Trouver un trésor plus rapidement avec des conseils angulaires
Sébastien Bouchard, Yoann Dieudonné, Andrzej Pelc, Franck Petit

To cite this version:
Sébastien Bouchard, Yoann Dieudonné, Andrzej Pelc, Franck Petit. Trouver un trésor plus rapidement avec des conseils angulaires. ALGOTEL 2019 - 21èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications, Jun 2019, Saint Laurent de la Cabrerisse, France. hal-02118362

HAL Id: hal-02118362
https://inria.hal.science/hal-02118362
Submitted on 3 May 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Trouver un trésor plus rapidement avec des conseils angulaires†

Sébastien Bouchard1, Yoann Dieudonné2, Andrzej Pelc3, and Franck Petit1

1Sorbonne Université, CNRS, INRIA, LIP6, F-75005 Paris, France
2Laboratoire MIS, Université de Picardie Jules Verne, Amiens, France
3Département d’informatique, Université du Québec en Outaouais, Gatineau, Canada

Un touriste égaré souhaite retrouver son chemin vers la gare de train la plus proche. Heureusement pour lui, au cours de sa recherche, il rencontre de nombreuses personnes enclines à le renseigner. Toutefois, la plupart ne connaissent pas la direction exacte à suivre. N’ayant qu’une vague idée du chemin à prendre, lorsqu’elles sont interrogeées, elles répondent en levant et en écartant simplement les bras pour indiquer un angle où la gare recherchée peut se trouver. Selon la précision du renseignement (c’est-à-dire de la largeur de l’angle), le touriste peut évidemment être amené à prendre des chemins l’écartant de la bonne direction.

De tels renseignements sont-ils de nature à amener notre touriste à destination en suivant un algorithme déterministe ? Et dans l’affirmative, après avoir parcouru quelle distance ? Le présent papier présente les réponses que nous avons apportées à ces deux questions dans un article récemment publié [BDPP18]. Le problème abordé ici, plus connu dans la littérature sous l’appellation de chasse au trésor, n’avait encore jamais été étudié en présence de conseils angulaires.

Mots-clefs : Chasse au trésor, agent mobile, algorithme déterministe

1 Introduction

1.1 The model and problem formulation

A mobile agent (modeling our lost tourist) equipped with a compass and a measure of length has to find an inert treasure (e.g., the train station) in the Euclidean plane. Both the agent and the treasure are modeled as points. In the beginning, the agent is at a distance at most $D > 1$ from the treasure, but knows neither the distance nor any bound on it. Finding the treasure means getting at distance at most 1 from it. Finding the treasure means getting at distance at most 1 from it. In applications, from such a distance the treasure can be seen. The agent makes a series of moves. Each of them consists in moving straight in a chosen direction at a chosen distance. At the beginning and after each move the agent gets a hint consisting of a positive angle smaller than $2\pi$ whose vertex is at the current position of the agent and within which the treasure is contained. Within a series of moves, the angles of any two hints are not necessarily equal. We investigate the problem of how these hints permit the agent to lower the cost of finding the treasure, using a deterministic algorithm, where the cost is the worst-case total length of the agent’s trajectory. It is well known from [BCR93] that the optimal cost of treasure hunt without hints is $\Theta(D^2)$.

1.2 Our results

In [BDPP18], we present three different results. First, we show that if all angles given as hints are at most $\pi$, then the cost of treasure hunt can be lowered to $O(D)$, which is optimal. The main difficulty of this result consists in handling all angles at most $\pi$. The case of all angles strictly smaller than $\pi$ is simpler and can be derived from existing literature on $\phi$-self approaching curves [AAI+01]. Our real challenge here is in the
fact that hints can be angles of size exactly $\pi$, in which case the design of a trajectory always leading to the
treasure, while being cost-efficient in terms of traveled distance, is far from obvious. Next, if all angles are
at most $\beta$, where $\beta < 2\pi$ is a constant unknown to the agent, then we prove that the cost is at most $O(D^{2-\epsilon})$,
for some $\epsilon > 0$. Finally, we show that arbitrary angles smaller than $2\pi$ given as hints cannot be of significant
help: using such hints the cost $\Theta(D^2)$ cannot be beaten.

For both our positive results, we presented in [BDPP18] deterministic algorithms achieving the above
costs. Both algorithms work in phases “assuming” that the treasure is contained in increasing squares centered
at the initial position of the agent. The common principle behind both algorithms is to move the agent
to strategically chosen points in the current square, depending on previously obtained hints, and sometimes
perform exhaustive search of small rectangles from these points, in order to guarantee that the treasure is
not there. This is done in such a way that, in a given phase, obtained hints together with small rectangles
exhaustively searched, eliminate a sufficient area of the square assumed in the phase to eventually permit
finding the treasure.

In both algorithms, the points to which the agent travels and where it gets hints are chosen in a natural
way, although very differently in each of the algorithms. The main difficulty is to prove that the distance
travelled by the agent is within the promised cost. In the case of the first algorithm, it is possible to cheaply
exclude large areas not containing the treasure, and thus find the treasure asymptotically optimally. For the
second algorithm, the agent eliminates smaller areas at each time, due to less precise hints, and thus finding
the treasure costs more. Due to the lack of space, Section 2 only describes the high level idea of the second
algorithm (working with angles at most $\beta < 2\pi$) that we will call TreasureHunt.

1.3 Related work

The problem of treasure hunt, i.e., searching for an inert target by one or more mobile agents was inves-
tigated under many different scenarios. The environment where the treasure is hidden may be a graph or a
plane, and the search may be deterministic or randomized. One major and famous result about this problem
is the early paper [BN70] showing that the best competitive ratio for deterministic treasure hunt on a line
is 9. Another important result is the work [Lan10] in which the author presents 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 $OP$. For the curious reader wishing to consider
the problem of treasure hunt in greater depth, a good starting point is to go through [AG03, BCR93].

2 High level idea of Algorithm TreasureHunt

In order to properly present the intuition of our algorithm, we first need to provide two precisions. First,
a hint given to the agent currently located at point $a$ is formally described as an ordered pair $(P_1, P_2)$ of
half-lines originating at $a$ such that the angle clockwise from $P_1$ to $P_2$ (including $P_1$ and $P_2$) contains
the treasure. For a hint $(P_1, P_2)$ we denote by $(\overline{P_1}, \overline{P_2})$ the complement of $(P_1, P_2)$. Second, our algorithm heavily
relies on the following notion of tiling. Given a square $S$ with side of length $x > 0$, $Tiling(i)$ of $S$, for any
non-negative integer $i$, is the partition of square $S$ into $4^i$ squares with side of length $\frac{x}{4^i}$. Each of these
squares, called tiles, is closed, i.e., contains its border, and hence neighboring tiles overlap in the common
border. Having precised that, we are now ready to give the high level idea of our solution.

When executing Algorithm TreasureHunt, the agent proceeds in phases $j = 1, 2, 3, \ldots$, where in each
phase $j$ the agent “supposes” that the treasure is in the straight square centered at its initial position and
of side length $2^j$. The intended goal is to search each supposed square at relatively low cost, and to ensure
the discovery of the treasure by the time the agent finishes the first phase for which the initially supposed
square contains the treasure.

Let us consider a simpler situation in which the angle of every hint $(P_1, P_2)$ is always equal to the bound
$\beta$: the general case, when the angles may vary while being at most $\beta$, adds a level of technical complexity
that is unnecessary to understand the intuition. In the considered situation, the angle of each excluded zone
$(\overline{P_1}, \overline{P_2})$ is always the same as well. The following property holds in this case: there exists an integer $i_0$p
such that for every square $S$ and every hint $(P_1, P_2)$ given at the center of $S$, at least one tile of $Tiling(i_0)$ of $S$
belongs to the excluded zone $(\overline{P_1}, \overline{P_2})$. 
Trouver un trésor plus rapidement avec des conseils angulaires

In phase $j$, the agent performs $k$ steps: we will indicate later how the value of $k$ should be chosen. At the beginning of the phase, the entire square $S$ is white. In the first step, the agent gets a hint $(P_1, P_2)$ at the center of $S$. By the above property, we know that $(P_1, P_2)$ contains at least one tile of $Tiling(i_β)$ of $S$, and we have the guarantee that such a tile cannot contain the treasure. All points of all tiles included in $(P_1, P_2)$ are painted black in the first step. This operation does not require any move, as painting is performed in the memory of the agent. As a result, at the end of the first step, each tile of $Tiling(i_β)$ is either black or white. These steps permit the agent to exclude some area without having to search it directly, while keeping some regularity of the shape of the black area.

In the second step, the agent repeats the painting procedure at a finer level. More precisely, the agent moves to the center of each white tile $t$ of $Tiling(i_β)$ of $S$. When it gets a hint at the center of a white tile $t$, there is at least one tile of $Tiling(i_β)$ of $t$ that can be excluded. As in the first step, all points of these excluded tiles are painted black. Note that a tile of $Tiling(i_β)$ of $t$ is actually a tile of $Tiling(2i_β)$ of $S$. Moreover, each tile of $Tiling(i_β)$ of $S$ is made of exactly $4^β$ tiles of $Tiling(2i_β)$ of $S$. Hence, as depicted in Figure 1, the property we obtain at the end of the second step is as follows: each tile of $Tiling(2i_β)$ of $S$ is either black or white.

![Figure 1: White and black tiles at the end of the first and the second step of a phase, for square $S = ABCD$ and $i_β = 2$.](image)

In the next steps, the agent applies a similar process at increasingly finer levels of tiling. More precisely, in step $2 < s \leq k$, the agent moves to the center of each white tile of $Tiling((s-1)i_β)$ of $S$ and gets a hint that allows it to paint black at least one tile of $Tiling(s \cdot i_β)$ of $S$. At the end of step $s$, each tile of $Tiling(s \cdot i_β)$ of $S$ is either black or white. We can show that at each step $s$ the agent paints black at least $\frac{1}{4^β}$-th of the area of $S$ that is white at the beginning of step $s$.

After step $k$, each tile of $Tiling(k \cdot i_β)$ of $S$ is either black or white. These steps permit the agent to exclude some area without having to search it directly, while keeping some regularity of the shape of the black area. The agent paints black a smaller area than excluded by the hints but a more regular one. This regularity enables in turn the next process in the area remaining white. Indeed, the agent subsequently executes a brute-force searching that consists in moving to each white tile of $Tiling(k \cdot i_β)$ of $S$ in order to scan it using a procedure called RectangleScan: for every tile of area $A$, this procedure permits the agent to see all points of the tile at cost $O(A)$. If, after having scanned all the remaining white tiles, it has not found the treasure, the agent repaints white all the square $S$ and enters the next phase. Thus we have the guarantee that the agent finds the treasure by the end of phase $\lceil \log_2 D \rceil + 1$, i.e., a phase in which the initially supposed square is large enough to contain the treasure.

At this point, a natural question arises: how much do we have to pay for all of this?

In fact, the cost depends on the value that is assigned to $k$ in each phase $j$. The value of $k$ must be large
enough so that the distance travelled by the agent during the brute-force searching is relatively small. At the same time, this value must be small enough so that the the distance travelled during the $k$ steps is not too large. A good trade-off can be reached when $k = \lceil \log_{4/\beta} \sqrt{2} \rceil$. Indeed, we show that it is due to this carefully chosen value of $k$ that we can beat the cost $\Theta(D^2)$ necessary without hints, and get a complexity of $O(D^{2-\varepsilon})$, where $\varepsilon = \frac{1}{2}(1 - \log_{4/\beta} (4/\beta - 1))$ is a positive real depending on $i_\beta$, and hence depending on the angle $\beta$.

3 Conclusion

Let us now give some concluding remarks on our work [BDPP18]. For hints that are angles at most $\pi$ we gave a treasure hunt algorithm with optimal cost linear in $D$. For larger angles we showed a separation between the case where angles are bounded away from $2\pi$, when we designed an algorithm with cost strictly subquadratic in $D$ (the intuition of which is described in Section 2 of this paper), and the case where angles have arbitrary values smaller than $2\pi$, when we showed a quadratic lower bound on the cost.

The optimal cost of treasure hunt with large angles bounded away from $2\pi$ remains open. In particular, the following questions seem intriguing:
— Is the optimal cost linear in $D$ in this case, or is it possible to prove a super-linear lower bound on it?
— Does the order of magnitude of this optimal cost depend on the bound $\pi < \beta < 2\pi$ on the angles given as hints?

Références


