

Advice Complexity of Treasure Hunt in Geometric Terrains

Andrzej Pelc ^{*†}

Ram Narayan Yadav [‡]

Abstract

Treasure hunt is the task of finding an inert target by a mobile agent in an unknown environment. We consider treasure hunt in geometric terrains with obstacles. Both the terrain and the obstacles are modeled as polygons and both the agent and the treasure are modeled as points. The agent navigates in the terrain, avoiding obstacles, and finds the treasure when there is a segment of length at most 1 between them, unobstructed by the boundary of the terrain or by the obstacles. The cost of finding the treasure is the length of the trajectory of the agent. We investigate the amount of information that the agent needs *a priori* in order to find the treasure at cost $O(L)$, where L is the length of a shortest path in the terrain from the initial position of the agent to the treasure, avoiding obstacles. Following the well-established paradigm of *algorithms with advice*, this information is given to the agent in advance as a binary string, by an oracle cooperating with the agent and knowing the whole environment: in our case, the terrain, the position of the treasure and the initial position of the agent. Advice complexity of treasure hunt is the minimum length of the advice string (up to multiplicative constants) that enables the agent to find the treasure at cost $O(L)$.

We first consider treasure hunt in *regular* terrains which are defined as convex polygons with convex c -fat obstacles, for some constant $c > 1$. A polygon is c -fat if the ratio of the radius of the smallest disc containing it to the radius of the largest disc contained in it is at most c . For the class of regular terrains, we establish the exact advice complexity of treasure hunt. We then show that advice complexity of treasure hunt for the class of arbitrary terrains (even for non-convex polygons without obstacles, and even for those with only horizontal or vertical sides) is exponentially larger than for regular terrains.

keywords: mobile robot, treasure hunt, polygon, obstacle.

*Département d'informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada. E-mail: pelc@uqo.ca

†Partially supported by NSERC discovery grant 2018-03899 and by the Research Chair in Distributed Computing at the Université du Québec en Outaouais.

‡Department of Computer Science and Engineering, Indian Institute of Information Technology, Dharwad, India. E-mail: narayanram1988@gmail.com

1 Introduction

1.1 The background and the problem

Treasure hunt is the task of finding an inert target by a mobile agent in an unknown environment. We consider treasure hunt in geometric terrains with obstacles. This task has important applications when the terrain is dangerous or difficult to access for humans. Rescuing operations in mines contaminated or submerged by water are an example of situations where a lost miner is a target that has to be found fast by a mobile robot, and hence the length of the robot’s trajectory should be as short as possible.

We model the treasure hunt problem as follows. The terrain is represented by an arbitrary polygon \mathcal{P}_0 with pairwise disjoint polygonal obstacles $\mathcal{P}_1, \dots, \mathcal{P}_k$, included in the interior of \mathcal{P}_0 , i.e., the terrain is $\mathcal{T} = \mathcal{P}_0 \setminus (\mathcal{P}_1 \cup \dots \cup \mathcal{P}_k)$. We assume that the polygon \mathcal{P}_0 is closed (i.e., contains its boundary) and the polygons $\mathcal{P}_1, \dots, \mathcal{P}_k$ are open (i.e., do not contain their boundaries). In this way the terrain \mathcal{T} is a closed subset of the plane, i.e., it contains its entire boundary, which is the union of boundaries of all polygons. It should be noted that the restriction to polygons is only to simplify the description, and all our results can be generalized to the case where polygons are replaced by compact subsets of the plane homeotopic with a disc (i.e., without holes) and regular enough to have well-defined boundary length. The treasure is modeled as an inert interior point of the terrain.

The mobile agent (robot) is modeled as a point starting inside the terrain and moving along a polygonal line inside it. It is equipped with a compass and a unit of length, and we assume that it has unbounded memory: from the computational point of view the agent is a Turing machine. The moves of the agent are of two types: *free* moves and *boundary* moves. A free move is a move of the agent in the terrain along a segment of a chosen length in a chosen direction. Such a move may be interrupted if the agent hits the boundary of the terrain during its execution, and the agent becomes aware of this interruption. A boundary move is executed by an agent located on the boundary of the terrain. Such a move is of the form: follow the boundary that you are on (this can be the boundary of any of the polygons \mathcal{P}_i) in a chosen direction (there are two possible directions), either at a chosen distance or until getting to a point with a given property.

The aim of treasure hunt is for the agent to *see* the treasure. We assume that the agent currently located at a point p of the terrain sees all points q for which the segment pq is entirely contained in \mathcal{T} and is of length at most 1. The cost of a treasure hunt algorithm on an instance is the length of the trajectory of the agent from its initial position until it sees the treasure. We assume that the agent does not know the terrain nor the location of the treasure before starting treasure hunt.

We investigate the amount of information that the agent needs *a priori* in order to find the treasure at cost $O(L)$, where L is the length of a shortest path in the terrain from the initial position of the agent to the treasure. * (Since the agent sees at distance 1, we have $C \leq L \leq C + 1$, where C is the optimal cost of treasure hunt with full knowledge). Following the well-established paradigm of *algorithms with advice* (see the subsection “Related work”), this information is given to the

*Since we use the O -notation for functions with positive real values that can be smaller than 1, it is important to give a precise definition that we will use: For two functions $f, g : \mathcal{R}^+ \rightarrow \mathcal{R}^+$, we say that $f(x)$ is in $O(g(x))$ if there exists a positive constant c such that $f(x) \leq cg(x)$, for all $x \in \mathcal{R}^+$.

agent in advance as a binary string, by an oracle cooperating with the agent and knowing the whole environment: in our case, the terrain, the position of the treasure and the initial position of the agent. Advice complexity of treasure hunt is the minimum length of the advice string (up to multiplicative constants) that enables the agent to find the treasure at cost $O(L)$. Advice complexity of a task can be considered to be a measure of its difficulty. Hence our aim is to estimate the difficulty of treasure hunt in geometric terrains. It is well known that many algorithmic tasks become feasible or easier, when the algorithm is supplied with a particular item of information, such as the size or diameter of the graph. However, the paradigm of algorithms with advice permits us to establish the minimum size of the information needed, regardless of its nature. Hence the measure of advice complexity is a quantitative approach to the knowledge provided to the algorithm, as opposed to the qualitative approach, studying the impact of knowing particular items of information, such as various numerical parameters of the problem. To the best of our knowledge, this is the first time that the advice complexity approach is applied to the problem of treasure hunt in the geometric setting.

Coming back to our application in the context of a miner lost in the mine, advice complexity of treasure hunt may be crucial. The miner knows the terrain, knows his/her position and knows the entrance to the mine. How to text as little information as possible to the rescuing team (time is precious) to allow a robot to reach the miner fast?

In order to formulate our results, we define the following parameter λ called the *accessibility* of the treasure. $\lambda = \min(1/2, \rho)$, where ρ is the largest radius, such that some disc with radius ρ contains the treasure and is contained in the terrain. Since the treasure is located in an interior point of the terrain, we have $\lambda > 0$. By definition, any disc of radius λ containing the treasure and contained in the terrain has the property that the agent reaching any point of this disc can see the treasure.

1.2 Our results

We first consider treasure hunt in *regular* terrains which are defined as convex polygons with convex c -fat obstacles, for some constant $c > 1$. A polygon is c -fat if the ratio of the radius of the smallest disc containing it to the radius of the largest disc contained in it is at most c . (For example, all regular convex polygons are 2-fat). For the class of regular terrains, we establish the exact advice complexity of treasure hunt. For $L > \lambda$, we provide a treasure hunt algorithm working at cost $O(L)$ for all regular terrains, using advice of size $O(\log(L/\lambda))$, and we construct a class of regular terrains for which there is no treasure hunt algorithm working at cost $O(L)$ with advice of size $o(\log(L/\lambda))$. For $L \leq \lambda$, we construct a treasure hunt algorithm working at cost $O(L)$ for all regular terrains, without any advice.

In order to appreciate the strength of the tightness result for $L > \lambda$, notice that its positive part gives a concrete advice of size $O(\log(L/\lambda))$ (in our case indicating the approximate direction towards the treasure with respect to the initial position of the agent), and a treasure hunt algorithm of cost $O(L)$ for the class of regular terrains, using this advice, while the negative part shows that no advice of size of smaller order, *regardless of its kind and meaning*, can help to accomplish treasure hunt at cost $O(L)$ in all regular terrains.

We then show that advice complexity of treasure hunt for the class of arbitrary terrains (even

for non-convex polygons without obstacles, and even for those with only horizontal or vertical sides) is exponentially larger than for regular terrains. Our negative result is even stronger: we construct terrains with treasure accessibility $1/2$ for which advice complexity of treasure hunt can be a function of L growing arbitrarily fast.

1.3 Related work

Treasure hunt. The problem of searching for a target by one or more mobile agents was investigated under many different scenarios. The environment where the target is hidden may be a graph or a plane, and the search may be deterministic or randomized. The book [3] surveys both the search for a fixed target and the related rendezvous problem, where the target and the searching agent are both mobile and their role is symmetric: they cooperate to meet. This book is concerned mostly with randomized search strategies. In [36, 42] the authors studied relations between the problems of treasure hunt (searching for a fixed target) and rendezvous in graphs. The authors of [4] studied the task of finding a fixed point on the line and in the grid, and initiated the study of the task of searching for an unknown line in the plane. This research was continued, e.g., in [30, 35]. In [41] the authors concentrated on game-theoretic aspects of the situation where multiple selfish pursuers compete to find a target, e.g., in a ring. The main result of [34] is an optimal algorithm to sweep a plane in order to locate an unknown fixed target, where locating means to get the agent originating at point O to a point P such that the target is in the segment OP . In [20] the authors considered the generalization of the search problem in the plane to the case of several searchers. Efficient search for a fixed or a moving target in the plane, under complete ignorance of the searching agent, was studied in [40].

Exploration of terrains. Exploration of unknown terrains by mobile robots is a subject closely related to treasure hunt. A mobile agent has to see all points of the terrain, where seeing a point p means either the existence of a segment between the current position of the agent and p inside the terrain (unlimited vision), or the existence of such a segment of length at most 1 (limited vision). Most of the research in this domain concerns the competitive framework, where the trajectory of the robot not knowing the environment is compared to that of the optimal exploration algorithm having full knowledge.

In [11], the authors gave a 2-competitive algorithm for rectilinear polygon exploration with unlimited vision. The case of non-rectilinear polygons (without obstacles) was also studied in [10, 28] and a competitive algorithm was given in this case.

For polygonal environments with an arbitrary number of polygonal obstacles, it was shown in [11] that no competitive strategy exists, even if all obstacles are parallelograms. Later, this result was improved in [2] by giving a lower bound in $\Omega(\sqrt{k})$ for the competitive ratio of any on-line algorithm exploring a polygon with k obstacles. This bound remains true even for rectangular obstacles. Nevertheless, if the number of obstacles is bounded by a constant k , then there exists a competitive algorithm with competitive ratio in $O(k)$ [10].

Exploration of polygons by a robot with limited vision has been studied, e.g., in [23, 25, 38]. In [23] the authors described an on-line algorithm with competitive ratio $1 + 3(\Pi D/A)$, where Π is a quantity depending on the perimeter of the polygon, D is the area seen by the robot, and A is

the area of the polygon. In [38] the author studied exploration of the boundary of a terrain with limited vision. The cost of exploration of arbitrary terrains with obstacles, both for limited and unlimited vision, was studied in [9].

Navigation in a $n \times n$ square room filled with rectangle obstacles aligned with sides of the square was considered in [5, 6, 7, 39]. It was shown in [5] that the navigation from a corner to the center of a room can be performed with a competitive ratio $O(\log n)$, only using tactile information (i.e., the robot modeled as a point sees an obstacle only when it touches it). No deterministic algorithm can achieve a better competitive ratio, even with unlimited vision [5]. For navigation between any pair of points, there is a deterministic algorithm achieving a competitive ratio of $O(\sqrt{n})$ [7]. No deterministic algorithm can achieve a better competitive ratio [39]. However, there is a randomized approach performing navigation with a competitive ratio of $O(n^{\frac{4}{9}} \log n)$ [6].

Algorithms with advice. The paradigm of algorithms with advice was developed mostly for tasks in graphs. Providing arbitrary types of knowledge that can be used to increase efficiency of solutions to network problems has been proposed in [1, 12, 15, 16, 17, 18, 19, 21, 22, 24, 29, 31, 33, 36, 37, 43]. This approach was referred to as *algorithms with advice*. The advice is given either to the nodes of the network or to mobile agents performing some task in it. In the first case, instead of advice, the term *informative labeling schemes* is sometimes used if different nodes can get different information.

Several authors studied the minimum size of advice required to solve network problems in an efficient way. In [17], the authors compared the minimum size of advice required to solve two information dissemination problems using a linear number of messages. In [19], it was shown that advice of constant size given to the nodes enables the distributed construction of a minimum spanning tree in logarithmic time. In [14, 15], the advice paradigm was used for online problems. In [16], the authors established lower bounds on the size of advice needed to beat time $\Theta(\log^* n)$ for 3-coloring cycles and to achieve time $\Theta(\log^* n)$ for 3-coloring unoriented trees. In the case of [37], the issue was not efficiency but feasibility: it was shown that $\Theta(n \log n)$ is the minimum size of advice required to perform monotone connected graph clearing. In [29], the authors studied radio networks for which it is possible to perform centralized broadcasting in constant time. They proved that constant time is achievable with $O(n)$ bits of advice in such networks, while $o(n)$ bits are not enough. In [22], the authors studied the problem of topology recognition with advice given to the nodes. In [12], the task of drawing an isomorphic map by an agent in a graph was considered, and the problem was to determine the minimum advice that has to be given to the agent for the task to be feasible. Leader election with advice was studied in [26] for trees, and in [13] for arbitrary graphs. Graph exploration with advice was studied in [8, 27] and treasure hunt with advice in graph environments was investigated in [32, 36].

2 Regular terrains

In this section we give three results concerning treasure hunt in regular terrains. First we consider the case $L > \lambda$, where L is the length of a shortest path in the terrain between the initial position of the agent and the location of the treasure, and λ is the accessibility of the treasure. In this case we construct a treasure hunt algorithm working in any regular terrain at cost $O(L)$, using advice of size $O(\log(L/\lambda))$ and we show that this size of advice is optimal for the class of regular terrains.

Then we consider the case $L \leq \lambda$ and provide a treasure hunt algorithm working at cost $O(L)$ for all regular terrains, without any advice.

In our algorithm for $L > \lambda$ we will need to convey advice that is conceptually a pair (a_1, a_2) , where a_i are positive integers. However, by definition, the advice has to be a single binary string, hence it is important to efficiently and unambiguously code such pairs as binary strings, so that the decoding be unambiguous as well and correctly restore the coded pair. This can be done as follows. A pair (a_1, a_2) can be viewed as a string over the 3-symbol alphabet with symbols 0, 1 and comma, where a_1 and a_2 are represented in binary. Code any such string replacing 0 by 01, 1 by 10, and comma by 11. Denote the obtained binary string by $Code(a_1, a_2)$. It is clear that the pair (a_1, a_2) can be unambiguously decoded from $Code(a_1, a_2)$ and that the length of $Code(a_1, a_2)$ is $O(\log(\max(a_1, a_2)))$.

We start by describing the procedure $Walk(\gamma, x)$, that is at the core of both our algorithms for regular terrains. Its high-level idea is the following. Suppose that the initial position of the agent is p and that N is the half-line starting at p that forms angle γ with the direction North. This half-line can possibly intersect some obstacles in the terrain. The aim of the procedure is to travel along the line N circumventing the encountered obstacles in an efficient way, until the total trajectory travelled by the agent has length x . The agent walks along N starting at p . When it hits an obstacle at a point r , the agent finds the other point r' of the intersection of N with the perimeter of the obstacle (such a point r' is unique by the convexity of the obstacle), using a version of the Cow Path walk (searching for an unknown point in the line, cf. [4]) executed on the perimeter. Then the agent goes further along the line N , circumventing each encountered obstacle as above, until the total trajectory traveled by the agent has length x . We will show that if a point q is in the line N at distance y from p and outside of all obstacles then the smallest x such that $Walk(\gamma, x)$ reaches q is $O(y)$. This is due to the fact that if a convex c -fat polygon is cut by a line then the smaller part of its perimeter between the cutting points is only d times larger than the Euclidean distance between these points, where d depends only on c . This is proved in Lemma 2.1.

Suppose that the agent hits an obstacle O at a point r while moving along the line N . The agent starts searching for the other point r' of intersection of N with the perimeter R of the obstacle O by using Procedure $CowPath(R, N, r)$ described below. $CowPath$ is a version of the Cow Path walk on the line transposed to the walk on the perimeter R .

The agent identifies two directions of travelling on R , call these directions dir_1 and dir_2 , defined as follows. Let a and b , with $a \leq b$, be the distances between r and the two closest vertices of the polygon O along the perimeter R .

If $a < b$ and $a \leq 1$ then dir_1 is towards the vertex at distance a . (In this case the agent can see this vertex). Otherwise (either both distances a and b are equal, or they are both larger than 1), the agent could start in any of the two directions but it must be unambiguously defined. There are two cases.

If the point r is one of the vertices of the polygon O then dir_1 corresponds to the side adjacent to r forming a smaller angle with direction North.

If the point r lies in the interior of a side e of the polygon O then dir_1 is defined as follows. If e is horizontal, then direction dir_1 corresponds to West. Otherwise, the direction dir_1 corresponds

to the part of e in the Northern half-plane defined by the horizontal line passing through r . In all cases, dir_2 is the other direction than dir_1 .

The agent walks on the perimeter R of the obstacle O starting at point r . Let $z = \min(1, a)$. First, it goes in direction dir_1 at distance z . Then the agent goes back to r and goes in direction dir_2 until distance $2z$ is travelled or until it visits the point r' , whichever comes first. The agent swings in this way each time doubling the travelled distance until reaching point r' , when it finishes the execution of the procedure. Notice that the agent starts with distance z , rather than distance 1, as opposed to the classic Cow Path walk (that assumes that the target is at distance at least 1) because point r' could be much closer to r than 1 along R . Starting with distance z is safe, as z is smaller than the shorter path between r and r' along R . Algorithm 1 gives the pseudocode of Procedure $CowPath(R, N, r)$.

Algorithm 1: Procedure $CowPath(R, N, r)$

```

1 begin
2   Compute  $dir_1, dir_2$  and  $z$ 
3    $i \leftarrow z$ 
4   while the point  $r'$  on the line  $N$  other than  $r$  is not visited do
5     Go on  $R$  in direction  $dir_1$  until ( $r'$  is visited) or (distance  $i$  is traveled)
6     if the point  $r'$  is visited then
7       | Exit
8     end
9     else
10      | Go back on  $R$  to point  $r$ 
11      |  $i \leftarrow 2i$ 
12     end
13     Go on  $R$  in direction  $dir_2$  until ( $r'$  is visited) or (distance  $i$  is traveled)
14     if the point  $r'$  is visited then
15       | Exit
16     end
17     else
18       | Go back on  $R$  to point  $r$ 
19       |  $i \leftarrow 2i$ 
20     end
21   end
22 end

```

After finding the point r' on the perimeter of the obstacle, the agent continues along the line N , using procedure $CowPath$ whenever an obstacle is hit, until it travels a total trajectory of length x . Algorithm 2 gives the pseudocode of procedure $Walk(\gamma, x)$. The algorithm is interrupted when the length of the total trajectory traveled by the agent is x .

Algorithm 2: Procedure $Walk(\gamma, x)$

```
1 begin
2   Let  $N$  be the half-line starting at  $p$  that forms angle  $\gamma$  with the direction North
3   repeat
4   begin
5     Go along the line  $N$ 
6     if the perimeter  $R$  of an obstacle is hit at point  $r$  then
7       | Call Procedure  $CowPath(R, N, r)$ 
8     end
9   end
10 end
```

Remark. Since perimeters of obstacles are included in the terrain, if the line N along which the agent travels intersects only the perimeter of an obstacle, this is not considered as hitting the obstacle, and the agent continues its travel along N .

The following geometric result will be crucial for the analysis of our algorithms.

Lemma 2.1 *Let $c > 1$ be a constant and let P be a convex c -fat polygon. Consider any line M cutting P at points a and b . Then, there is a constant d such that the length s of the smaller part of the perimeter of P between points a and b is at most $d \cdot |ab|$.*

Proof: Let α and β be the interior angles at points a and b , induced by the cut of the polygon P by the line M . Let $|ab| = x$. We consider the following three cases:

Case 1. $\alpha = \pi - \beta$

In this case the sides of the polygon that are cut by the line M are parallel, see Fig. 1. Let x' be the distance between the parallel lines containing these sides. In Fig. 1, $x' = |ab'|$. By the definition of r , we have, $2r \leq |ab'|$. As the angle $\angle ab'b$ is a right angle, we have $2r \leq |ab'| \leq |ab|$. The length s of the smaller part of the perimeter of P between points a and b is at most $2\pi R$ because the entire convex polygon P is contained in a circle of diameter $2R$. Hence, $s/|ab| \leq 2\pi R/2r \leq \pi \cdot c$. This proves the lemma in Case 1.

Case 2. $\alpha < \pi/2$ and $\beta \leq \pi/2$, or $\alpha > \pi/2$ and $\beta \geq \pi/2$

It is enough to consider the first conjunction, as the second follows by interchanging the two parts of the perimeter of P cut by M . Consider the part of the perimeter of P between points a and b corresponding to angles α and β in the cut by line M . (This is the upper part of the perimeter in Fig. 2). There are two subcases.

Subcase 2.1. The center of the largest circle contained in P is in the upper part of P : Fig. 2 (a).

In this case $x = |ab| \geq |a'b'| \geq 2r$, see Fig. 2 (a). The rest of the proof is as in Case 1.

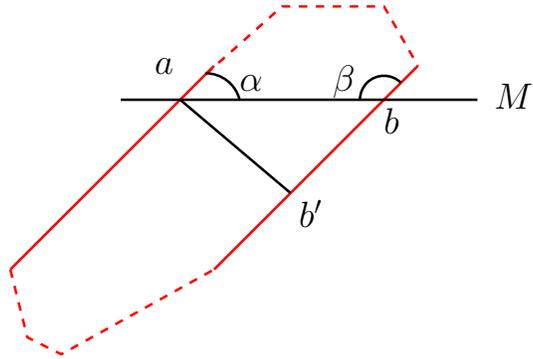


Figure 1: Proof of Lemma 2.1 – Case 1: $\alpha = \pi - \beta$

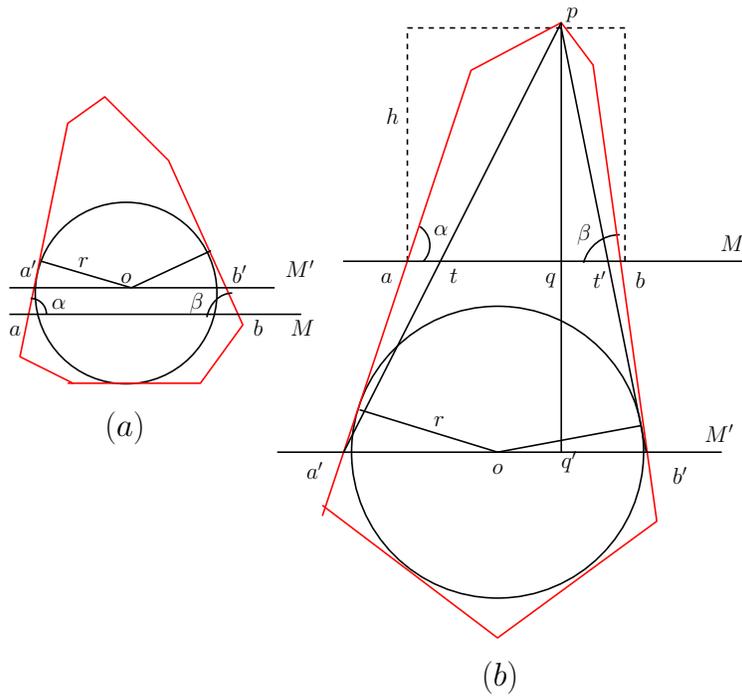


Figure 2: Proof of Lemma 2.1 – Case 2: $\alpha < \pi/2$ and $\beta \leq \pi/2$

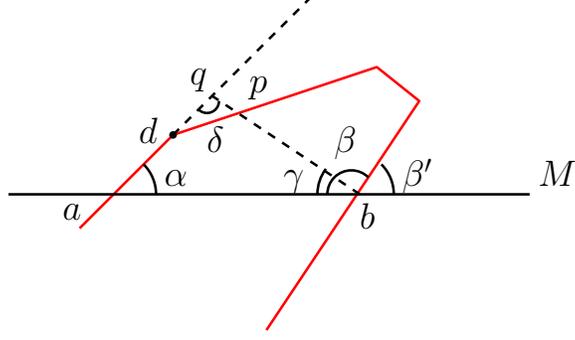


Figure 3: Proof of Lemma 2.1 – Case 3: $\beta \neq \pi - \alpha$, $\alpha < \pi/2$ and $\beta > \pi/2$

Subcase 2.2. The center of the largest circle contained in P is in the lower part of P : Fig. 2 (b).

Let p be the point in the upper part of the perimeter of P farthest from the line M . Consider the line M' parallel to M containing the center of the largest circle contained in P . Let a' and b' be the points in which the line M' cuts the lines containing the sides of P that are cut by M . Let $y = |a'b'|$. Consider the triangle $a'b'p$. Let t and t' be the points where line M cuts the sides $a'p$ and $b'p$ of this triangle, respectively. Let $x' = |tt'|$. Hence $x' \leq x$. Let q and q' be the points of intersection with M and M' , respectively, of the line perpendicular to M and containing p . Let $h = |pq|$ and $h' = |p'q'|$. We have $h/x' = h'/y$. By definition of R and r we have $h' \leq 2R$ and $y \geq 2r$. Hence $h/x' \leq (2R)/(2r) \leq c$. Since $x' \leq x$, we have $h/x \leq c$. Consider the rectangle (with dotted sides in Fig. 2 (b)) whose one side is the segment ab and that contains p . The length of the upper part of the perimeter of P is at most $2h + x$. Hence this length is at most $(2c + 1)x$, which proves the lemma in Case 2.

Case 3. $\beta \neq \pi - \alpha$, $\alpha < \pi/2$ and $\beta > \pi/2$

Let $\beta' = \pi - \beta$ be the exterior angle at point b induced by the cut of polygon P by the line M , see Fig. 3. Without loss of generality, we can assume $\alpha < \beta'$ in the rest of the proof of this case. For $\alpha > \beta'$, the proof is similar. Let d be the endpoint of the side of the polygon P containing point a , in the part of the perimeter corresponding to angle α (the upper part in Fig. 3). Let $y = |ad|$. Let q be the point of intersection of the line containing points a and d and the line perpendicular to the side of polygon P containing point b . Let $z = |dq|$ and $x'' = |bq|$. The segment bq cuts the perimeter of the polygon P at point p , see Fig. 3. Let $x' = |bp|$, and let $t = |dp|$. Denote the angles $\angle abq$ and $\angle aqb$ by γ and δ respectively.

Since $\beta - \gamma = \pi/2$, we have $\gamma = \pi/2 - \beta'$. As $\alpha < \beta'$, we have $\gamma = \pi/2 - \beta' < \pi/2 - \alpha$. Consider the triangle aqb . Since $\alpha + \gamma + \delta = \pi$ and $\alpha + \gamma < \pi/2$, we have $\delta > \pi/2$. Hence, $y + z + x'' \leq 2x$. Since $t \leq z + (x'' - x')$, we have $y + t + x' \leq 2x$. Let the length of the part of the perimeter P , between the points p and b (clockwise in Fig. 3) be s' . By Case 2, there is a constant $k \geq 1$ such that $s'/x' \leq k$. We have $(y + t + s')/x \leq (y + t + kx')/x \leq k(y + t + x')/x$. Since $y + t + x' \leq 2x$, we have $k(y + t + x')/x \leq 2k$. This implies that $(y + t + s')/x \leq 2k$. This proves the lemma in Case 3. \square

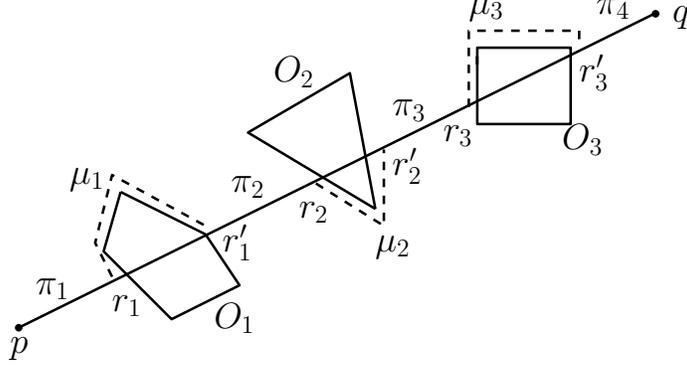


Figure 4: Obstacles intersecting the line N and the construction of the polygonal line Q

We use Lemma 2.1 to prove the following result that will be used in the analysis of our algorithms.

Lemma 2.2 *Let N be the half-line starting at p that forms angle γ with the direction North. Let q be a point in the line N at distance y from p and outside of all obstacles. Then the smallest x such that $Walk(\gamma, x)$ reaches q is $O(y)$.*

Proof: Let O_1, \dots, O_t be the obstacles intersecting line N between p and q , in the order in which they are intersecting it, when the segment pq is traversed from p to q . Let $r_1, r'_1, r_2, r'_2, \dots, r_t, r'_t$ be the points of intersection of N with the perimeters of the obstacles, so that r_i, r'_i belong to the perimeter of O_i . Let π_1 be the segment pr_1 , let π_i , for $i = 2, \dots, t$, be the segment $r'_{i-1}r_i$, and let π_{t+1} be the segment r'_tq . Let μ_i , for $i = 1, \dots, t$, be the shorter part of the perimeter of the obstacle O_i , between r_i and r'_i . (In the case of equality, take any of the two parts). The polygonal line Q is defined as the concatenation of segments $\pi_1, \mu_1, \pi_2, \mu_2, \dots, \pi_t, \mu_t, \pi_{t+1}$ (cf. Fig. 4). By Lemma 2.1, the length of each part μ_i is at most d times larger than the length of the segment $r_i r'_i$, for some constant $d \geq 1$, and thus the length of Q is at most dy .

Consider the smallest x such that $Walk(\gamma, x)$ reaches q . Let \mathcal{T} be the trajectory of the agent, starting from point p and executing procedure $Walk(\gamma, x)$. By definition, the length of \mathcal{T} is x . The competitive ratio of the Cow Path walk on the line is 9 (cf. [4]), and this holds regardless of the length of the first segment used, as long as it does not exceed the distance to the target (this length is 1 in the classic case and z in our case). Hence the length of the part of the trajectory \mathcal{T} corresponding to the perimeter of O_i is at most 9 times larger than μ_i . Hence the length of \mathcal{T} is at most 9 times larger than the length of Q , and hence it is at most $9dy$. This proves the lemma. \square

2.1 The case $L > \lambda$

In the case $L > \lambda$, the high-level idea of the advice and of the algorithm that uses it is the following. Let $k = \lceil 7L/\lambda \rceil$. The oracle constructs k half-lines H_0, \dots, H_{k-1} , starting at the initial position p of the agent, such that H_0 goes North and the angle between consecutive half-lines is $2\pi/k$. For $i = 0, \dots, k-1$, the sector S_i is defined as the set of points in the plane between H_i and $H_{(i+1) \bmod k}$, including H_i and excluding $H_{(i+1) \bmod k}$. Suppose that the treasure is in sector S_j .

The oracle gives the string $Code(k, j)$ as advice. The algorithm decodes the couple (k, j) , and constructs angles $\gamma_1 = 2\pi j/k$ and $\gamma_2 = 2\pi(j+1)/k$. Let M_i , for $i = 1, 2$, be the half-lines starting at p forming angle γ_i with direction North (these are the half-lines bounding sector S_j). Let w' be the distance between p and the closest point in any of the lines M_1, M_2 that does not belong to the terrain. Let $w = \min(1, w')$. The agent walks in phases, alternately on M_1 and M_2 , using procedures $Walk(\gamma_1, 2^j w)$ and $Walk(\gamma_2, 2^j w)$ in phase $j = 0, 1, \dots$, and backtracking to p in each phase using the reverse trajectory, until it sees the treasure. The pseudocode of the algorithm is given in Algorithm 3. It is interrupted as soon as the agent sees the treasure.

Algorithm 3: Algorithm FarTreasure

```

1 begin
2   Decode the couple  $(k, j)$ 
3   Compute angles  $\gamma_1 = 2\pi j/k$  and  $\gamma_2 = 2\pi(j+1)/k$ 
4   Find  $w$ 
5    $j \leftarrow 0$ 
6   repeat
7     begin
8        $Walk(\gamma_1, 2^j w)$ 
9       backtrack to  $p$ 
10       $Walk(\gamma_2, 2^j w)$ 
11      backtrack to  $p$ 
12       $j \leftarrow j + 1$ 
13    end
14 end

```

The following theorem is the main positive result of this section.

Theorem 2.1 *For $L > \lambda$, Algorithm FarTreasure accomplishes treasure hunt in any regular terrain at cost $O(L)$ with advice of size $O(\log(L/\lambda))$.*

In order to prove the theorem, we need the following lemmas.

Lemma 2.3 *The size of advice is $O(\log(\frac{L}{\lambda}))$.*

Proof: As mentioned in the description of the string $Code(a_1, a_2)$, the length of this string is $O(\log(\max(a_1, a_2)))$. We have $k = \lceil 7L/\lambda \rceil$ and $j \leq k$, hence $\log k \in O(\log(L/\lambda))$ and $\log j \in O(\log(L/\lambda))$, which proves the lemma. \square

Lemma 2.4 *For $L > \lambda$, Algorithm FarTreasure correctly accomplishes treasure hunt in any regular terrain at cost $O(L)$.*

Proof: The proof relies on the following geometric claim.

Claim 2.1 Consider a sector with angle $\alpha = 2\pi/k$ formed by half-lines M_1 and M_2 starting at a point p . Let q be a point in this sector at distance at most $L > \lambda$ from p . Then any circle of radius λ containing point q must intersect at least one of the half-lines M_1 or M_2 at distance at most $2L$ from point p .

In order to prove the claim, we first show that any circle of radius λ containing point q must intersect at least one of the half-lines M_1 or M_2 . Suppose by contradiction that there exists such a circle C not intersecting any of these lines. C has the center at a distance $r \leq \lambda + L < 2L$ from p . The largest circle containing q with center at distance r from p not intersecting any of the half-lines M_1 or M_2 has the center c on the bisectrix of angle α . This center c is at distance x from both lines M_1 and M_2 . By our assumption, $x \geq \lambda$. The angle $\beta = \alpha/2$ between the bisectrix and any of the lines M_1 and M_2 corresponds to the arc of length $\rho = \beta r$ of a circle with radius r centered at p . We have $x < \rho = \beta r = \pi r/k < \frac{\pi \cdot 2L}{7L/\lambda} < \lambda$. This is a contradiction that proves that any circle of radius λ containing point q must intersect at least one of the half-lines M_1 or M_2 .

It remains to show that some point of intersection of such a circle with at least one of the half-lines M_1 or M_2 is at distance at most $2L$ from p . The distance from p to the closest such point must be smaller than $r < 2L$, which concludes the proof of the claim.

Using the above claim, the lemma can be proved as follows. Let Δ be a circle of radius λ containing the treasure and contained in the terrain. Let M_i , for $i = 1, 2$, be the half-line starting at p forming angle γ_i with direction North (these are the half-lines bounding the sector containing the treasure). Since the treasure is at distance at most L from p , it follows from the claim that there exists a point q in Δ and situated in one of the lines M_i , at distance at most $2L$ from p . By Lemma 2.2, the agent will get to point q during the execution of the repeat loop of Algorithm *FarTreasure* for the smallest j such that $bL \leq 2^j w$, where b is some constant. Denote this smallest j by j_0 . At the point q the agent sees the treasure. Hence the cost of Algorithm *FarTreasure* is at most $4w(1 + 2 + \dots + 2^{j_0}) \leq 8w2^{j_0}$. Since by definition of j_0 we have $w2^{j_0-1} < bL$, the cost of Algorithm *FarTreasure* is at most $16bL$, which concludes the proof of the lemma. \square

Now the proof of Theorem 2.1 is a direct consequence of Lemmas 2.3 and 2.4.

We end this section by showing that the size $O(\log(L/\lambda))$ of advice used by Algorithm *FarTreasure* is optimal for the class of regular terrains. In order to prove this, we construct a class of regular terrains with treasure accessibility λ for which any treasure hunt algorithm working at cost $O(L)$ must use advice at least $\frac{1}{2} \log(L/\lambda)$.

We start the construction by defining the *gadget* $G(o)$, for any point o in the plane. The gadget consists of 8 squares of side $x = 3\lambda/2$, situated as follows (see Fig. 5). There are four squares $\sigma_N, \sigma_E, \sigma_S, \sigma_W$ whose centers are at distance $\lambda + x/2$ from point o , respectively, North, East, South and West from this point. The remaining four squares are placed as follows. Let $y = (2\lambda - x)/2$. The center of σ_{NW} is West of the center of σ_N , at distance $x + y$ from it. The center of σ_{NE} is East of the center of σ_N , at distance $x + y$ from it. The center of σ_{SE} is East of the center of σ_S , at distance $x + y$ from it. The center of σ_{SW} is West of the center of σ_S , at distance $x + y$ from it. Notice that the NW corner of σ_{NW} , the NE corner of σ_{NE} , the SE corner of σ_{SE} and the SW corner of σ_{SW} are corners of a square $S(o)$ of side $3x + 2y = 5\lambda$ which is the convex hull of the eight squares of the gadget.

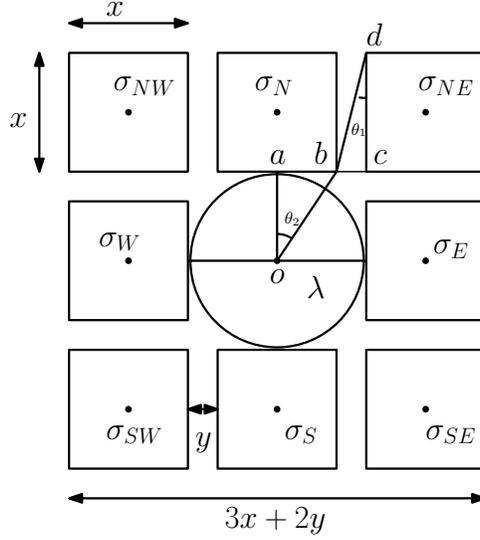


Figure 5: The gadget $G(o)$

Lemma 2.5 *An agent located outside of the square $S(o)$ cannot see the point o .*

Proof: Let a be the point North from o at distance λ from it. Let b be the SE corner of σ_N . Let θ_2 be the angle $\angle aob$. Let c be the SW corner of σ_{NE} . Let d the NW corner of σ_{NE} . Let θ_1 be the angle $\angle cdb$, see Fig. 5. Due to the symmetries of the gadget $G(o)$, the lemma follows from the inequality $\theta_2 > \theta_1$. This inequality is proved as follows. $\tan \theta_2 = x/(2\lambda)$. $\tan \theta_1 = y/x$. Hence

$$\tan \theta_2 = \frac{3\lambda/2}{2\lambda} = 3/4 > 1/6 = \frac{\lambda/4}{3\lambda/2} = \frac{(2\lambda - x)/2}{x} = y/x = \tan \theta_1.$$

Hence $\theta_2 > \theta_1$. □

Theorem 2.2 *Let λ be a real in the interval $(0, 1/2]$. There exist arbitrarily large reals L and regular terrains with an initial position p of the agent and the location q of the treasure in it, such that L is the length of a shortest path in the terrain between p and q , λ is the accessibility of the treasure, and the agent needs advice of size at least $\frac{1}{2} \log(L/\lambda)$ to accomplish treasure hunt at cost $O(L)$ in this terrain.*

Proof: We construct the regular terrain as follows. Consider a square with side $A = 20k\lambda$, where k is a positive integer. Let the initial position p of the agent be the South-West corner of S , see Fig. 6. Partition the square S into four equal quadrants. Let S' be the top-right quadrant. S' has side of length $A/2$. Partition S' into $\frac{A^2}{100\lambda^2}$ square tiles of side 5λ . Tile rows are indexed $1, 2, \dots$ from the North side of S' going South, and tile columns, and are indexed $1, 2, \dots$ from the West side of S' going East. Now, place the gadget $G(o)$ contained in square $S(o)$ of side 5λ (see Fig. 5) in every other tile of odd-indexed tile rows in S' (see Fig. 6, where these tiles are shaded: we will call them shaded tiles in the sequel). The distance between any two shaded tiles is at least 5λ .

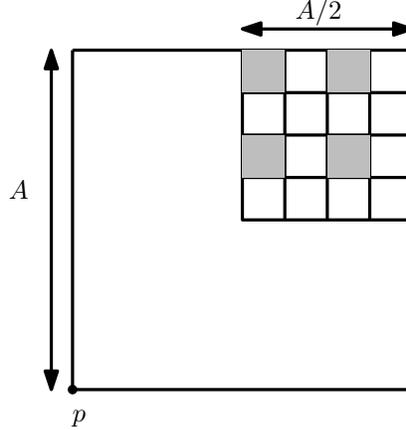


Figure 6: Regular terrain with initial position p of the agent

At least $1/4$ of all tiles are shaded, so the number of such tiles is at least $\frac{A^2}{400\lambda^2}$. This finishes the description of the terrain. Since all obstacles are squares, the terrain is regular. Notice that if the treasure is placed in the center of any shaded tile then λ is the accessibility of the treasure.

We prove the theorem by contradiction. Suppose that the minimum size of advice to accomplish treasure hunt at cost $O(L)$ in any terrain from the above described class is less than $\frac{1}{2} \log(L/\lambda)$. Consider all possible locations of the treasure in the center of any of the shaded tiles. Notice that the length L of a shortest path in the terrain from p to any such center is in the interval $[\sqrt{2}A/2, \sqrt{2}A/2 + A]$.

Using less than $\frac{1}{2} \log(L/\lambda)$ bits, we have at most $\sqrt{L/\lambda}$ different advice strings. By the pigeonhole principle, there is a subset Σ of at least $\frac{A^2/(400\lambda^2)}{\sqrt{L/\lambda}}$ shaded tiles, such that the location of the treasure at their center corresponds to the same advice string. By Lemma 2.5, in order to see the treasure located at the center of a shaded tile, the agent must get to this tile. Since the minimum distance between any two shaded tiles is at least 5λ , any trajectory T of the agent that enables it to accomplish treasure hunt when the treasure is located at the center of some tile in the set Σ , must have length at least $\frac{(A^2/(400\lambda^2))}{\sqrt{L/\lambda}} \cdot 5\lambda = \frac{A^2}{80\sqrt{\lambda L}}$. Since L is in the interval $[\sqrt{2}A/2, \sqrt{2}A/2 + A]$, the length of T is at least $c \frac{L^{3/2}}{\sqrt{\lambda}}$ for some positive constant c . Since $\lambda \leq 1/2$, the cost of treasure hunt is at least $cL^{3/2}$ and hence cannot be linear in L , which gives a contradiction. \square

Theorems 2.1 and 2.2 imply that for $L > \lambda$, the size $O(\log(L/\lambda))$ of advice used by Algorithm *FarTreasure* is sufficient to find the treasure in all regular terrains at cost $O(L)$ and cannot be improved.

2.2 The case $L \leq \lambda$

In the case $L \leq \lambda$, we show an algorithm working without any advice and finding the treasure at cost $O(L)$. The high-level idea of the algorithm is the following. Let L_i , for $i = 0, \dots, 11$, be the half-lines starting at the initial position p of the agent and forming angle $\gamma_i = \pi i/6$ with direction

North. Let w' be the distance between p and the closest point in any of the lines L_i that does not belong to the terrain. Let $w = \min(1, w')$. The agent walks in phases, in a round-robin fashion on lines L_0, L_1, \dots, L_{11} , using procedures $Walk(\gamma_0, 2^j w)$, $Walk(\gamma_1, 2^j w)$, ..., $Walk(\gamma_{11}, 2^j w)$ in phase $j = 0, 1, \dots$, and backtracking to p in each phase using the reverse trajectory, until it sees the treasure. The pseudocode of the algorithm is given in Algorithm 4. It is interrupted as soon as the agent sees the treasure.

Algorithm 4: Algorithm CloseTreasure

```

1 begin
2   Find  $w$ 
3    $j \leftarrow 0$ 
4   repeat
5     begin
6       for  $i := 0$  to 11 do
7          $Walk(\pi i/6, 2^j w)$ 
8         backtrack to  $p$ 
9       end
10       $j \leftarrow j + 1$ 
11    end
12 end

```

The following result proves the correctness and estimates the cost of Algorithm *CloseTreasure*.

Theorem 2.3 *For $L \leq \lambda$, Algorithm CloseTreasure accomplishes treasure hunt in any regular terrain at cost $O(L)$ with no advice.*

Proof: We consider two cases. Let Δ be a circle of radius λ containing the treasure and contained in the terrain.

Case 1. $\lambda/9 \leq L \leq \lambda$.

We first prove the following claim, similar to Claim 2.1

Claim 2.2 *Consider a sector with angle $\alpha = \pi/6$ formed by half-lines M_1 and M_2 starting at a point p . Let q be a point in this sector at distance at most L from p , where $\lambda/9 \leq L \leq \lambda$. Then any circle of radius λ containing point q must intersect at least one of the half-lines M_1 or M_2 at distance at most $10L$ from point p .*

In order to prove the claim, we first show that any circle of radius λ containing point q must intersect at least one of the half-lines M_1 or M_2 . Suppose by contradiction that there exists such a circle C not intersecting any of these lines. C has the center at a distance $r \leq \lambda + L < 2\lambda$ from p . The largest circle containing q , with center at distance r from p , not intersecting any of the half-lines M_1 or M_2 has the center c on the bisectrix of angle α . This center c is at distance x from both lines M_1 and M_2 . By our assumption, $x \geq \lambda$. The angle $\beta = \alpha/2 = \pi/12$ between the

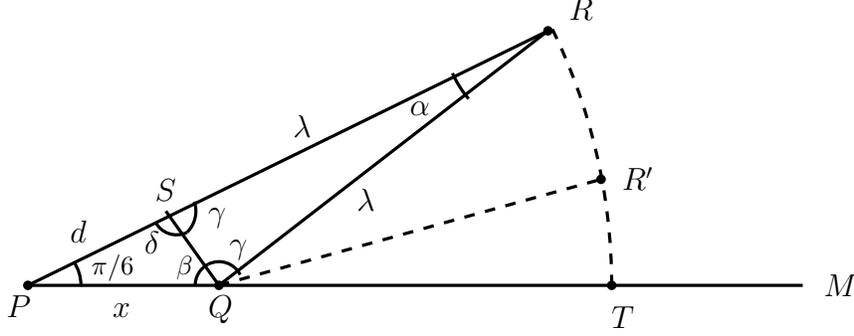


Figure 7: Illustration of the proof of Lemma 2.3

bisectrix and any of the lines M_1 and M_2 corresponds to the arc of length $\rho = \beta r$ of a circle with radius r centered at p . We have $x < \rho = \beta r = \pi r/12 \leq \frac{\pi \cdot 2\lambda}{12} < \lambda$. This is a contradiction that proves that any circle of radius λ containing point q must intersect at least one of the half-lines M_1 or M_2 .

It remains to show that some point of intersection of such a circle with at least one of the half-lines M_1 or M_2 is at distance at most $10L$ from p . The distance from p to the closest such point must be smaller than $r \leq \lambda + L \leq 10L$, which concludes the proof of the claim.

Let Q be the half-line starting at p and containing the treasure. Suppose that the angle γ between the direction North and Q satisfies $\pi i/6 \leq \gamma < \pi(i+1)/6$. Hence the treasure is in the sector formed by half-lines L_i and L_{i+1} . Let $M_1 = L_i$ and $M_2 = L_{i+1}$. Since the treasure is at distance at most L from p , it follows from Claim 2.2 that there exists a point q' in Δ and situated in one of the lines M_i , at distance at most $10L$ from p .

Case 2. $L < \lambda/9$.

In this case we will use the following claim.

Claim 2.3 Consider positive reals d, λ , such that $\lambda \geq 9d$. Let PR be a line segment of length $\lambda + d$ and let S be a point in this segment such that $|PS| = d$ and $|SR| = \lambda$. Let M be a half-line starting from point P , forming an angle $\pi/6$ with the segment PR . Let Q be the point in M closest to P such that $|QR| = \lambda$. Then $|PQ| \leq 1.2 \cdot |PS|$.

Let $|PQ| = x$ and $|PS| = d$. Denote the angle $\angle PSQ$ by δ , the angle $\angle SQR$ by γ , the angle $\angle QRS$ by α and the angle $\angle PQS$ by β , see Fig. 7.

By the sine rule applied to the triangle PQR , we have $\frac{\sin(\pi/6)}{\sin(\beta+\gamma)} = \frac{\lambda}{\lambda+d}$. Since $\lambda/(\lambda+d) \geq 0.9$, we have $\sin(\beta+\gamma) \leq \sin(\pi/6)/0.9 \leq 0.56$. We have $\arcsin 0.56 = 0.594$. Let $A = (\pi - 0.594) \geq 2.54$. Since Q is the point in M closest to P such that $|QR| = \lambda$, we have $(\beta+\gamma) > \pi/2$. Thus, by definition of A we have $(\beta+\gamma) \geq A$. Using the triangle PQR , we have $\alpha = \pi - \pi/6 - (\beta+\gamma) \leq \frac{5\pi}{6} - A$. Since the triangle SQR is isoceles, we have $\gamma = (\pi - \alpha)/2$. Since $\alpha \leq \frac{5\pi}{6} - A$, we have $\gamma \geq \frac{\pi}{12} + \frac{A}{2}$. By definition, we have $\delta = \pi - \gamma$. Since $\gamma \geq \frac{\pi}{12} + \frac{A}{2}$, we have $\delta \leq \frac{11\pi}{12} - \frac{A}{2}$. Using the triangle PQS , we have $\beta = (\pi - \frac{\pi}{6} - \delta) = \frac{5\pi}{6} - \delta$. Since $\delta \leq \frac{11\pi}{12} - \frac{A}{2}$, we have $\beta \geq \frac{A}{2} - \frac{\pi}{12} \geq \frac{2.54}{2} - \frac{\pi}{12} \geq 1$.

By the sine rule applied to the triangle PQS , we $\frac{x}{d} = \frac{\sin \delta}{\sin \beta} \leq \frac{1}{\sin \beta}$. Since $\pi/2 \geq \beta \geq 1$, we have $\sin \beta \geq \sin 1$ and thus $\frac{x}{d} \leq \frac{1}{\sin 1} \leq 1.2$. This proves the claim.

Let d be the distance between point p and the treasure. We have $d \leq L < \lambda/9$. The distance between p and the center of Δ is at most $\lambda + d$. Let $M_1 = L_i$ and $M_2 = L_{i+1}$ be half-lines such that the center of Δ is located in the sector formed by M_1 and M_2 .

First suppose that the center of Δ is located on one of the lines M_1 or M_2 at distance exactly $d + \lambda$ from p . Since the angle between lines M_1 and M_2 is $\pi/6$, it follows from Claim 2.3 (with P replaced by p) that circle Δ intersects the other line M_i at distance $x \leq 1.2 \cdot d$ from p . Hence, if the center of Δ is located anywhere in the sector formed by M_1 and M_2 at some distance at most $\lambda + d$ from p , the circle Δ intersects both lines M_i at distance $x \leq 1.2 \cdot d$ from p . This is due to the fact that in the situation depicted in Fig.7 the length of the segment PQ is a decreasing function of the length of the segment PR and of the angle between segments PR and PQ .

Hence in both cases we showed the existence of a point q' in Δ and situated in one of the lines M_i , at distance at most $10L$ from p . By Lemma 2.2, the agent will get to point q' during the execution of the repeat loop of Algorithm *CloseTreasure* for the smallest j such that $gL \leq 2^j w$, where g is some constant. Denote this smallest j by j_0 . At the point q' the agent sees the treasure. Hence the cost of Algorithm *CloseTreasure* is at most $24w(1 + 2 + \dots + 2^{j_0}) \leq 48w2^{j_0}$. Since by definition of j_0 we have $w2^{j_0-1} < gL$, the cost of Algorithm *CloseTreasure* is at most $96gL$, which concludes the proof of the theorem in both cases. \square

Remarks.

1. Our algorithms *FarTreasure* and *CloseTreasure* can be merged into a single treasure hunt algorithm preserving the features of the two algorithms in the respective cases: with no advice use algorithm *CloseTreasure* and with a non-empty advice string use algorithm *FarTreasure*.
2. The threshold for L in comparison to λ separating the cases that yield treasure hunt with advice $O(\log(L/\lambda))$ and treasure hunt with no advice is somewhat arbitrary. It is easy to see that whenever $L \leq c\lambda$, for some positive (even very large) constant c , a treasure hunt algorithm at cost $O(L)$ with no advice can be designed by suitably increasing the number of half-lines L_i along which procedure *Walk* is executed in Algorithm *CloseTreasure*, i.e., decreasing the angle $\pi/6$ between consecutive half-lines. This would result in increasing the constant hidden in the $O(L)$ cost bound. Using Algorithm *FarTreasure* for $\lambda < L \leq c\lambda$ makes this hidden constant lower at the expense of a constant number of bits of advice.

3 Arbitrary terrains

In this section we show that the advice complexity of treasure hunt is dramatically larger for the class of arbitrary terrains than for that of regular terrains. We show that the size of advice required for treasure hunt at cost $O(L)$ in some non-convex polygons, even without obstacles and even with all sides horizontal or vertical, is exponentially larger than that for regular terrains. In fact, we will show that this difference may be even more significant.

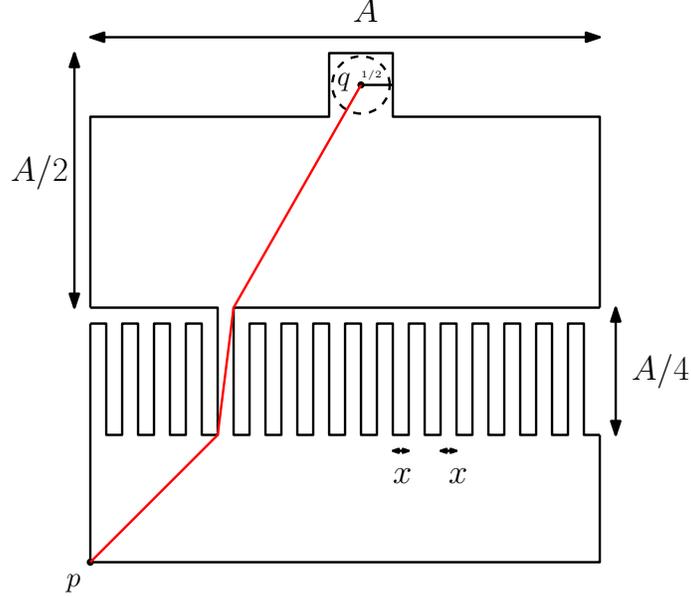


Figure 8: A non-convex terrain (without obstacles) with the initial position p of the agent and the location q of the treasure in it. The accessibility of the treasure is $\lambda = 1/2$.

Theorem 3.1 *For arbitrarily large integers A , there exists a class $\mathcal{C}(A)$ of (non-convex) polygons \mathcal{P} without obstacles, with an initial position p of the agent and the location q of the treasure in each of these polygons, so that the length L of a shortest path between p and q in \mathcal{P} is $\Theta(A)$, the accessibility of the treasure is $\lambda = 1/2$, and the smallest size of advice required for treasure hunt at cost $O(L)$ in all polygons of this class, is at least linear in L .*

Proof: Fix an integer $A > 8$. We construct the class $\mathcal{C}(A)$ of (nonconvex) polygons \mathcal{P}_i as follows. Consider the square S of side length A with vertical and horizontal sides. Remove from the square S two rectangles with horizontal side of length $(A - 1)/2$ and vertical side of length 1: one of these removed rectangles has its upper-left corner at the upper-left corner of S and the other has its upper-right corner at the upper-right corner of S (see Fig. 8). In the third quarter of the height of the square S (counting from the upper horizontal side) we remove vertical stripes of width $x = 1/2^A$ and height $A/4 - x$, and two horizontal stripes with upper side at height $A/2$ of the square, of height x , and lengths respectively $2(i - 1)x$, for the left horizontal stripe and $A - (2i - 3)x$, for the right horizontal stripe (see Fig. 8). Let $k = A/(2x) = A \cdot 2^{A-1}$. In each of the polygons \mathcal{P}_i there are k vertical corridors of width x , $k - 1$ of them closed from above at the height $A/2 + x$ (counting from the upper horizontal side of the square S), and one corridor open, exactly the i th corridor counting from the left.

The initial position p of the agent is the lower-left corner of the square S , and the location q of the treasure is at distance $1/2$ from the upper horizontal side of S and in the middle between the two vertical sides of S (see Fig. 8). Thus the accessibility of the treasure is $\lambda = 1/2$ and the length L of a shortest path between p and q in \mathcal{P} satisfies the inequalities $A/2 < L < 5A/2$, and hence $L \in \Theta(A)$.

We prove the theorem by contradiction. Suppose that the size of advice is at most $A/2$. Hence there are at most $2^{A/2}$ distinct pieces of advice. The number of polygons in the class $\mathcal{C}(\mathcal{A})$ is $k = A \cdot 2^{A-1}$. By the pigeonhole principle there are at least $y = k/2^{A/2} = A \cdot 2^{A/2-1}$ polygons in the class $\mathcal{C}(\mathcal{A})$ to which corresponds the same advice α . Let $\mathcal{P}_{i_1}, \dots, \mathcal{P}_{i_y}$ be these polygons. Consider the trajectory of the agent corresponding to advice α . This trajectory must enter each of the corridors i_1, \dots, i_y , counted from the left, at the height at least $A/4 + 1/2$, counting from the bottom side of the square S . This means that the agent must enter each of these corridors at depth at least $1/2$ from the beginning of the corridor. At any lower point in a corridor, the agent cannot see if the corridor is open or closed (because it can see only at distance at most 1), and hence not going deeper in one of the corridors $j \in \{i_1, \dots, i_y\}$ would preclude it from seeing the treasure, if the actual polygon is \mathcal{P}_j . Hence the trajectory of the agent corresponding to advice α must have a length of at least $2 \cdot (1/2) \cdot y = y$, in the case when the actual polygon is \mathcal{P}_j , where the last visited corridor has index j . However, $y = A \cdot 2^{A/2-1}$ is not in $O(A)$ and hence not in $O(L)$, which is a contradiction. This contradiction proves that the size of advice must be larger than $A/2$, and hence it must be in $\Omega(L)$. \square

Remark. By replacing $x = 1/2^A$ in the above proof by $1/f(A)$, for any faster growing function $f(A)$ (for example $f(A) = 2^{2^A}$) we could get the lower bound $\Omega(\log f(L))$ on the required size of advice (instead of just $\Omega(L)$), and hence show an arbitrarily large difference between advice complexity of treasure hunt in arbitrary vs. regular terrains.

4 Conclusion

Using advice complexity as a measure of the difficulty of a task, we established that treasure hunt in the class of arbitrary terrains is dramatically more difficult than in the class of regular terrains. A natural intermediate class of terrains is that of convex polygons with arbitrary convex obstacles (not necessarily c -fat). It remains open what is the advice complexity of treasure hunt in this class.

A problem related to treasure hunt is that of finding a shortest path in a terrain. What is the advice complexity of this problem, i.e., what is the smallest advice that the agent needs in order to find a path of length exactly L (rather than of length $O(L)$) to the target? Unfortunately, this is not a good formulation, as no finite advice could permit the agent to solve this problem, even in the empty plane. Intuitively, the advice would have to convey the exact direction to the target, which cannot be done with a finite number of bits, as the target is a point. (This simple observation can be easily formalized). An attempt to relax the task by requiring the (exact) shortest path not to hit the target but to “see it”, i.e., get at distance 1 from it, must still fail for the same reason. It seems that a reasonable formulation of the shortest path problem in a terrain, in the context of advice complexity, has to relax the term “shortest”. For example, the relaxation could be up to an *additive* constant (rather than up to a *multiplicative* constant, as we did, requiring cost $O(L)$). More precisely, the following problem remains open. What is the best complexity of advice sufficient to solve the treasure hunt problem in a terrain (again, the agent has to see the treasure), at cost $L + O(1)$? Since the agent can see at distance 1, this version of treasure hunt is equivalent to solving the shortest path problem up to an additive constant. Similarly as in this paper, it would be interesting to find whether the difficulty of this problem (measured by advice complexity) varies for different classes of terrains.

References

- [1] S. Abiteboul, H. Kaplan, T. Milo, Compact labeling schemes for ancestor queries, Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2001), 547–556.
- [2] S. Albers and K. Kursawe and S. Schuierer, Exploring unknown environments with obstacles, *Algorithmica* 32 (2002), 123–143.
- [3] S. Alpern and S. Gal, *The Theory of Search Games and Rendezvous*, Kluwer Academic Publications, 2003.
- [4] R. Baeza-Yates, J. Culberson, and J. Rawlins, Searching the plane, *Information and Computation* 106 (1993), 234–252.
- [5] E. Bar-Eli, P. Berman, A. Fiat and R. Yan, On-line navigation in a room, *Journal of Algorithms* 17 (1994), 319–341.
- [6] P. Berman, A. Blum, A. Fiat, H. Karloff, A. Rosen and M. Saks, Randomized robot navigation algorithms, Proc. 7th ACM-SIAM Symp. on Discrete Algorithms (1996), 74–84.
- [7] A. Blum, P. Raghavan and B. Schieber, Navigating in unfamiliar geometric terrain, *SIAM Journal on Computing* 26 (1997), 110–137.
- [8] H.J. Bockenhauer, J. Fuchs, W. Unger, The Graph Exploration Problem with Advice. CoRR abs/1804.06675 (2018).
- [9] J. Czyzowicz, D. Ilcinkas, A. Labourel, A. Pelc, Worst-case optimal exploration of terrains with obstacles, *Information and Computation* 225 (2013), 16–28.
- [10] X. Deng, T. Kameda and C. H. Papadimitriou, How to learn an unknown environment, Proc. 32nd Symp. on Foundations of Computer Science (FOCS 1991), 298–303.
- [11] X. Deng, T. Kameda and C. H. Papadimitriou, How to learn an unknown environment I: the rectilinear case, *Journal of the ACM* 45 (1998), 215–245.
- [12] D. Dereniowski, A. Pelc, Drawing maps with advice, *Journal of Parallel and Distributed Computing* 72 (2012), 132–143.
- [13] Y. Dieudonné, A. Pelc, Impact of knowledge on election time in anonymous networks, Proc. 29th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2017), 207–215.
- [14] S. Dobrev, R. Kralovic, E. Markou. Online graph exploration with advice. Proc. 19th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2012), 267–278.
- [15] Y. Emek, P. Fraigniaud, A. Korman, A. Rosen, Online computation with advice, *Theoretical Computer Science* 412 (2011), 2642–2656.
- [16] P. Fraigniaud, C. Gavoille, D. Ilcinkas, A. Pelc, Distributed computing with advice: Information sensitivity of graph coloring, *Distributed Computing* 21 (2009), 395–403.

- [17] P. Fraigniaud, D. Ilcinkas, A. Pelc, Communication algorithms with advice, *Journal of Computer and System Sciences* 76 (2010), 222–232.
- [18] P. Fraigniaud, D. Ilcinkas, A. Pelc, Tree exploration with advice, *Information and Computation* 206 (2008), 1276–1287.
- [19] P. Fraigniaud, A. Korman, E. Lebhar, Local MST computation with short advice, *Theory of Computing Systems* 47 (2010), 920–933.
- [20] G. M. Fricke, J. P. Hecker, A. D. Griego, L. T. Tran and Melanie E. Moses, A Distributed Deterministic Spiral Search Algorithm for Swarms, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*, 4430-4436.
- [21] E. Fusco, A. Pelc, Trade-offs between the size of advice and broadcasting time in trees, *Algoritmica* 60 (2011), 719–734.
- [22] E. Fusco, A. Pelc, R. Petreschi, Topology recognition with advice, *Information and Computation* 247 (2016), 254-265.
- [23] Y. Gabriely, Y. Rimon, Spanning-tree based coverage of continuous areas by a mobile robot, *Proc. Int. Conf. of Robotics and Automaton (ICRA 2001)*, 1927-1933.
- [24] C. Gavoille, D. Peleg, S. Pérennes, R. Raz. Distance labeling in graphs, *Journal of Algorithms* 53 (2004), 85-112.
- [25] S.K. Ghosh, J.W. Burdick, A. Bhattacharya and S. Sarkar, Online algorithms with discrete visibility - exploring unknown polygonal environments, *Robotics & Automation Magazine* 15 (2008), 67-76.
- [26] C. Glacet, A. Miller, A. Pelc, Time vs. information tradeoffs for leader election in anonymous trees, *ACM Transactions on Algorithms* 13 (2017), 31:1-31:41.
- [27] B. Gorain, A. Pelc, Deterministic graph exploration with advice, *Proc. 44th International Colloquium on Automata, Languages and Programming (ICALP 2017)*, 132:1-132:14.
- [28] F. Hoffmann. C. Icking, R. Klein and K. Kriegel, The polygon exploration problem, *SIAM J. Comput.* 31 (2001), 577–600.
- [29] D. Ilcinkas, D. Kowalski, A. Pelc, Fast radio broadcasting with advice, *Theoretical Computer Science*, 411 (2012), 1544–1557.
- [30] A. Jez and J. Lopuszanski, On the two-dimensional cow search problem, *Information Processing Letters* 109 (2009), 543 - 547.
- [31] M. Katz, N. Katz, A. Korman, D. Peleg, Labeling schemes for flow and connectivity, *SIAM Journal of Computing* 34 (2004), 23–40.
- [32] D. Komm, R. Kralovic, R. Kralovic, J. Smula, Treasure hunt with advice, *Proc. 22nd International Colloquium on Structural Information and Communication Complexity (SIROCCO 2015)*, 328-341.

- [33] A. Korman, S. Kutten, D. Peleg, Proof labeling schemes, *Distributed Computing* 22 (2010), 215–233.
- [34] E. Langetepe, On the Optimality of Spiral Search, *Proc. 21st Ann. ACM-SIAM Symp. Disc. Algor. (SODA 2010)*, 1-12.
- [35] E. Langetepe, Searching for an axis-parallel shoreline, *Theoretical Computer Science* 447 (2012), 85-99.
- [36] A. Miller, A. Pelc, Tradeoffs between cost and information for rendezvous and treasure hunt, *Journal of Parallel and Distributed Computing* 83 (2015), 159-167.
- [37] N. Nisse, D. Soguet, Graph searching with advice, *Theoretical Computer Science* 410 (2009), 1307–1318.
- [38] S. Ntafos, Watchman routes under limited visibility, *Comput. Geom. Theory Appl.* 1 (1992), 149–170.
- [39] C. H. Papadimitriou, M. Yannakakis, Shortest paths without a map, *Theor. Comput. Sci.* 84 (1991), 127–150.
- [40] A. Pelc, Reaching a target in the plane with no information, *Information Processing Letters* 140 (2018), 13-17.
- [41] K. Spieser and E. Frazzoli, The Cow-Path Game: A Competitive Vehicle Routing Problem, *Proc. 51st IEEE Conference on Decision and Control* (2012), 6513 - 6520.
- [42] A. Ta-Shma and U. Zwick, Deterministic rendezvous, treasure hunts and strongly universal exploration sequences. *ACM Transactions on Algorithms* 10 (2014), 12:1-12:15.
- [43] M. Thorup, U. Zwick, Approximate distance oracles, *Journal of the ACM*, 52 (2005), 1–24.