmetric, i.e., the views of the graph from the initial positions of the agents are distinguishable, cf. [30]. In [31] the problem of gathering many agents with unique labels was studied. In [15,23] deterministic rendezvous in graphs with labeled agents was considered. One usual approach used for labeled agents consists in finding a (usually non simple) cycle in the graph, computable by an agent placed at any starting vertex, and an integer bijection f on the set of labels. The agent A goes $f(A)$ times around such cycle and different agents are forced to meet (see e.g. [11,13,14,23]). In [13] it was proved that the log-space memory is sufficient in order to decide whether a given instance of the rendezvous problem is feasible for any graph and any initial positions of the agents.

However, in most of the papers above, the *synchronous* setting was assumed. In the *asynchronous* setting it is assumed that the timing of each action may be arbitrarily slowed down by an adversary. The efficiency of the asynchronous algorithms is determined by the worst-case possible behavior of the adversary. Asynchronous gathering in geometric environments has been studied, e.g., in [10,18] in different models than ours: anonymous agents are oblivious (they can not remember past events), but they are assumed to have at least partial visibility of the scene. The first paper to consider deterministic asynchronous rendezvous in graphs was [14], where the complexity of rendezvous in simple classes of graphs, such as rings and infinite lines, was studied for labeled agents. In [12] the feasibility of asynchronous rendezvous for labeled agents was considered both for graphs and the 2D-space. It was proved that rendezvous is feasible in any connected (even countably infinite) graph. For this purpose, an enumeration of all quadruples containing possible pairs of agent labels and their starting positions is considered. According to this enumeration, the routes of the agents involved in each quadruple is extended in such a way that their meeting is always ensured. If the graph is unknown, the enumeration is constructed while the graph

is explored. The cost of the rendezvous algorithm is exponential in the original distance between the agents. On the other hand, asynchronous rendezvous is unfeasible for agents starting at arbitrary positions in the plane, unless the agents have an $\epsilon > 0$ visibility range, see [12]. The assumption that the agents operating in the geometric environment has a bounded, non-zero visibility range is very natural (cf. [4,5,12,21]).

The assumption that the distributed mobile entities know their initial location in the geometric environment was considered in the past, e.g., in the context of geometric routing, see, e.g., [1,8,24,25], where it is typically assumed that the source node knows the position of the destination as well as its own position, or broadcasting [16,17], where the position awareness of only the broadcasting node is admitted. Such assumption, partly fueled by the expansion of the Global Positioning System (GPS), is sometimes called *location awareness* of agents or nodes of the network, and it often leads to better bounds of the proposed solutions. A technique consisting in the construction of the partition of the plane into bounded diameter cells is usually used here.

An interesting context of this work is its relationship to *space-filling curves* extensively studied at various contexts in the literature, see, e.g., [9,22,26,29]. One of the most important attributes of space-filling curves is sometimes called the *preservation of locality*, i.e., that if two points are close in the 2D-space they are also closely located on the space-filling curve. In this context, however, Gotsman and Lindenbaum pointed out in [22] that space-filling curves fail in preserving the locality in the worst case. They show that, for any space-filling curve, there always exist some close points in the 2D-space that are arbitrarily far apart on the space-filling curve. In this paper we show that such deficiency of space-filling curves comes from a very strong assumption that curves visit each point in a discrete 2D-space exactly once. We propose an alternative to space-filling curves that preserves locality also in the worst case. Namely, we introduce *space-covering sequences* that traverse points in a discrete 2D-space multiple times. We show that, for any $\varepsilon > 0$, there exists a space-covering sequence, s.t., for any two points located at distance d in the 2D-space there are well defined and efficiently computable instances of these two points in the sequence at distance $O(d^{2+\varepsilon})$ apart.

2 Efficient Construction of Space-Covering Sequences

The trajectories constructed in the paper follow grid lines except for their initial segments by which each agent, starting from its *arbitrary* initial position, reaches its closest grid point. It is possible that the first agent traverses an arbitrarily long portion of its route, while the adversary holds the other agent on its initial segment being not visible from any grid line. To prevent this we assume, w.l.o.g., that the 2D-space is rescaled so that the value of $\frac{\sqrt{2}}{2}$ corresponds to the unit length of the 2D-space. This way the considered grid is fine enough, and the agent visiting an integer grid point v will see another agent, situated in any interior point of a unit grid square with vertex v.

The fundamental concept used in the design of the rendezvous algorithm is the *space-covering sequence* on which both robots are walking until rendezvous. The space-covering sequence is infinite in both directions. It is formed of short segments with the endpoints located at the integer grid points. Every point of the integer grid is visited by the space-covering sequence infinitely many times. Each agent starting from its arbitrary initial position in the plane walks first to the closest grid point and then it starts making a special zigzag movement on the sequence, each time covering more and more distance. The actual points at which the agent changes the direction of its movement are determined by the coordinates of the initial position of the agent and the purpose of our algorithm is to determine them so that an efficient rendezvous will always be possible.

The construction of the space-covering sequence utilizes a hierarchy of infinite grids of squares of increasing sizes This hierarchy is an amalgamate $\mathcal{H}_{QC}$ of two complementary hierarchies of square partitions: the *quad-tree partition hierarchy* $\mathcal{H}_Q$ and the *central square partition hierarchy* $\mathcal{H}_C$. For the clarity of presentation we first demonstrate an argument leading to the rendezvous algorithm of cost $O(d^4)$ and later extend it to obtain a $O(d^{2+\varepsilon})$ cost solution.

Quad-tree hierarchy $\mathcal{H}_Q$: The first hierarchy of partitions $\mathcal{H}_Q$ has a form of an infinite *quad-tree* like structure, (for information on quad-trees see, e.g., [28]) in which the central point of the partition from each layer is aligned with the origin $(0,0)$ of the plane. The i^{th} layer L_Q^i of the hierarchy, for $i = 0, 1, 2, ...$, is formed of an infinite grid of squares of size 2^i. Hence the lowest level of the hierarchy $\mathcal{H}_Q$ corresponds to the standard integer grid. In layer L_Q^i we denote by $S_Q^i(x,y)$ a square with the corners located at points (listed in the clockwise order counting from the top-left corner) $(x\cdot 2^i, y\cdot 2^i)$, $((x+1)\cdot 2^i, y\cdot 2^i)$, $((x+1)\cdot 2^i, (y-1)\cdot 2^i)$ and $(x\cdot 2^i, (y-1)\cdot 2^i)$. The overlapping squares at two neighboring layers L_Q^i and L_Q^{i+1} are engaged in a parent-child relationship. In particular, a square $S_Q^{i+1}(x,y)$ at layer L_Q^{i+1} has four children squares $S_Q^i(2x,2y)$, $S_Q^i(2x+1,2y)$, $S_Q^i(2x+1,2y-1)$ and $S_Q^i(2x,2y-1)$ at the layer L_Q^i, for $i = 0, 1, 2, \ldots$

Central-square hierarchy $\mathcal{H}_C$: The second hierarchical partition $\mathcal{H}_C$ is formed of (infinitely many) enumerated layers, where the i^{th} layer L_C^i is an infinite grid with squares of size 2^i, for $i = 1, 2, \ldots$ Each layer in $\mathcal{H}_C$ is aligned, s.t., the origin $(0,0)$ of the plane is associated with the center of one of the squares in this layer. In particular, in layer L_C^i we denote by $S_C^i(x,y)$ a square with the corners located at points (listed in the clockwise order counting from the top-left corner) $((x\cdot 2^i) - 2^{i-1}, (y\cdot 2^i) + 2^{i-1})$, $((x\cdot 2^i) + 2^{i-1}, (y\cdot 2^i) + 2^{i-1})$, $((x\cdot 2^i) + 2^{i-1}, (y\cdot 2^i) - 2^{i-1})$, and $((x\cdot 2^i) - 2^{i-1}, (y\cdot 2^i) - 2^{i-1})$.

Hierarchy $\mathcal{H}_{QC}$: By $\mathcal{H}_{QC}$ we understand the infinite sequence of plane partitions

$$\pi_1 = L_Q^0, \pi_2 = L_C^1, \pi_3 = L_Q^1, \pi_4 = L_C^2, \pi_5 = L_Q^2, \ldots$$

Hence π_i is a grid partition of the 2D-space into squares of size $2^{\lfloor i/2 \rfloor}$, with grid point having each coordinate of the form $2^{\lfloor i-1/2 \rfloor} + k2^{\lfloor i/2 \rfloor}$, for any integer k. To assure that each layer of $\mathcal{H}_{QC}$ forms an exact partition, we assume that each

square contains, besides its interior points, its right and top open sides as well as the top-right vertex.

Intuitively, the hierarchical partition $\mathcal{H}_{QC}$ provides a mechanism used to guarantee that any two points p_1 and p_2 located at distance d in the 2D-space are covered by some square of size $O(d)$ in either $\mathcal{H}_Q$ or in $\mathcal{H}_C$. The smallest square in the hierarchical structure with this property is referred to as the *rendezvous square* $R(p_1, p_2)$.

We state two lemmas directly following from the definitions of $\mathcal{H}_Q$ and $\mathcal{H}_C$.

Lemma 1. *Any square $S_C^i(x, y)$ located in layer L_C^i, for $i = 1, 2, \ldots$, encapsulates exactly four squares $S_Q^{i-1}(2x - 1, 2y + 1)$, $S_Q^{i-1}(2x, 2y + 1)$, $S_Q^{i-1}(2x, 2y)$, and $S_Q^{i-1}(2x - 1, 2y)$, in layer L_Q^{i-1}.*

Lemma 2. *Any square $S_Q^i(x, y)$ located in layer L_Q^i, for $i = 1, 2, \ldots$, overlaps with exactly four squares $S_C^i(x, y)$, $S_C^i(x+1, y)$, $S_C^i(x+1, y-1)$ and $S_C^i(x, y-1)$ in layer L_C^i.*

The following tree-like structure $\mathcal{T}_{QC}$ is useful to visualize the functioning of our approach: each square of every layer of $\mathcal{H}_{QC}$ is a node of $\mathcal{T}_{QC}$. For every such square S from layer $i > 1$ of $\mathcal{H}_{QC}$ the squares from layer $i - 1$ intersecting S are children of S in $\mathcal{T}_{QC}$. By lemmas 1 and 2 $\mathcal{T}_{QC}$ is a quaternary tree. $\mathcal{T}_{QC}$ is infinite (does not a have a root) and its leaves are unit squares of the integer grid.

The observation stated in the following lemma is the basis of the complexity proof of our algorithm.

Lemma 3. *For any two points p_1 and p_2 located at distance d in the 2D-space there exists a square in $\mathcal{H}_{QC}$ of size $O(d)$ that contains p_1 and p_2.*

Proof. Assume that $2^{i-1} < d \leq 2^i$. Consider five consecutive layers from $\mathcal{H}_{QC}$, L_Q^i, L_C^{i+1}, L_Q^{i+1}, L_C^{i+2}, and L_Q^{i+2}, where i meets the condition stated. Since any layer in $\mathcal{H}_{QC}$ forms a partition of the 2D-space into squares, within the layer L_Q^{i+2} there exists a unique square $S_Q^{i+2}(x, y)$ containing p_1. Since four quad-tree children of $S_Q^{i+2}(x, y)$ partition it, exactly one of them, a square belonging to L_Q^{i+1} also contains p_1. Similarly, there is exactly one square from L_Q^i, one of the sixteen grandchildren of $S_Q^{i+2}(x, y)$ in the quad-tree, which contains p_1, see Figure 1. Let $S^* \in L_Q^i$ denote the square containing p_1. Note that, since $d \leq 2^i$, the d-neighborhood of S^* intersects nine squares of L_Q^i - the square S^* itself and eight squares surrounding S^*. Hence p_2 must belong to one of these nine squares.

We consider now the cases depending which of these sixteen squares is S^*. Due to the symmetry of the five layers in $\mathcal{H}_{QC}$, w.l.o.g., we can consider only one, e.g., the top-left quadrant of $S_C^{i+2}(x, y)$, which contains four squares at L_Q^i. One of the three possible cases can occur.

Case 1. If S^* containing p_1 corresponds to the square depicted by α_1, then p_2 must be located within $S_C^{i+2}(x, y)$, since $S_C^{i+2}(x, y)$ contains the entire d-neighborhood of α_1.

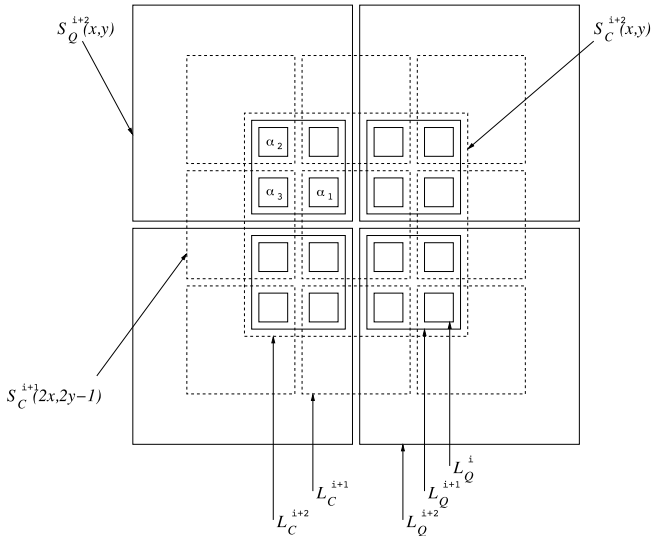

Fig. 1. The symmetric structure of layers $L_Q^i, L_C^{i+1}, L_Q^{i+1}, L_C^{i+2}$ and L_Q^{i+2}

Case 2. If S^* corresponds to the square depicted by α_2 then p_2 must be located within $S_Q^{i+2}(x, y)$.

Case 3. If S^* is one of the two remaining, symmetrically located squares within the selected quadrant depicted by α_3, then p_2 is located either in $S_Q^{i+2}(x, y)$, $S_C^{i+2}(x, y)$ or in $S_C^{i+1}(2x, 2y - 1)$.

Thus in all three cases there exists a square within layers L_Q^i, L_C^{i+1}, L_Q^{i+1}, L_C^{i+2}, and L_Q^{i+2}, that contains both p_1 and p_2. Moreover, since squares at those layers are of size $O(d)$, the thesis of the lemma follows. □

In fact a stronger result holds too.

Corollary 1. *Take any fragment of $\mathcal{H}_{QC}$ formed of three consecutive layers L_Q^i, L_Q^{i+1}, and L_Q^{i+2} in $\mathcal{H}_Q$ interleaved with two respective layers L_C^{i+1} and L_C^{i+2} in $\mathcal{H}_C$, s.t., $d \leq 2^i$. This fragment contains also a square that contains p_1 and p_2.*

Proof. The three cases from Lemma 3 also apply here. □

Space-covering sequences. Recall, that at the lowest layer of the structure $\mathcal{H}_{QC}$ there is partition π_1 containing *unit squares* of L_Q^0. On the basis of unit squares, we form *atomic* space-covering sequences, constituting basic components of length $O(1)$. The atomic sequence based on a point p belonging to a unit square in the 2D-space is referred to as the source $s(p)$ of p. We suppose that $s(p)$ is the sequence formed of a single point, being the top left corner of the unit square containing p. At any higher layer π_i we recursively form a longer

sequence associated with each square S from this layer by concatenating the sequences from layer π_{i-1}, corresponding to the children of S in the tree $\mathcal{T}_{QC}$ (i.e., for even i - squares from π_{i-1} that are covered by S^*, see Lemma 1, and, for odd i - squares from π_{i-1} that overlap with S^*, see Lemma 2). To perform such construction we need to define *connectors* linking together the portions of the space-covering curve already created for the squares at the lower level. For the simplicity of presentation we suppose that the portion of the space-covering curve corresponding to any square S starts and ends at the top left corner of S. We assume that the children of the same parent of $\mathcal{T}_{QC}$ are arranged in the clockwise order starting from the top left child. The connectors are the line segments joining the top left corners of the siblings. The connectors are used twice, once in the increasing order of the siblings (which, by convention, corresponds to the left-to-right traversal of the space-covering sequence) and the other time in the decreasing order. For example connectors A, B, C link, respectively, squares $S^i_Q(2x-1, 2y+1)$, $S^i_Q(2x, 2y+1)$, $S^i_Q(2x, 2y)$ and $S^i_Q(2x-1, 2y)$ in Figure 2 (a) and squares $S^i_C(x, y)$, $S^i_C(x+1, y)$, $S^i_C(x+1, y-1)$ and $S^i_C(x, y-1)$ in Figure 2(b). Note that, in the former case, the obtained curve already starts and ends at the top left corner of the parent square $S^{i+1}_C(x, y)$. In the latter case, we also add connector D (cf. Fig. 2 (b)), taken twice, in order to make the constructed portion start and end at the top left corner of the parent square $S^i_Q(x, y)$. This process can be iterated for as long as it is required. The space-covering sequence associated with the *rendezvous square* $R(p_1, p_2)$ will be used to obtain rendezvous of two participating agents located initially at points p_1 and p_2. In what follows, we show how to explore the space-covering sequence efficiently during the rendezvous process.

Note that, since each layer in $\mathcal{H}_{QC}$ is a partition of the 2D-space into squares, every point p in the 2D-space can be associated with a unique infinite *list of squares* $S_1(p), S_2(p), \ldots$ from the consecutive layers $\pi_1(p), \pi_2(p), \ldots$ in $\mathcal{H}_{QC}$, respectively, s.t., each square contains p. Note also, that the space-covering sequence associated with $S_i(p)$, for each $i \geq 1$ forms a contiguous segment of positions in the space-covering sequence associated with $S_{i+1}(p)$. Let $left(S)$ and $right(S)$ denote, respectively, the leftmost and the rightmost position on the space-covering sequence corresponding to square S. Then we have $left(S_{i+1}(p)) \leq left(S_i(p)) \leq right(S_i(p)) \leq right(S_{i+1}(p))$.

Lemma 4. *For any two points p_1 and p_2 at distance d in the 2D-space, the space-covering sequence associated with the rendezvous square $R(p_1, p_2)$, is of length $O(d^4)$.*

Proof. Recall that the lengths of space-covering sequences associated with squares in $\pi_1 = L^0_Q$ are $O(1)$. Assume now that the sequences associated with squares of size 2^{k-1} at layer L^{k-1}_Q are of size $f(2^{k-1})$, for some positive integer function f. Note that the connectors from layer k, used for linking the endpoints of the space-covering sequences for the squares from layer $k-1$, are of length $O(2^k)$. Thus the space-covering sequences associated with squares in L^k_C have length $4 \cdot f(2^{k-1}) + O(2^k)$. Similarly, we

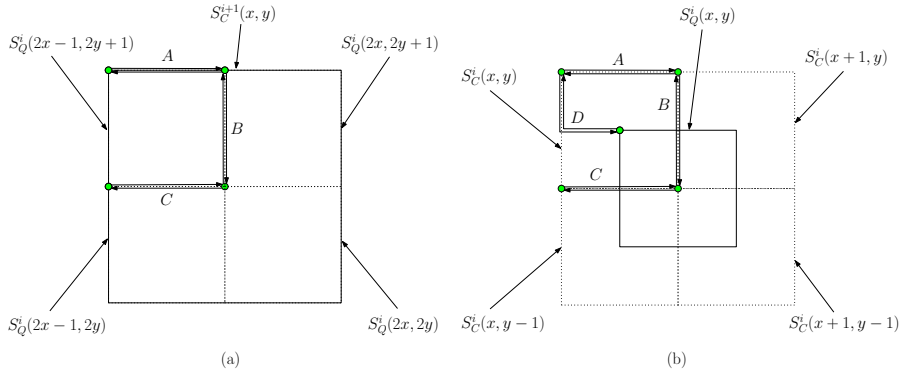

Fig. 2. Connectors between siblings (dotted line squares) and parent (solid lines). In case (a) parent comes from $\mathcal{H}_C$ family and in case (b) parent comes from $\mathcal{H}_Q$.

associate the squares in layer L^k_Q with space-covering sequences of length $4(4 \cdot f(2^{k-1}) + O(2^k)) + O(2^k) = 16 \cdot f(2^{k-1}) + O(2^k)$. Thus the length of the space-covering sequences associated with the squares in L^k_Q (also in L^k_C) can be described by the recurrence:

(1) $f(2^1) = O(1)$,
(2) $f(2^k) = 16 \cdot f(2^{k-1}) + O(2^k)$, for any integer $k > 1$.

Since, by Lemma 3, the rendezvous square $R(p_1, p_2)$ is of size $O(d)$, the recurrence will be applied at most $\log d + O(1)$ times. Thus the total length of the space-covering sequences for the squares from the layers L^k_Q and L^k_C is $O(d^4)$. □

We show now, that we can remove from $\mathcal{H}_{QC}$ certain layers coming from $\mathcal{H}_C$, obtaining a new hierarchy $\mathcal{H}^*_{QC}$, such that the distance separating p_1 and p_2 on the corresponding space-covering sequence can be reduced to $O(d^{2+\varepsilon})$, for any constant $\varepsilon > 0$.

Hierarchy $\mathcal{H}^*_{QC}$: Fix a natural number z. $\mathcal{H}^*_{QC}$ is formed of a sequence of blocks, each block containing $z + 4$ layers. The i-th block, for $i = 0, 1, \ldots$, contains z consecutive layers of $\mathcal{H}_Q$ - $L_Q^{i(z+2)}, L_Q^{i(z+2)+1}, \ldots, L_Q^{i(z+2)+z-1}$, followed by four layers $L_C^{i(z+2)+z}, L_Q^{i(z+2)+z}, L_C^{i(z+2)+z+1}$, $L_Q^{i(z+2)+z+1}$. E.g., the portion of $\mathcal{H}^*_{QC}$ corresponding to the first two blocks is

$$L^0_Q, L^1_Q, ..., L^{z-1}_Q, L^z_C, L^z_Q, L^{z+1}_C, L^{z+1}_Q, L^{z+2}_Q, ..., L^{2z+1}_Q, L^{2z+2}_C, L^{2z+2}_Q, L^{2z+3}_C, L^{2z+3}_Q$$

Note that, if some layer of $\mathcal{H}^*_{QC}$ comes from $\mathcal{H}_C$, i.e., it is L^k_C, for some value of k, its predecessor in $\mathcal{H}^*_{QC}$ is L^{k-1}_Q, hence Lemma 1 directly applies. On the other hand, for any layer of $\mathcal{H}^*_{QC}$ coming from $\mathcal{H}_Q$, say L^k_Q, its predecessor in $\mathcal{H}^*_{QC}$ is either L^k_C or L^{k-1}_Q. In the former case Lemma 2 directly applies. In the latter case each square from L^k_C partitions exactly into four squares of L^{k-1}_Q, hence

the claim of Lemma 2 is true as well. Therefore the tree $\mathcal{T}^*_{QC}$ representing hierarchy $\mathcal{H}^*_{QC}$ may be obtained and the space-covering sequences are constructed similarly as in the case of $\mathcal{H}_{QC}$, where in case of missing layers from $\mathcal{H}_C$ the concatenation process is performed according to the structure of squares located in the hierarchical partition $\mathcal{H}_Q$. Similarly, in $\mathcal{H}^*_{QC}$ the rendezvous square $R(p_1, p_2)$ of two points p_1 and p_2 is the square on a lowest layer which contains the two points. The following lemma holds.

Lemma 5. *For any constant $\varepsilon > 0$, there exists $\mathcal{H}^*_{QC}$ in which the space-covering sequence associated with the rendezvous square $R(p_1, p_2)$ of two arbitrary points p_1 and p_2 located at distance d in the* 2D*-space is of length $O(d^{2+\varepsilon})$.*

Proof. According to Corollary 1, $R(p_1, p_2)$ - the rendezvous square for p_1 and p_2 is present in $\mathcal{H}^*_{QC}$. Moreover, $R(p_1, p_2)$ belongs to layer k, such that $k = \log d + z + O(1) = \log d + O(1)$, for some constant z, meaning that the size of the rendezvous square is still $O(d)$. The size of the space-covering sequence at layer k in $\mathcal{H}^*_{QC}$ is then bounded by $O(4^k \cdot 4^{2 \cdot \frac{k}{z+2}})$, where contribution $O(4^k)$ comes from the structure of $\mathcal{H}_Q$ and $O(4^{2 \cdot \frac{k}{z+2}})$ comes from $\mathcal{H}_C$. Note that we can choose the constant z, s.t., the exponent in the second term is the smallest constant that we require. Also since $k = \log d + z + O(1)$ the second term translates to

$$O(4^{2 \cdot \frac{\log d + z + O(1)}{z+2}}) \leq O(4^{2 \cdot \frac{\log d + z + O(1)}{z}}) = O(4^{\frac{2}{z} \cdot \log d}) = O(d^{\frac{4}{z}}).$$

Thus for any $\varepsilon > 0$, we can find an integer constant $z \geq \frac{4}{\varepsilon}$, s.t., the length of the space-covering sequence in $\mathcal{H}^*_{QC}$ for any two points at distance d in the 2D-space is $O(d^{2+\varepsilon})$. □

3 The Rendezvous Algorithm

Our rendezvous algorithm utilizes the nested structure of space-covering sequences associated with the list of squares defined for each point p in the 2D-space. The agent determines the list of squares in $\mathcal{H}^*_{QC}$ according to its initial location p. Then, for each square $S_i(p)$ from the list, the agent visits the leftmost and the rightmost point on the space-covering sequence, corresponding to the computed traversal of S_i, until it encounters the other agent.

Algorithm. `RV` (point $p \in$ 2D-space)
1. visit an integer grid point of the 2D-space which the closest to p;
2. $i \leftarrow 1$;
3. **repeat**
4. Let $S_i(p)$ be the square from layer i of $\mathcal{H}^*_{QC}$ containing p;
5. Go right on the space-covering curve until reaching $right(S_i(p))$;
6. Go left on the space-covering curve until reaching $left(S_i(p))$;
7. $i \leftarrow i + 1$;
8. **until** rendezvous is reached ;

Theorem 1. *Two location aware agents located in points p_1 and p_2 at distance d in the* 2D*-space executing algorithm* RV *will meet after traversing asynchronously a trajectory of length $O(d^{2+\varepsilon})$, for any constant $\varepsilon > 0$.*

Proof. By Corollary 1, there exists the square $R(p_1, p_2)$ in $\mathcal{H}^*_{QC}$ containing p_1 and p_2. This square is on the list of squares considered in step 4 of the algorithm by each agent. Suppose, by symmetry, that agent $\mathcal{A}_1$ is the first one to terminate at time t step 6 of the algorithm for the iteration i during which $S_i(p_1) = R(p_1, p_2)$. If agent $\mathcal{A}_2$ has not yet completed its execution of step 1 of the algorithm, it must belong to $S_1(p_2)$ - a unit square included in $R(p_1, p_2)$ - and $\mathcal{A}_1$ completing step 6 must meet $\mathcal{A}_2$. Hence we may assume that, at time t agent $\mathcal{A}_2$ must be traversing a portion of the space-covering sequence, corresponding to some square included in $R(p_1, p_2)$. All intermediate space-covering sequences associated with the predecessors of $R(p_1, p_2)$ in the lists of squares for p_2 form the space-covering sequences included in the interval $[left(R(p_1, p_2)), right(R(p_1, p_2))]$. Hence, while agent $\mathcal{A}_1$ traverses this interval in step 6, it must meet agent $\mathcal{A}_2$, which, by our assumption, is within this segment at time t. The total length of the trajectory adopted by the agents is linear in the size of the space-covering sequence due to exponential growth of intermediate sequences associated with the consecutive squares in the list of squares. □

Remark. Note that there are $\Omega(d^2)$ integer grid points within distance d from any point in the 2D-space. Therefore, since the adversary can keep one of the agents immobile, the rendezvous implies that the other agent has to explore its d-environment, adopting a route of length $\Omega(d^2)$. Thus the cost of our algorithm is almost optimal.

4 Final Comments and Open Problems

Our algorithm may be extended in a standard way in order to solve *gathering* of a set of agents. However, instead of instantly stopping while a meeting occurs, a group of $n \geq 2$ agents involved in the meeting chooses a leader (e.g. the agent with the lexicographically smallest initial position) and all agents follow its route. If the total number of agents is known in advance they gather after traversing a route of length $O(d^{2+\varepsilon})$, provided they were initially within a d-neighborhood of some point in the 2D-space.

References

1. Abraham, I., Dolev, D., Malkhi, D.: LLS: a locality aware location service for mobile ad hoc networks. In: Proc. DIALM-POMC 2004, pp. 75–84 (2004)
2. Alpern, S.: The rendezvous search problem. SIAM J. Control and Optimization 33, 673–683 (1995)
3. Alpern, S., Baston, V., Essegaier, S.: Rendezvous search on a graph. J. App. Probability 36, 223–231 (1999)

4. Anderson, E., Lin, J., Morse, A.S.: The Multi-Agent Rendezvous Problem - The Asynchronous Case. In: 43rd IEEE Conf. on Decision and Control, pp. 1926–1931 (2004)
5. Anderson, E., Fekete, S.: Two-dimensional rendezvous search. Operations Research 49, 107–118 (2001)
6. Anderson, E., Essegaier, S.: Rendezvous search on the line with indistinguishable players. SIAM J. on Control and Optimization 33, 1637–1642 (1995)
7. Baston, V., Gal, S.: Rendezvous on the line when the players' initial distance is given by an unknown probability distribution. SIAM J. on Control and Optimization 36, 1880–1889 (1998)
8. Bose, P., Morin, P., Stojmenovic, I., Urrutia, J.: Routing with guaranteed delivery in ad hoc wireless networks. Wireless Networks 7(6), 609–616 (2001)
9. Buchin, K.: Constructing Delaunay Triangulations along Space-Filling Curves. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 119–130. Springer, Heidelberg (2009)
10. Cieliebak, M., Flocchini, P., Prencipe, G., Santoro, N.: Solving the Robots Gathering Problem. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 1181–1196. Springer, Heidelberg (2003)
11. Czyzowicz, J., Ilcinkas, D., Labourel, A., Pelc, A.: Asynchronous deterministic rendezvous in bounded terrains. In: SIROCCO (2010)
12. Czyzowicz, J., Labourel, A., Pelc, A.: How to meet asynchronously (almost) everywhere. In: Proc. of SODA 2010, pp. 22–30 (2010)
13. Czyzowicz, J., Kosowski, A., Pelc, A.: How to Meet when you Forget: Log-space Rendezvous in Arbitrary Graphs. In: Proc. of 29th Ann. Symp. on Principles of Distributed Computing, PODC 2010 (to appear, 2010)
14. De Marco, G., Gargano, L., Kranakis, E., Krizanc, D., Pelc, A., Vaccaro, U.: Asynchronous deterministic rendezvous in graphs. Th. Comp. Sc. 355, 315–326 (2006)
15. Dessmark, A., Fraigniaud, P., Kowalski, D., Pelc, A.: Deterministic rendezvous in graphs. Algorithmica 46, 69–96 (2006)
16. Emek, Y., Gasieniec, L., Kantor, E., Pelc, A., Peleg, D., Su, C.: Broadcasting in UDG radio networks with unknown topology. Distributed Computing 21(5), 331–351 (2009)
17. Emek, Y., Kantor, E., Peleg, D.: On the effect of the deployment setting on broadcasting in Euclidean radio networks. In: Proc. PODC 2008, pp. 223–232 (2008)
18. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous oblivious robots with limited visibility. In: Ferreira, A., Reichel, H. (eds.) STACS 2001. LNCS, vol. 2010, pp. 247–258. Springer, Heidelberg (2001)
19. Fraigniaud, P., Pelc, A.: Deterministic rendezvous in trees with little memory. In: Taubenfeld, G. (ed.) DISC 2008. LNCS, vol. 5218, pp. 242–256. Springer, Heidelberg (2008)
20. Gal, S.: Rendezvous search on the line. Operations Research 47, 974–976 (1999)
21. Ganguli, A., Cortés, J., Bullo, F.: Multirobot rendezvous with visibility sensors in nonconvex environments. IEEE Transactions on Robotics 25(2), 340–352 (2009)
22. Gotsman, C., Lindenbaum, M.: On the metric properties of discrete space-filling curves. IEEE Transactions on Image Processing 5(5), 794–797 (1996)
23. Kowalski, D., Malinowski, A.: How to meet in anonymous network. Th. Comp. Science 399, 141–156 (2008)
24. Kozma, G., Lotker, Z., Sharir, M., Stupp, G.: Geometrically aware communication in random wireless networks. In: Proc. PODC 2004, pp. 310–319 (2004)
25. Kuhn, F., Wattenhofer, R., Zhang, Y., Zollinger, A.: Geometric ad-hoc routing: theory and practice. In: Proc. PODC 2003, pp. 63–72 (2003)

26. Moon, B., Jagadish, H.V., Faloutsos, C., Saltz, J.H.: Analysis of the Clustering Properties of the Hilbert Space-Filling Curve. IEEE Transactions on Knowledge Data Engineering 14(1), 124–141 (2001)
27. Paterson, M.S., Yao, F.F.: Efficient binary space partitions for hidden-surface removal and solid modeling. Discr. and Comp. Geom. 5(5), 485–503 (1990)
28. Samet, H.: The quadtree and related hierarchical data structures. Surveys 16(2), 187–260 (1984)
29. Xu, B., Chen, D.Z.: Density-Based Data Clustering Algorithms for Lower Dimensions Using Space-Filling Curves. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 997–1005. Springer, Heidelberg (2007)
30. Yamashita, M., Kameda, T.: Computing on Anonymous Networks: Part I-Characterizing the Solvable Cases. IEEE Trans. Parallel Dist. Syst. 7, 69–89 (1996)
31. Yu, X., Yung, M.: Agent rendezvous: a dynamic symmetry-breaking problem. In: Meyer auf der Heide, F., Monien, B. (eds.) ICALP 1996. LNCS, vol. 1099, pp. 610–621. Springer, Heidelberg (1996)