Scientia Iranica D (2021) 28(3), 1515–1528

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
http://scientiairanica.sharif.edu

Research Note

Multi-robot exploration on grids with a bounded time

M. Davoodi*, E. Delfaraz, S. Ghobadi, and M. Masoori

Department of Computer Science and Information Technology, Institute for Advanced Studies in Basic Science, Gavazang, Zanjan.

Received 17 November 2018; received in revised form 18 August 2020; accepted 26 October 2020

KEYWORDS
Approximation algorithm;
Path planning;
Exploration;
Grid environments;
Lower bound.

Abstract. In this paper, the problem of exploring a grid environment in the offline setting is studied. The goal is to propose an algorithm to find the minimum number of robots for exploring a rectangular grid environment with n rows and m columns, denoted by $R(n, m)$, at a predefined time T. In case of no obstacles in the environment, an optimal solution is proposed for the problem. In another case when the environment may contain some obstacles, it is pointed out that the problem is NP-complete and it cannot be approximated within better than a factor 2. Finally, a 4-approximation algorithm is presented in order to explore $R(n, m)$ in the presence of obstacles.

1. Introduction

Robot path planning is one of the most fundamental problems in robotics that has received much attention among researchers. This problem is the task of finding a path between the start and goal positions in which finding an efficient path is the main purpose. Potential applications of mobile robots include a wide range of functions such as *vacuum the room, clearing a mine, harvester*, etc. [1,2]. For each autonomous robot, given start and goal positions, solutions to this problem can be applied in scenarios where each robot has to find a minimum path from its start to its goal position without hitting obstacles 10 or other robots in an environment [3].

Robot path planning has been extensively studied [4–6]. Due to the shape of robots and their abilities, several studies have tried to improve the efficiency of problem by considering across a variety of dimensions pertaining to the environment. The environment could be *either offline or online* [7]. In the offline version, the environment is known to the robots in advance; in contrast, in the online version, the environment is unknown to the robots and they should move without knowing the environment. Also, the environment could be modeled as discrete or continuous [5,8]. The discrete model which is known as grid environments is a powerful model that it is easy to implement. Variations in single or multi-*objective* problem is also considered [5,8,9]. For instance, authors [8] proposed a multi objective model with focus on length and clearance of the finding path. In a wide variety of applications, in order to perform the job, one needs more than one robot to complete the task. So, single or multiple robots versions of the path planning problem have also been considered [3,9]. Another variation of the path planning problem could be defined as static or dynamic environment [10,11]. In the static setting, there is not any change in the environment in the whole process; however, in the dynamic setting, the environment could undergo changes in each time step.

Robot exploration is the problem of exploring the whole given environment in the minimum amount of time. This problem is among the most widely studied problems in path planning. According to different

*. *Corresponding author.*
E-mail addresses: mdmonfared@iasbs.ac.ir (M. Davoodi); esmaiel.delfaraz@gssi.it (E. Delfaraz); sajjad.ghobadi@gssi.it (S. Ghobadi); mahtab.masoori@gmail.com (M. Masoori)

doi: 10.24200/sci.2020.51795.2366

variants of the path planning, robot exploration also has been investigated in different settings. However, in this paper, the concentration is on the exploration problem when the environment is a rectangular grid. Here, the robots are unable to communicate. Moreover, it is assumed that the robots are uniform and the same in shape and moving ability.

2. Related work

Path planning is one of the most challenging problems in the field of robotics. It is not easy to find the shortest possible path between the start and goal positions in the minimum amount of time. Researchers have tried to find such a path by heuristic methods. However, the heuristic approaches presented in [5,8] do not guarantee achieving optimal paths. But, Davoodi [4] presented an efficient algorithm for the problem. Haynes et al. [6] used a virtual geometric structure in order to coordinate a team of robots and proposed an efficient algorithm for solving the online version of it.

The *exploration problem* is closely related to the *Hamiltonian cycle* and *Traveling-Salesman Problems (TSP)* both of which are NP-complete [12]. In its full generality, these problems cannot be approximated. Christofides [13] considered a specific version of this problem which is named the metric *TSP*. Although this problem remains NP-complete, it is no longer hard to approximate. Indeed, the algorithm proposed by Christofides can approximate the problem within a factor of 3/2. The mechanism presented in [14–16] having identical guarantee of $1+\epsilon$ approximation factor for TSP. Arkin et al. [17] proposed a 53/40 factor approximation algorithm for this problem. Arkin et al. [17] and Ntafos [18] gave 3/4 and 5/6 factor deterministic algorithms, respectively.

There is a polynomial-time reduction from TSP to the *exploration Problem* in grid graphs. The main idea of the problem is that finding a minimum cost cycle for a weighted graph equals obtaining an optimal exploration path for the corresponding grid graph. Although, TSP is NP-complete and the *grid exploration problem* without holes is still an important open problem [19,20].

In grid exploration, the aim is to find a shortest possible tour in a grid environment in which every cell of the grid should be visited at least once [19]. In fact, the robot is assumed to occupy one cell of the grid and, in each time step, moved one cell up, down, left or right [19,21]. Consider the grid graph $G(V, E)$ that represents a grid environment $\mathcal{E}$ in which each vertex of V and edge of E in G corresponds to a free cell and connection between two adjacent free cells, respectively [19]. It is clear that $|V|$ is equal to the number of free cells in $\mathcal{E}$ and the maximum degree of each vertex V is 4. A grid graph G in the absence of obstacles is called *solid* grid graph. Otherwise, it is called *general* grid graph [22]. Itai et al. [23] proved that finding a Hamiltonian cycle in general grid graphs is an NP-complete problem. Based on this study, the exploration problem for general grid graphs is NP-complete as well. Umans and Lenhart [22] presented a polynomial time algorithm in solid grid graphs with N vertices in which the existence of a Hamiltonian cycle could be determined at $O(N^4)$ time. Otherwise, they reported at $O(N^2)$ time that no Hamiltonian cycle existed. The existence of a Hamiltonian Path in rectangular graph $R(n, m)$ came from [23] for the first time. They presented a linear time algorithm in the number of vertices of the graph. There is a polynomial time algorithm for deciding if a given $R(n, m)$ has a Hamiltonian cycle if and only if $m \times n$ is even. Otherwise, $R(m, n)$ has no Hamiltonian cycle [24]. Fischer and Hungerländer [25] presented an optimal approach to exploring $R(n, m)$ by one robot in the offline setting which the environment is known in advance. The length of the exploration path P means that the sum of the lengths of edges on it is $m \times n + 1$ if the value of $m \times n$ is odd. Otherwise, its value is $m \times n$. Davoodi et al. [26] investigated this problem in the online setting where a robot only has accurate knowledge about its four adjacent ones without knowing the location of the other cells. They presented an optimal algorithm when the start position lied on the boundary of $R(n, m)$. Also, the competitive ratio of their algorithm is $1 + 4/mn$ when the start position of the robot is an arbitrary cell. In [27], the authors proposed an incremental algorithm for efficient frontier detection for robot exploration. Gabriely and Rimon [28] transformed a continues area to discrete cells and presented an algorithm to explore cells by only one robot. In fact, they considered a continuous planar area by a square-shaped tool attached to a mobile robot which is a square of size D. They subdivided the area into square cells of size $2D$ in which the number of cells is N, regardless of whether or not the cells were partially covered by obstacles. They proved that the total complexity of their method was $O(N)$ and the length of the exploration was $c = N \times 4D$, which is optimal. The main drawback of their algorithm is that it does not work well for an area that cannot be divided into cells of size $2D$, completely. Czyzowicz et al. [29] provided the exact bound on the collision-free exploration time for trees in the offline setting. In the online setting, they proposed collision-free exploration strategies running in $O(n^2)$ rounds in the tree network. Disser et al. [30] proposed a tight lower bound on the number of robots needed for any competitive algorithm for tree exploration. The authors in [31] provided a 3-competitive algorithm that divided the tree into subtrees during the exploration process.

Although the above-mentioned studies have

achieved good results in optimization of exploring in a condition where the exploration time of the robots is not determined, in practice each job needs to be performed in a limited time T and therefore, these methods may not work well. The aim of this study is to find the minimum numbers of robots in order to explore a *Rectangle Grid Environment (RGE)* at maximum predefined time T. Application of this problem includes emergency cases where time is generally crucial and vital; for instance, probing a whole area in natural disasters like earthquakes or flooding in order to find injured people, sweeping a field by a harvester for gathering crops, or visiting an area in space by a space explorer in a predefined time.

In this paper, first, the case where there is no obstacle in RGE is considered and an optimal algorithm to find the minimum number of robots for exploring the environment is presented. Second, by using the Hamiltonian cycle problem, it is shown that determining the minimum number of robots for exploring RGE in the presence of obstacle cells is NP-hard and it cannot be approximated within a factor better than 2. Furthermore, a 4-approximation factor algorithm for this problem is proposed.

3. Methodology

Consider a rectangular grid environment $R(n, m)$ whose cells have unit size, where n and m are the number of rows and columns, respectively, and a predefined time T. In this study, the objective is to compute the minimum number of robots to explore $R(n, m)$ in which each robot has a limited time T to explore a certain part of RGE. Notice that an exploration path is a tour, T must be an even number. Each robot can just occupy one cell at any time step $t \leq T$. It is assumed that each robot can move freely in four cardinal directions and each cell should be visited by at least one robot.

Suppose that the exploration path of an arbitrary robot rb_i is EP_i which is shown as a sequence $c_1^i, c_2^i, \cdots, c_{e-1}^i, c_e^i$ in which $\forall i, j, c_j^i$ is a free cell with a unit size and $c_1^i = c_e^i$ for some $e \in [T]$. Let $visit_j$ be the number of times that rb_i visits cell c_j. The length of exploration path EP_i is denoted by $|EP_i|$. Thus, it is wise to be careful that the exploration path EP_i does not contain two arbitrary cells $c_k = c_j$ such that $k \neq j$, as possible, except for c_1 and c_e.

Definition 1. *Given a robot rb_i, the number of "misses" m_i in the rb_i's exploration path $EP_i = c_1^i, c_2^i, \cdots, c_{e-1}^i, c_e^i$ is defined as $m_i = T - |EP_i| + \sum_{j=1}^{e} (visit_j - 1) - 1$.*

Simply put, m_i represents the amount of time robot rb_i wastes in an exploration process. So, m_i is equal to the difference between the predefined time T and the length of exploration path EP_i, rb_i intends to walk through plus the total number of times rb_i revisits some cells more than once in EP_i minus 1 (since the start and goal position are the same and it is required to be visited twice). Let $nrb_{\mathcal{A}}$ denote the number of robots an arbitrary algorithm $\mathcal{A}$ decides to use for exploring RGE. Let $M_{\mathcal{A}} = \sum_{i=1}^{nrb_{\mathcal{A}}} m_i$ be the total number of misses of $\mathcal{A}$ in the exploration of RGE. If $\mathcal{A}$ is obvious from the context, the $\mathcal{A}$ subscript will be dropped (i.e., nrb is used instead of $nrb_{\mathcal{A}}$, M instead of $M_{\mathcal{A}}$).

Definition 2. *An Algorithm $\mathcal{A}$ is optimum for exploring RGE if there is no other algorithm that results in less misses than $\mathcal{A}$. In particular, $M_{\mathcal{A}} \leq M_{\mathcal{A}'}$ for any algorithm $\mathcal{A}'$ exploring RGE.*

Therefore, the aim is to find an optimal algorithm to explore RGE. In other words, the number of robots nrb determines the number of misses M in an RGE exploration process. Also, let nrb_{opt} and M_{opt} be the optimal number of robots and misses in exploring RGE, respectively.

Observation 1. *Consider an arbitrary algorithm $\mathcal{A}$ used to explore $R(n, m)$. If $M_{\mathcal{A}} < z \times T$ for some integer $z \geq 1$, then $nrb_{\mathcal{A}} = nrb_{\text{opt}} + z - 1$.*

Theorem 1. *Consider an algorithm $\mathcal{A}$ exploring RGE. If $M_{\mathcal{A}} = M_{\text{opt}}$, then $nrb_{\mathcal{A}} = nrb_{\text{opt}}$.*

Proof. 5Algorithm $\mathcal{A}$ uses $nrb_{\mathcal{A}}$ robots to explore RGE and each robot rb_i visits $T - m_i$ cells. So:

$$\#\text{ cell} \geq (nrb_{\mathcal{A}} \times T - M_{\mathcal{A}}). \tag{1}$$

By the assumption that $M_{\mathcal{A}} = M_{opt}$,

$$\#\text{ cell} \geq (nrb_{\mathcal{A}} \times T - M_{opt}). \tag{2}$$

Similar argument applies to an optimal solution. Therefore:

$$nrb_{opt} \geq \frac{\#\text{ cell} + M_{opt}}{T}. \tag{3}$$

Thus, by Eqs. (2) and (3), it can be concluded that $nrb_{opt} \geq nrb_{\mathcal{A}}$. □

By following Theorem 1, some patterns will be presented, where the total number of misses is optimal (also the number of robots).

It is easy to show that it is impossible to give one optimal particular pattern in order to explore an arbitrary $R(n, m)$ for any given time T. Unfortunately, the drawing patterns of exploring the environment depend quite heavily on m, n and T. Hence, one cannot hope to come up with an optimal approach in which only one drawing pattern solves the problem. In fact, in this study, the environment will be partitioned into specific areas based on m, n and T in order to

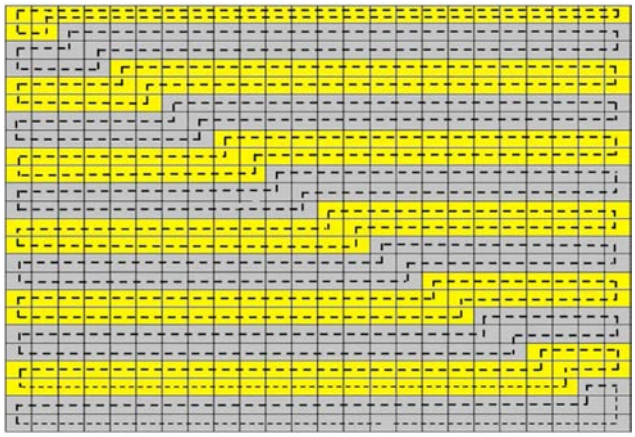

Figure 1. Subdividing $R(x < T/2, y < T/2)$ into L-Shapeds (LSs) and 2-L-Shapeds (2-LSs).

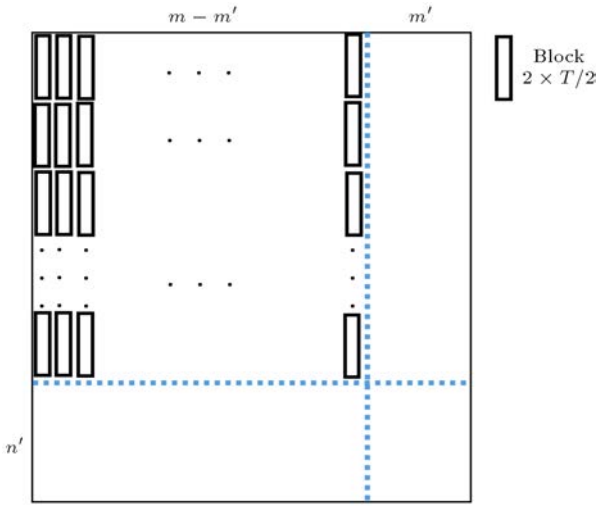

Figure 2. General drawing pattern for exploring $R(n, m)$, where $3 \leq n', m' \leq T/2$.

explore these local areas optimally such that the local optimums give a global optimum. Therefore, first, some of the basic notions and techniques are introduced to explore some particular environments. Then, the more general RGE is considered and explored optimally by subdividing it to particular environments. To be clearer, consider Figure 1, this grid must be explored at time T with minimum number of robots. To this end, each robot should visit most of unexplored cells such that it satisfies the exploration condition means that it must return to the start cell. If m (or n) is equal to $T/2$, the exploring path will be a rectangle environment since every two columns could be explored by one robot. Therefore, $m/2$ robots are required to explore the whole environment. Otherwise, for exploring $R(x, y)$, it is subdivided into *L-Shapeds (LSs)* and *2-L-Shapeds (2-LSs)*, which are described in Section 3.1 in detail. Throughout this study, for exploring $R(n, m)$, $\lceil m \times n/T \rceil$ is considered as an optimal number of robots.

Before jumping right into details, it is more important to take a quick look at the proposed algorithm with respect to T, which is even. In the appendix, some approaches are described for exploring the special environments where the number of rows and/or columns is less than a specific threshold (less than 6) in order to get rid of abundant descriptions. However, for greater columns, according to m and n, splitting strategies are developed to produce small grid areas. Generally speaking, suppose exploring starts from the upper left-hand corner cell with $2 \times T/2$ blocks vertically (see Figure 2) until the remaining rows n' and columns m' are less than $T/2$ and more than 3 (i.e., $3 \leq n', m' \leq T/2$) which will be explained in more detail in Subsections 3.1 and 3.2. It is obvious that no misses occur in exploring $R(m - m', n - n')$. Therefore, it is enough to provide some optimal patterns to explore the rest of the environment. Indeed, with respect to n' and m', different strategies are proposed for the unexplored areas, i.e., $R(n, m')$ and $R(n', m - m')$. In the following sections, optimal patterns are given in order to explore other areas and show that the whole area is explored optimally. Based on the general drawing pattern, $3 \leq n', m' \leq T/2$. So, in the following, a particular environment is examined where $n' < T/2$ and $m' < T/2$.

3.1. Exploring $R(x < T/2, y < T/2)$

Here, it is assumed that the number of rows and columns of RGE is less than $T/2$, which is denoted by $R(x < T/2, y < T/2)$.

Definition 3. *An $LS(a \geq 1, k \geq 2, x)$ is like an RGE, where the total number of cells is at most T, in which a is the number of columns of the first row, where $a \leq T/2 - 2$. Also, k and x show the total number of rows and columns of LS, respectively, where $x, k \leq T/2 - 1$ and $a < x$ (Figure 3(a)).*

Definition 4. *A $2 - LS(a \geq 1, k \geq 3, x, b \geq 1)$ is like an LS, where the total number of cells is at most T, b is the number of columns of the last row, $b \leq T/2 - 2$ (Figure 3(b)), and $a, b < x$.*

For exploring $R(x, y)$, it is subdivided into *LS* and *2-LS*.

First, exploration patterns are presented to explore each of these shapes by one arbitrary robot rb_i at maximum time T and show that m_i is optimal. To this end, it is shown that the total number of misses M occurring during the exploration of $R(x, y)$, is optimal. Consider an *LS*, a grey area A whose width and length are $k - 2$ and x; the rest of *LS* on the top of A which is shown as a white area in Figure 4.

According to a, k, and x, different exploration patterns are presented for exploring an *LS*. The following exploration patterns state optimal results. Con-

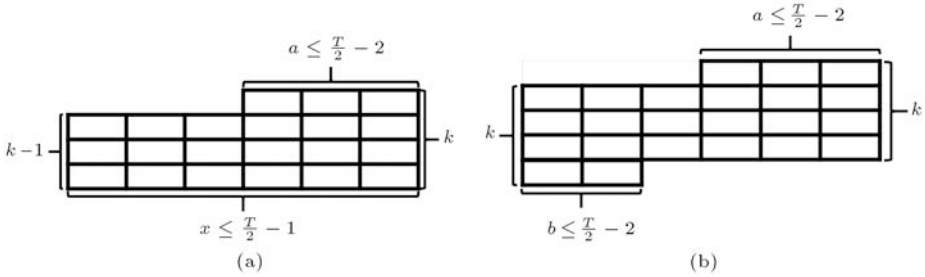

Figure 3. Examples of (a) LSs and (b) 2-LSs environments.

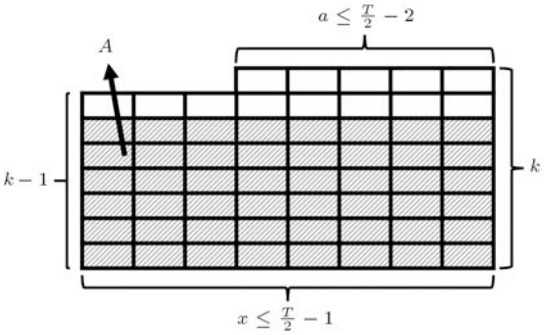

Figure 4. An arbitrary L-Shaped (LS), where width and length of area A are $k-2$ and x, respectively.

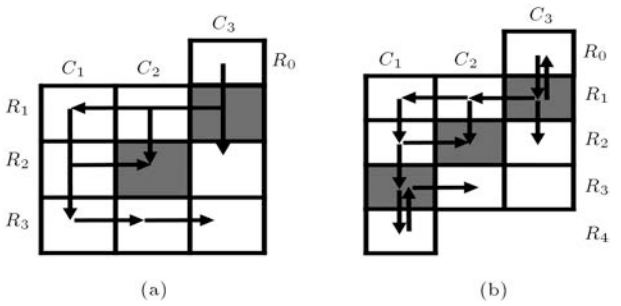

Figure 6. Tight examples on the number of misses for (a) L-Shaped (LS) and (b) 2-L-Shaped (2-LS).

sider $LS(a = 2o, k = 2o+1, x = 2o+1)$ such that $o \in N$ means that a is even, k and x are odd, and the drawing pattern in Figure 5(a) can be applied for this case. The presented drawing pattern is shown in Figure 5(b) for $LS(a = 2o, k = 2o, x = 2o)$ such that $o \in N$. Obviously, no miss occurs in these two cases. Similarly, for other values of $a \ge 2$, k and x, exploration patterns can be presented and the number of misses is at most 1, which is optimal in the worst case.

In the following, it is proven that these drawing patterns are optimal. Also, it is shown that the number of misses of every possible exploration pattern for a given LS in which $a \ge 1$ is at least 2 in the worst case.

Lemma 1. *The number of misses M of any deterministic algorithm for exploring an $LS(a \ge 1, k, x)$ is at least 2 in the worst case.*

Proof. Similar to the approach by Fischer and Hungerländer [25], consider an $LS(a = 1, k = 4, x = 3)$ as Figure 6(a). For visiting cell (R_0, C_3), any deterministic algorithm must visit cell (R_1, C_3) twice. Simply, checking all the drawing patterns for exploring the rest of LS leads to at least one miss for each area. So, M is at least 2 in the worst case. □

In the same way, exploration patterns can be presented for a given 2-LS in which a and b are bigger than 1 and the number of misses for each pattern is optimal. For instance, consider two special cases *2-LS*$(a = 2o, k = 2o, x = 2o, b = 2o)$ and *2-LS*$(a = 2o, k = 2o+1, x = 2o, b = 2o)$ such that $o \in N$ which can be explored without any misses (see Figure 7).

The other cases of *2-LS*$(a \ge 2, k, x, b \ge 2)$ can be explored optimally by simply following the mentioned patterns in which the number of misses is at most 2. Also, in a similar way, one can prove that as for Lemma 1, any algorithm that explores a *2-LS* in which

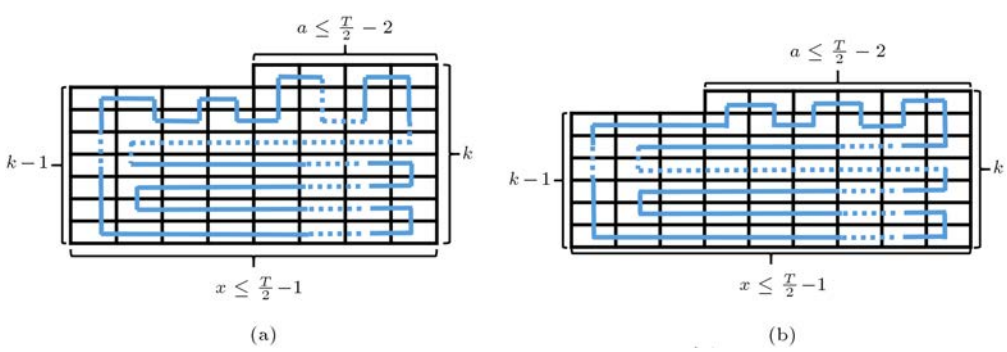

Figure 5. Optimal patterns for exploring: (a) $LS(a = 2o, k = 2o+1, x = 2o+1)$ and (b) $LS(a = 2o, k = 2o, x = 2o)$.

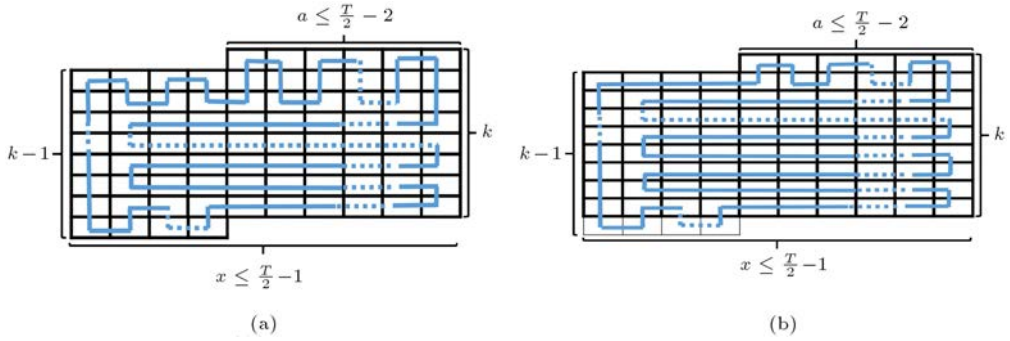

Figure 7. Optimal patterns for exploring (a) *2-LS*$(a = 2o, k = 2o + 1, x = 2o, b = 2o)$ and (b) *2-LS*$(a = 2o, k = 2o, x = 2o, b = 2o)$.

a and b are equal to 1 must have at least 3 misses in the worst case. This leads to the following lemma.

Lemma 2. *Any deterministic algorithm has at least 3 misses for exploring a* $2 - LS(a \geq 1, k, x, b \geq 1)$ *in the worst case.*

Proof. The lemma is correct due to the optimal exploration pattern for the example *2-LS*$(a = 1, k = 5, x = 3, b = 1)$, as shown in Figure 6(b). □

Here, a strategy for exploring $R(x < T/2, y < T/2)$ by subdividing R into *LS*s and *2-LS*s is presented (see Figure 1). Each dashed closed walk shows the pattern of exploration of LSs and 2-LSs.

The following approaches, denoted by LEP, that stand for LS exploration pattern can be obtained:

1. If x or y are even. Without loss of generality, assume x is even. In this strategy, the third parameter of LSs and 2-LSs is equal to the second parameter of R.

   Suppose that R is explored by LSs and 2-LSs as proposed in Figures 5 and 7 from top to bottom. As mentioned before, these drawing patterns explore the environment without any misses. The exploring continues until it is possible to use the LSs or 2-LSs. In other words, R is explored by these LSs and 2-LSs until the total number of remaining unexplored cells, denoted by C', is less than T. In this case, one robot is used for exploring C'. Thus, the number of misses in this case is less than T which is optimal (see Figure 8);
2. If both x and y are odd, R is subdivided into $R(x, 5)$ and $R(x, y - 5)$. Suppose $R(x, 5)$ is explored as mentioned in the Appendix, in Section A.2 from the top.

   If the remaining rows of $R(x, 5)$, denoted by r, that are not explored, are more than or equal to $T/10 + 4/5$, then the remaining rows r and $R(x, y - 5)$ are completely explored as mentioned in the Appendix, in Section A.2 and the previous item 1, respectively. In fact, in this case, by measuring the remaining cells, it can be found that this threshold guarantees that M is less than T. Otherwise, in order to prevent unnecessary misses, C' is combined with an LS or 2-LS that is on top of C', denoted by C'' (see Figure 9). Hence, according to $r < T/10 + 4/5$, the remaining rows r and unexplored cells $R(x, y - 5)$ contain at most $2T - 4$ cells. Therefore, two robots at most are required to explore this area. Imagine one is exploring C" from the right side to left side as drawing pattern shown in Figure 10. Obviously, 3 misses may occur in this part of C" in the worst case which are shown as grey cells (see Figure 10). Also, the remaining cells of C'' can be explored by an *LS* and the number of the corresponding misses is at most one. So, the total number of misses is 4 in the worst case which

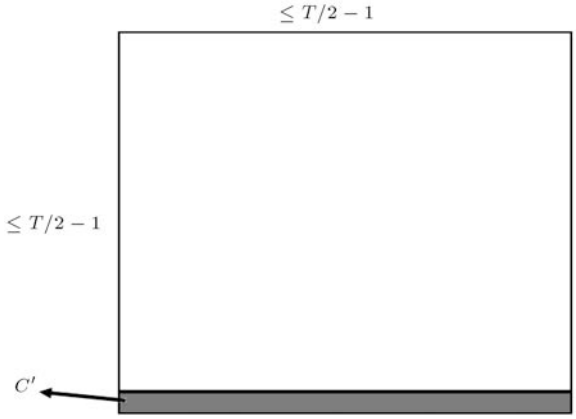

Figure 8. C' explored optimally.

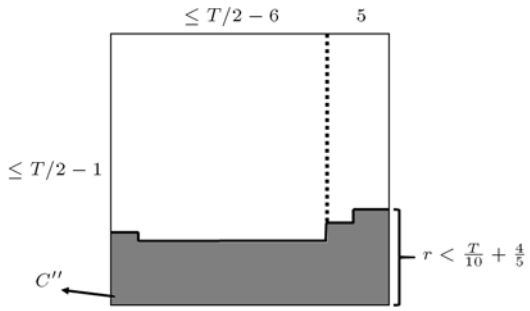

Figure 9. The area C'' including at most $2T - 4$ cells.

Input: T, $R(x, y)$, where $x, y < T/2$.

Output: The number of robots nrb and the exploration pattern $EP = \{EP_1, ..., EP_{nrb}\}$.

1: **if** x or y is even **then**
2: As Fig. 1, subdivide R from its top into LSs and $2 - LS$s, until the remaining cells are less than T;
3: Explore LSs and $2 - LS$s as Fig. 6a and 6b, respectively;
4: Explore the remaining cells by one robot;
5: **else**
6: Subdivide $R(x, y)$ into $R(x, y - 5)$ and $R(x, 5)$;
7: Subdivide $R(x, y - 5)$ as the previous case. C' is the remaining cells;
8: Subdivide $R(x, 5)$ as mentioned in Section 6.2. The number of remaining unexplored rows is denoted by r;
9: **if** $r \geq T/10 + 4/5$ **then**
10: Explore the remaining cells $R(x, y - 5)$ and $R(x, 5)$ as mentioned in Section 6.2;
11: **else**
12: Combine C' with an *LS* or *2-LS* that is on top of it;
13: Combine C' and r and explore them with two robots;
14: **end if**
15: **end if**

Algorithm 1. LEP L-shaped exploration pattern.

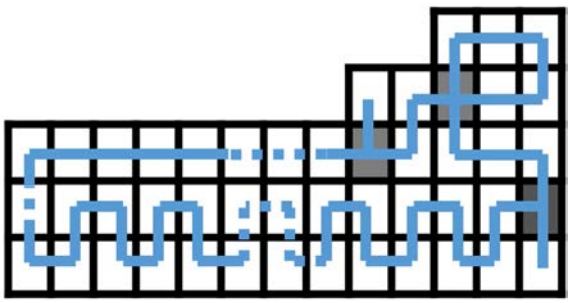

Figure 10. Drawing pattern for C''.

is optimal. Recall that C'' may include one cell. In this case, the total number of misses is at most $T - 1$.

The pseudocode of L-Shaped Exploration Pattern (LEP) algorithm is given as Algorithm 1.

Theorem 2. *When $R(x < T/2, y < T/2)$ is explored by LEP strategy, the number of misses is less than T in the worst case.*

Proof. According to LEP strategy, in each possible case, the number of misses is less than T in the worst case. □

Consider a BLS as an $LS(a, k, x)$ such that the number of columns of its first b' rows is equal to $a > 1$ and $b' \geq 1$. Also, the total number of its cells is more than or equal to T (see Figure 11). Also, LEP can be applied to exploring the *BLS* environments.

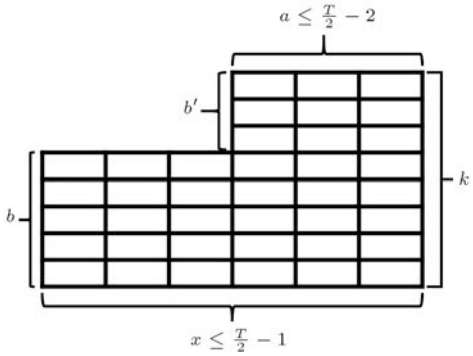

Figure 11. An arbitrary BLS.

Theorem 3. *M in exploring a BLS is less than T using LEP at the maximum time T.*

Proof. This theorem can be proved by following the proof of Theorem 2 quite closely. □

3.2. Exploring $R(n, m)$

In the previous subsection, LEP strategy was described for exploring RGE, where $n, m < T/2$. Now, by

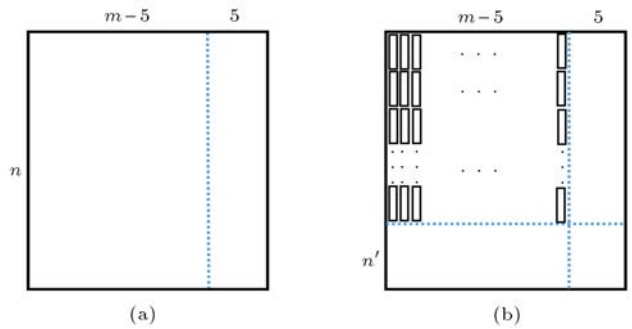

Figure 12. (a) Subdividing $R(n, m)$ into two environments. (b) Exploring the even rectangle $R(n - n', m - 5)$, $n' < T/2$.

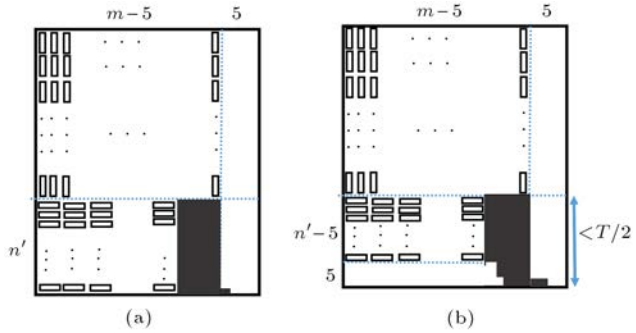

Figure 13. (a) n' is even. (b) n' is odd.

considering these methods, exploring a general size environment can be stated. Clearly, n, m, and T have considerable influence on the drawing exploration pattern of RGE. Hence, the concepts and techniques are covered which, as believed, should be presented in any approach to exploration of RGE. When more material can be covered, all exploration patterns can be detected from the remaining section. Indeed, RGE is considered in which n and m are odd, which is the difficult case, and an optimal algorithm is provided for this case. Of course, optimal approaches are applied to the other cases when n and/or m are even. However, they are left to the reader which are not difficult as the case will be presented here.

In this approach, called REP (stands for rectangular exploration pattern), $R(n, m)$ is subdivided into two RGE $R(n, m-5)$ and $R(n, 5)$ (Figure 12(a)). Based on the Appendix, Section A.2, $R(n, 5)$ and $R(n, m-5)$ are explored as follows. Since $m - 5$ is even, this area is partitioned into blocks of size $2 \times T/2$ (represents a $R(T/2, 2)$ environment) and the idea is to have cells that fit exactly into one robot. In other words, in exploring $R(n, m - 5)$, imagine each robot has to visit $2 \times T/2$ blocks vertically from its top to down until the remaining number of unexplored rows is less than $T/2$, which is denoted by n' (see Figure 12(b)). The exploration of $R(n', m-5)$ is dependent on n' and $T/2$. This leads us to examine all cases as follows:

1. n' is even. In this case, $R(n', m - 5)$ is explored in the same manner as $R(n, m - 5)$. In the worst-case scenario, only one rectangle block remains, denoted by $x \times y$. It is worth noting that both x and y are less than $T/2$. This block will link up to the two unexplored cells from $R(5, n)$ to form a BLS (see Figure 13(a)). The white area is visited without occurrence of any misses. Recall that the grey area is a *BLS*. According to Theorem 3, $T - 1$ misses occur in the worst case;
2. n' is odd:
   (a) If $n' \geq 5$, then $R(n', m - 5)$ is subdivided into $R(5, m-5)$ and $R(n'-5, m-5)$. So, $R(5, m-5)$ is explored as mentioned before. Also, the same strategy used for exploring $R(n, m - 5)$ can be used for assigning the robots to $R(n'-5, m-5)$ (see Figure 13(b)). The row and column widths of grey area are less than $T/2$. This area is considered as a BLS and based on Theorem 3, M is less than T;
   (b) If $n' \leq 3$ and $R(n', m - 5)$ is combined with $T/2 \times (m - 5)$ blocks from the top of it, then follow the previous methods for exploring $R(n - n' - T/2, m - 5)$.

So, the description of REP is as shown in Algorithm 2.

Finally, as mentioned, in all the cases, the number of misses is less than T, so according to Observation 1, $nrb = nrb_{opt}$.

4. Exploring $R(n, m)$ in the presence of obstacles

In this section, we consider the Rectangular Grid Exploration problem in the presence of Obstacles (RGEO) with a predefined time T. Under the widely believed assumption that $NP \neq P$, this section starts by giving a lemma in order to prove that there is no algorithm for solving RGEO at polynomial time to prevent unnecessary robots at the predefined time T. Then, an algorithm that can approximate the problem within a factor of 4 is presented.

Notice that the exploration problem plays a similar role in the hardness of approximation as the Hamiltonian cycle problem plays in the theory of NP completeness. One result of this section is:

Lemma 3. *The approximation factor of any deterministic algorithm for RGEO is at least 2.*

Proof. Suppose that T is equal to the number of free cells in RGEO. In this case, only one robot can explore the general grid graph. Hence, a feasible solution to the exploration problem exists if and only if Hamiltonian cycle problem has a feasible solution.

Input: T, $R(n,m)$, where n and m is odd.

Output: The number of robots nrb and the exploration pattern $EP = \{EP_1, EP_2, ..., EP_{nrb}\}$.

1: Subdivide $R(n,m)$ into $R(n,m-5)$ and $R(n,5)$;
2: Subdivide $R(x,5)$ as mentioned in Section 6.2;
3: Partition $R(n,m-5)$ into blocks $2 \times T/2$ form its top to down until the number of unexplored remaining rows is less than $T/2$ which is denoted by $n^{'}$;
4: **if** $n^{'}$ is even and $n^{'} > 3$ **then**
5: Subdivide $R(n^{'},m)$ from its left to right into blocks $2 \times T/2$ until the number of unexplored remaining rows is less than $T/2$;
6: Explore the remaining area, which is BLS, as explained.
7: **else if** $n^{'}$ is odd and $n^{'} > 3$ **then**
8: Subdivide $R(n^{'},m)$ into $R(n^{'},m-5)$ and $R(n^{'},5)$;
9: Use the same strategy for $R(n^{'}-5,m-5)$ as mentioned for $R(n,m-5)$;
10: Consider the remaining area as BLS and explore it as mentioned;
11: **else**
12: Combine $R(n^{'},m-5)$ with $T/2 \times (m-5)$ blocks from $R(n^{'},m-5)$'s top and follow the previous methods for exploring $R(n-n^{'}-T/2,m-5)$.
13: **end if**

Algorithm 2. REP rectangular exploration pattern.

Interestingly enough, the problem of finding the best such lower bound of α-factor on Hamiltonian cycle is intimately related to that of finding a lower bound on the exploring problem. Hence, according to Arora's algorithm [15] and also the number of robots being an integer, the approximation factor of any deterministic algorithm for the exploring problem is 2. □

4.1. Approximation algorithm for RGEO

In this section, a 4-approximation algorithm for RGEO is presented. Recall that the connected general grid graph G corresponds to the grid environment $\mathcal{E}$ (Figure 14(a)). The proposed Algorithm 3, denoted by EAO, is as follows.

Theorem 4. *The algorithm EAO gives a 4-approximation factor for RGEO.*

To prove Theorem 4, EAO is explained in more detail in the following. In other words, EAO receives as input T and $R(n,m)$ (i.e., a planar graph G) in the presence of obstacles and computes a spanning tree S of G (Figure 14(b)). It considers one arbitrary node as the root of S (Figure 14(c)).

Let v_L be the lowest leaf and T be the predefined time. Imagine that EAO explores all vertices from the lowest level until it satisfies the condition of the exploration which means that the robot must return to its starting position within the remaining steps. In other words, suppose that EAO assigns a robot rb_i exploring all the vertices from v_L to p, which are shown by black nodes in Figure 14(c). Let $t_{v_L,p}$ be the number of time steps that rb_i takes to visit all the vertices from

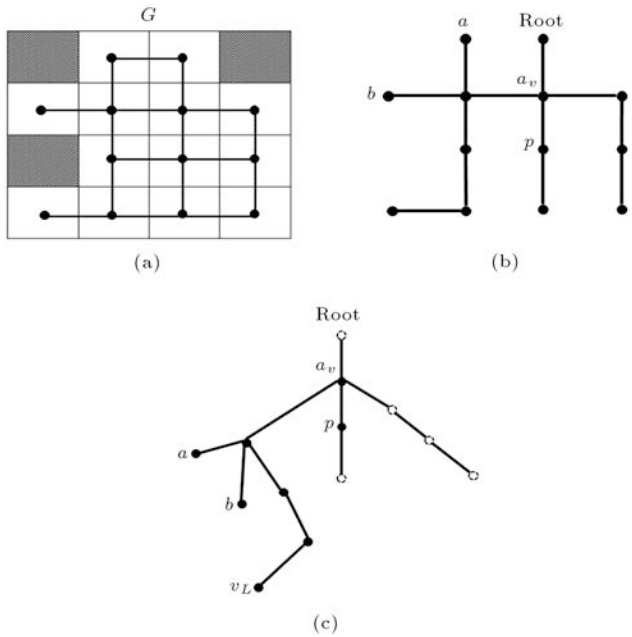

Figure 14. (a) An arbitrary general grid graph. (b) The spanning tree of G. (c) One arbitrary node is considered as the root of S.

Input: T, $R(n,m)$, where n and m is odd.

Output: The number of robots nrb and the exploration pattern $EP = \{EP_1, EP_2, ..., EP_{nrb}\}$.

1: Subdivide $R(n,m)$ into $R(n,m-5)$ and $R(n,5)$;
2: Subdivide $R(x,5)$ as mentioned in Section 6.2;
3: Partition $R(n,m-5)$ into blocks $2 \times T/2$ form its top to down until the number of unexplored remaining rows is less than $T/2$ which is denoted by $n^{'}$;
4: **if** $n^{'}$ is even and $n^{'} > 3$ **then**
5: Subdivide $R(n^{'},m)$ from its left to right into blocks $2 \times T/2$ until the number of unexplored remaining rows is less than $T/2$;
6: Explore the remaining area, which is BLS, as explained.
7: **else if** $n^{'}$ is odd and $n^{'} > 3$ **then**
8: Subdivide $R(n^{'},m)$ into $R(n^{'},m-5)$ and $R(n^{'},5)$;
9: Use the same strategy for $R(n^{'}-5,m-5)$ as mentioned for $R(n,m-5)$;
10: Consider the remaining area as BLS and explore it as mentioned;
11: **else**
12: Combine $R(n^{'},m-5)$ with $T/2 \times (m-5)$ blocks from $R(n^{'},m-5)$'s top and follow the previous methods for exploring $R(n-n^{'}-T/2,m-5)$.
13: **end if**

Algorithm 3. Psudocode for EAO.

v_L to p. Let $v_{v_L,p}$ be the shortest path from v_L to p. So, it is obvious that $|EP_i| = t_{v_L,p} + |v_{v_L,p}|$. This leads to the following equation:

$$T - |EP_i| = 0. \tag{4}$$

To be clearer, $|EP_i|$ is the length of the path that robot rb_i explores at predefined time T. Note that each robot should return to its start position. Given that each robot should visit the most possible vertices, Eq. (4) holds.

Consider the highest ancestor of v_L in EP_i which is shown by a_v (see Figure 14(c)). Since a_v has at most three children, depending on EP_i, three different cases may be distinguished and handled by EAO as follows:

1. Robot rb_i only visits all children of a_v which means it visits $T/2+1$ cells;
2. There are some vertices in the third branch of node a_v which are not visited by rb_i. Then, EAO explores this branch from the lowest leaf to the vertex w which satisfies Eq. (4) by robot rb_{i+1} (see Figure 15). Since the initial position of each robot is the lowest leaf and EAO starts from the lowest vertex and grows until the above condition is satisfied, $|v_{a_v,p}| = EP_i \cap EP_{i+1} \cong T/4$ in the worst case. Also, rb_{i+1} may visit $T/4$ cells above a_v which are visited by another robot. In fact, the height of branches may almost balance in the worst case. Thus, rb_i and rb_{i+1} together visit $T/2+T/2-|v_{a_v,p}|-T/4 > T/2$ cells. It is possible that rb_i and rb_{i+1} visit one cell at the same time. Note that EAO makes collision free path for rb_i and rb_{i+1} by following Lemma 4;
3. Robot rb_i explores the first branch and some cells of the second branch. In this case, rb_{i+1} explores the remaining cells of the second branch and some cells of the third branch, and rb_{i+2} explores the remaining cells of the third branch and some nodes above a_v where these nodes satisfy Eq. (4) (see Figure 16). Based on Lemma 4, there exists a collision-free path for rb_i and rb_{i+1}, so does, for rb_{i+1} and rb_{i+2}. As the previous case, the intersection of the two exploration paths that have

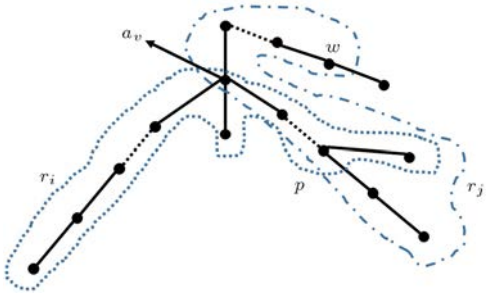

Figure 15. The size of $EP_i \cap EP_{i+1} \cong T/4$.

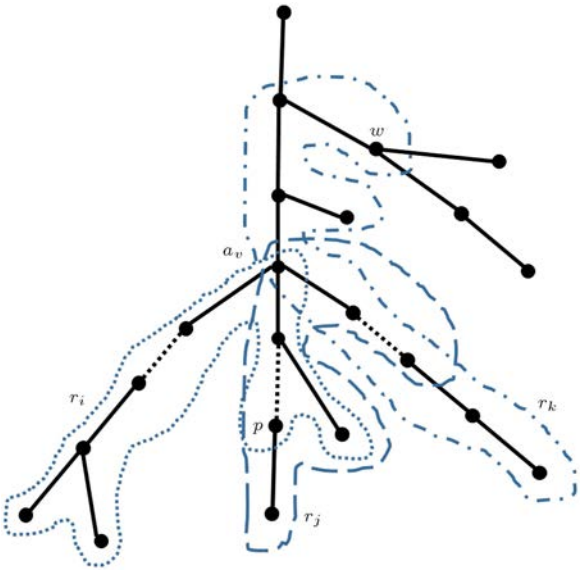

Figure 16. r_{i+2} visiting the vertices above a_v.

a node a_v in common is $T/4$ cells in the worst case. Since $EP_i \cap EP_{i+1} = T/4$, $EP_{i+1} \cap EP_{i+2} = T/4$, and $EP_{i+2} \cap EP_i$ are constant, the number of cells explored by rb_i and rb_{i+1} is equal to $T/2$ in the worst case.

Lemma 4. *Consider EP_i and EP_{i+1} where $EP_i \cap EP_{i+1} = c$. Suppose that rb_i and rb_{i+1} visit cell c at the same time within their exploration paths. Then, there exists at least one collision-free path for rb_i and rb_{i+1}.*

Proof. Suppose that c is visited by rb_i at time t_1^i and t_2^i and by rb_{i+1} at time t_1^{i+1} and t_2^{i+1}. Also, its initial position and destination denoted by $c_{1i}(c_{1i+1})$ are identical, since the exploration path for $rb_i(rb_{i+1})$ is a cycle. If rb_i and rb_{i+1} have intersection cell c, then it is possible to change the initial path of $rb_i(rb_{i+1})$ to another arbitrary cell as follows:

1. If $t_2^i = t_2^{i+1}$ and $t_1^i = t_1^{i+1}$, or $|t_2^i - t_2^{i+1}| \geq 2$ and $t_2^i \neq t_2^{i+1}$ and $t_1^i = t_1^{i+1}$, or $(|t_1^i - t_1^{i+1}| \geq 2)$ and $t_1^i \neq t_1^{i+1}$ and $t_2^i = t_2^{i+1}$;
2. If $|t_2^i - t_2^{i+1}| = 1$ and $t_2^i \neq t_2^{i+1}$ and $t_1^i = t_1^{i+1}$, or $|t_1^i - t_1^{i+1}| = 1$ and $t_1^i \neq t_1^{i+1}$ and $t_2^i = t_2^{i+1}$.

   Changing the initial position of the robot rb_i from c_{1i} to c_{2i} in which $t_1^i + t_2^i$ is minimum.

Based on the above descriptions, changing the initial position of $rb_i(rb_{i+1})$ makes a collision-free path for both of robots.

Proof of Theorem 4. In the algorithm EAO, in the worst case, each of two neighbor robots visits $T/2$ different vertices together, while at most $2T$ different cells can be explored by two robots in optimal algorithm. So, $\alpha = 2T/(T/2) = 4$. □

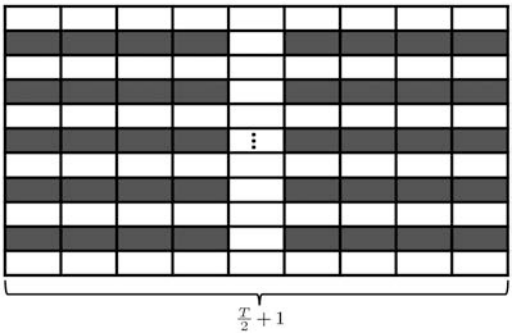

Figure 17. A tight example on a spanning tree of an arbitrary Rectangle Grid Environmental (RGE).

A tight example for any approximation algorithm is given by an arbitrary *RGE* with $T/2+1$ columns that is tree (see Figure 17). In fact, a simple inspection of Figure 17 demonstrates that each of the two robots visits at most $T/2$ cells in the best case. Since an arbitrary spanning tree of RGE is considered as an input of any approximation algorithm, factor of 4 is the lower bound in the worst case.

5. Conclusion and future work

This study investigated the *multi-robot exploration* problem in rectangle grid graphs at a predefined time T. In the absence of obstacles, an optimal algorithm to solve this problem was proposed. This problem was also studied in the presence of obstacles. According to the Hamiltonian cycle problem, it can be concluded that there is no algorithm better than 2-approximation factor for this version of the problem. Finally, a 4-approximation algorithm was proposed.

In this study, it was shown that factor 4 was the best lower bound for any deterministic algorithm on an arbitrary spanning tree of RGE. However, giving a good approximation algorithm for an arbitrary subgraph of RGE, which is not a tree, remains a clue for future work. In addition, it is noticed that exploring RGE in the presence of obstacles with the minimum number of robots is NP-complete. As a future research direction, it would be interesting to study this problem where the input is some special figure of the grid environment such as polygon, alphabet grid, etc.

References

1. Walenz, B. "Multi robot coverage and exploration: a survey of existing techniques" (2016).
2. Blatt, F. and Szczerbicka, H. "Combining the multi-agent flood algorithm with frontier-based exploration in search & rescue applications", *International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), IEEE*, pp. 1–7 (2017).

3. Davoodi, M., Abedin, M., Banyassady, B., Khanteimouri, P., and Mohades, A. "An optimal algorithm for two robots path planning problem on the grid", *Robotics and Autonomous Systems*, **61**(12). pp. 1406–1414 (2013).

4. Davoodi, M. "Bi-objective path planning using deterministic algorithms", *Robotics and Autonomous Systems*, **93**, pp. 105–115 (2003).

5. Davoodi, M., Panahi, F., Mohades, A., and Hashemi, S.N. "Multi-objective path planning in discrete space", *Applied Soft Computing*, **13**(1), pp. 709–720 (2013).

6. Haynes, P.S., Alboul, L., and Penders, J. "Dynamic graph-based search in unknown environments", *Journal of Discrete Algorithms*, **12**, pp. 2–13 (2012).

7. Dereniowski, D., Disser, Y., Kosowski, A., Pajkak, D., and Uznański, P. "Fast collaborative graph exploration", *Information and Computation*, **243**, pp. 37–49 (2015).

8. Davoodi, M., Panahi, F., Mohades, A., and Hashemi, S.N. "Clear and smooth path planning", *Applied Soft Computing*, **32**, pp. 568–579 (2015).

9. Chen, C.Y. and Ko, C.C. "An evolutionary method to vision-based self-localization for soccer robots", *Scientia Iranica, Transaction B, Mechanical Engineering*, **22**(6), pp. 2071–2080 (2015).

10. Bahar, M.R.B., Ghiasi, A.R., and Bahar, H.B. "Grid roadmap based ANN corridor search for collision free, path planning", *Scientia Iranica*, **19**(6), pp. 1850–1855 (2012).

11. Kumar, P.B., Sahu, C., Parhi, D.R., Pandey, K.K., and Chhotray, A. "Static and dynamic path planning of humanoids using an advanced regression controller", *Scientia Iranica*, **26**(1), pp. 375–393 (2019).

12. Book, R.V., Michael, R.G., and David, S.J. "Computers and intractability: A guide to the theory of NP-completeness", *Bulletin (New Series) of the American Mathematical Society*, **3**(2), pp. 898–904 (1980).

13. Christofides, N. "Worst-case analysis of a new heuristic for the travelling salesman problem", *Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group* (1976).

14. Rigni, M., Koutsoupias, E., and Papadimitriou, C. "An approximation scheme for planar graph TSP", *Proceedings of IEEE 36th Annual Foundations of Computer Science*, **11**(4), pp. 640–645 (1995).

15. Arora, S. "Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems", *ACM*, **45**(5), pp. 753–782 (1998).

16. Griffith, J.C., *A Meta-Algorithm Analysis of the Traveling Salesman Problem Using Cluster-Analysis and Algorithm Recommendation*, Bradley University (2013).

17. Arkin, E.M., Fekete, S.P., and Mitchell, J.S. "Approximation algorithms for lawn mowing and milling", *Elsevier*, **17**(1–2), pp. 25–50 (2000).

18. Ntafos, S. "Watchman routes under limited visibility", *Elsevier*, **1**(3), pp. 149–170 (1992).

19. Wernli, D. "Grid exploration", Master Thesis, ETH Zurich, Department of Computer Science (2012).

20. Arkin, E.M., Bender, M.A., Demaine, E.D., Fekete, S.P., Mitchell, J.S., and Sethia, S. "Optimal covering tours with turn costs", *SIAM*, **35**(3), pp. 531–566 (2005).

21. Pajak, D., *Algorithms for Deterministic Parallel Graph Exploration*, Université Sciences et Technologies-Bordeaux I (2014).

22. Umans, C. and Lenhart, W. "Hamiltonian cycles in solid grid graphs", *IEEE*, **11**(4), pp. 496–505 (1997).

23. Itai, A., Papadimitriou, C.H., and Szwarcfiter, J.L. "Hamilton paths in grid graphs," *SIAM*, **11**(4), pp. 676–686 (1982).

24. Salman, A.N.M., Baskoro, E.T., and Broersma, H.J. "A note concerning Hamilton cycles in some classes of grid graphs", *ACM*, **35**(1), pp. 65–70 (2003).

25. Fischer, A. and Hungerländer, P. "The traveling salesman problem on grids with forbidden neighborhoods", *Journal of Combinatorial Optimization*, **34**(3), pp. 891–915 (2017).

26. Davoodi, M., Ghadikolaei, M.K., and Malekizadeh, M.M. "Exploring rectangular grid environments", *1st Iranian Conference on Computational Geometry*, Tehran, Iran (2018).

27. Senarathne, P.G.C.N. and Wang, D. "Incremental algorithms for safe and reachable Frontier detection for robot exploration", *Robotics and Autonomous Systems*, **72**, pp. 189–206 (2015).

28. Gabriely, Y., and Rimon, E. "Spanning-tree based coverage of continuous areas by a mobile robot", *Annals of Mathematics and Artificial, Intelligence*, **31**(1–4), pp. 77–98 (2001).

29. Czyzowicz, J., Dereniowski, D., Gkasieniec, L., Klasing, R., Kosowski, A., and Pajkak, D. "Collision-free network exploration", *Journal of Computer and System Sciences*, **86**, pp. 70–81 (2017).

30. Disser, Y., Mousset, F., Noever, A., Škorić, N., and Steger, A. "A general lower bound for collaborative tree exploration", *International Colloquium on Structural Information and Communication Complexity*, Springer, Cham, pp. 125–139 (2017).

31. Bampas, E., Chalopin, J., Das, S., Hackfeld, J., and Karousatou, C. "Maximal exploration of trees with energy-constrained agents", *arXiv preprint arXiv:1802.06636* (2018).

Appendix

In this section, some small cases are considered to explore the entire RGE. Thus, different basic notions and techniques are introduced in order to explore some special environments here.

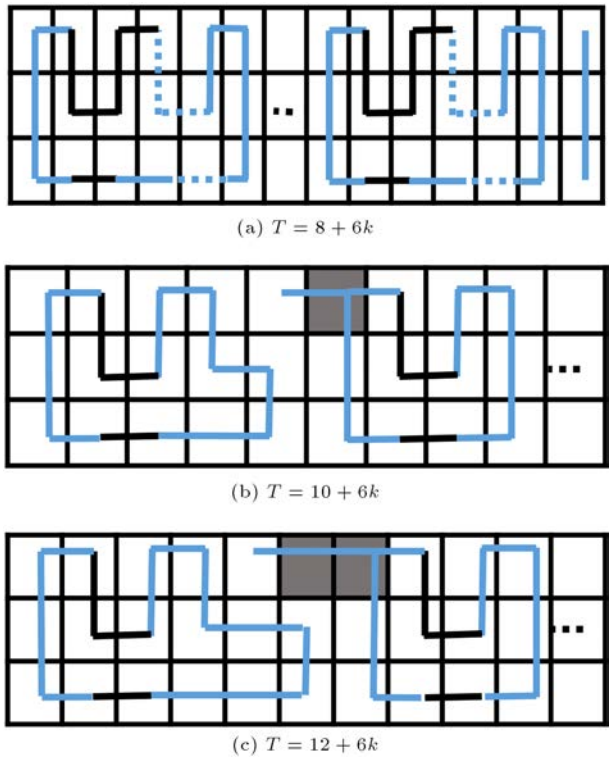

Figure A.1. Optimal patterns for $R(n, 3)$.

Exploring $R(n, 3)$

Here, an optimal approach, denoted by TRE, is described for exploring $R(n, 3)$. It is clear that T plays a crucial role in exploring an arbitrary environment. Consider three different classes $T = 6k+8$, $T = 6k+10$, and $T = 6k + 12$ ($k = 0, 1, \cdots$). Different methods are presented to visit all cells with the optimal number of robots for each class (see Figure A.1).

Consider the area of f_i that is explored by rb_i. The number of row widths of each f_i is denoted by h_i. The presented drawing patterns are shown in Figure A.1(b) and (c) for the case $h_i > T/3$. The number of misses is equal to 1 for each of two neighboring robots when $T = 6k+8$, in which the thick lines are repeated k times (Figure A.1(b)). The presented drawing pattern is shown in Figure 18(c) for the case $T = 6k + 10$. The number of misses is equal to 2 for each of two neighboring robots. The rest of this subsection shows that the proposed approach is optimal.

Observation A.1. *Since $T/3$ is not an integer, if h_i is less than or equal to $T/3$, then there is at least one miss that occurs in exploring f_i by rb_i in the best case by any optimal algorithm.*

Let us suppose that h_i is more than $T/3$. Also, let the first, second, and third columns of $R(n, 3)$ be denoted by C_1, C_2, and C_3, respectively. Clearly, the number of row widths of C_1, C_2, and C_3 of f_i can influence the number of misses that occur during exploration. In fact, a few columns can make a good piece of evidence that neighboring robots may influence each other and determine the number of misses in the environment. This is illustrated in Figure A.2, drawing pattern of f_i makes a corridor in C_3. Then, there are some cells that the neighboring robot rb_j of rb_i must visit more than once. Let rb_j be the *complementary* robot of rb_i. In this case, one would like to avoid having a corridor as possible to reduce the number of misses.

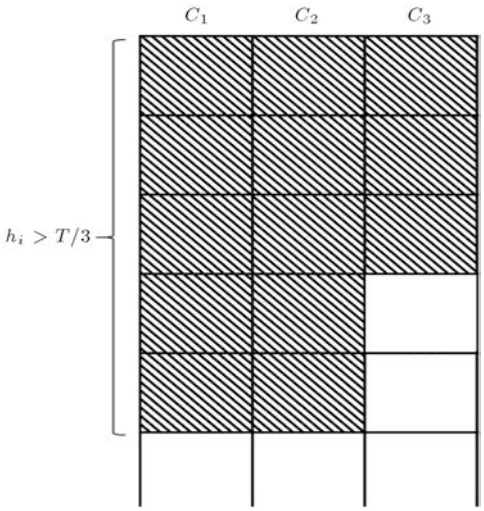

Figure A.2. f_i with $h_i > \lfloor T/3 \rfloor$.

Lemma A.1. *Given $R(n, 3)$, suppose that h_i is more than $T/3$ for each arbitrary robot rb_i. Let rb_i and rb_j be two neighboring robots; rb_j is the complementary of rb_i. Then, $m_i + m_j$ for any deterministic algorithm is equal to the number of misses in TRE strategy in the worst case.*

Proof. For each arbitrary rb_i, since h_i is more than $T/3$, it makes a corridor to one of $C_i (i = 1, 2, 3)$ that is consisting of at least one cell, denoted by C'_i. In the case, the size of C'_i equals 1, at least one miss occurs for rb_j when crossing the corridor C'_i. Hence, the number of misses in the case $T = 6k + 8$ is optimal in TRE. It remains to prove that TRE explores the environment optimally in the other case. Suppose that $T = 10$; by considering all the cases, there are at least two misses that occur for two complementary robots in $R(n, 3)$. By induction, the number of misses is handled correctly, meaning that M is optimal, in this case as well. □

A.2. Exploring $R(n, 5)$

In this section, a method is introduced to explore $R(n, 5)$ optimally, denoted by FRE. Consider different five classes $T_0 = 10k$, $T_1 = 10k + 12$, $T_2 = 10k + 14$, $T_3 = 10k + 16$, and $T_4 = 10k + 18$ ($k = 0, 1, \cdots$).

Different optimal drawing patterns for each class are provided in which thick pattern lines are repeated k times (see Figure A.3). According to these exploration

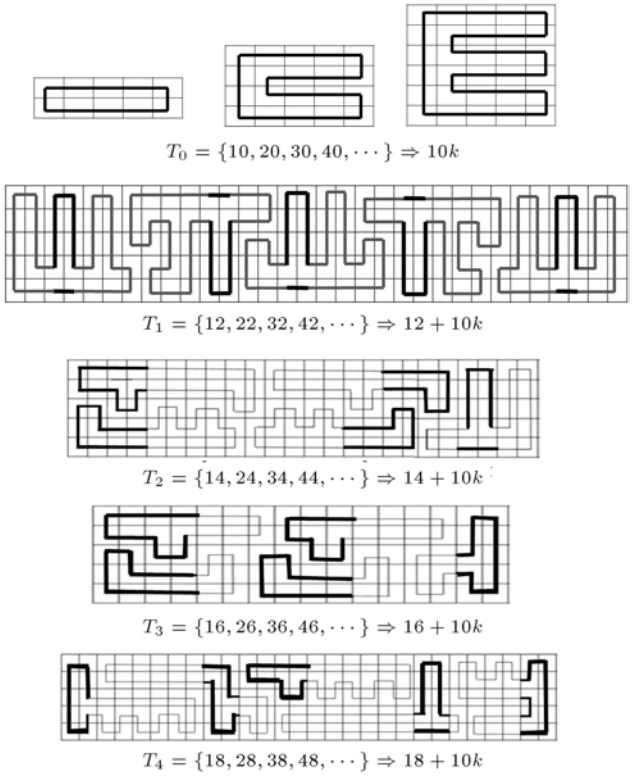

Figure A.3. The presented patterns for $R(n, 5)$.

patterns, no miss occurs for the robots except for the last robot in case perhaps there is at least one cell left to explore at the end of the environment in the worst case. Since one robot must be assigned to the remaining cells, there are $T - 1$ misses in the worst case. Then, based on Observation 1, the proposed method is optimal.

Also, one should restrict attention to a special case where $T = 8$. Imagine $R(n, 5)$ is explored from top to down. Various approaches are inspected for exploring $R(n, 5)$ in this case and realize that at least one miss occurs on the top of the environment (see Figure A.4). The following lemma can be obtained:

Lemma A.2. *For $T = 8$, there is at least one miss for exploring the top of $R(n, 5)$.*

Proof. By inspection. □

Lemma A.3. *FRE explores $R(n, 5)$ optimally.*

Proof. Based on Lemma A.2 and extending drawing pattern of Figure A.4, *FRE* gives an optimal pattern to explore $R(n, 5)$. □

Biographies

Mansoor Davoodi received his BS degree in Computer Science from Vali Asr University, Rafsanjan, Iran in 2005 and his MS and PhD degrees in Computer Science in Amirkabir University of Technology (Tehran Polytechnic), Tehran in 2007 and 2012, respectively. He is currently an Assistant Professor at the Department of Computer Science and Information Technology, Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan, Iran. His main research areas include algorithms, computational geometry, imprecise data handling, and multi-objective optimization.

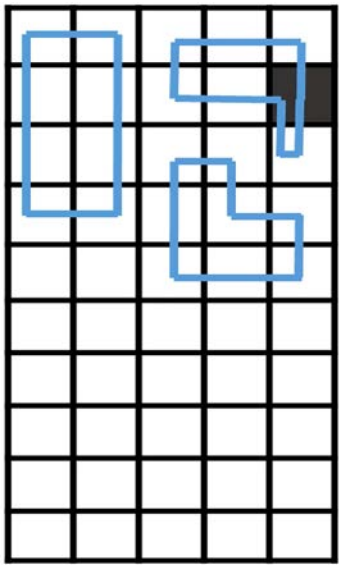

Figure A.4. An optimal approach to exploring the top of $R(n, 5)$, for $T = 8$.

Esmaeil Delfaraz is a PhD student of Computer Science, Gran Sasso Science Institute (GSSI), L'Aquila, Italy. He got his MSc degree in Computer Science from Institute for Advanced Studies in Basic Sciences, Zanjan in 2017. Further, he received his BSc degree in the same field from Shahid Bahonar University of Kerman, Kerman in 2014. His research interests include online algorithms, computational geometry, wireless sensor networks, approximation, and randomized algorithms.

Sajjad Ghobadi is a PhD student in Computer Science, Gran Sasso Science Institute (GSSI), L'Aquila, Italy. He got his MSc degree in Computer Science from Institute for Advanced Studies in Basic Sciences, Zanjan, Iran in 2017 and his BSc degree in Computer Science from the University of Tabriz, Tabriz in 2014. His current research interests are algorithms, wireless sensor networks, and computational geometry.

Mahtab Masoori is a PhD student in Computer Science, Concordia University, Montreal, Canada. She obtained her MSc degree in Computer Science at the Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan, Iran in 2017. Her research interests lie in algorithms containing online, approximation, and randomized algorithms.