International Journal of Foundations of Computer Science
(2021) 1–19

DOI: 10.1142/S0129054121500209

Searching on a Line by Byzantine Robots*

Jurek Czyzowicz
Département d'informatique, Université du Québec en Outaouais
Gatineau, QC, Canada
jurek.czyzowicz@uqo.ca

Konstantinos Georgiou
Department of Mathematics, Ryerson University
Toronto, ON, Canada
konstantinos@ryerson.ca

Evangelos Kranakis
School of Computer Science, Carleton University
Ottawa, ON, Canada
kranakis@scs.carleton.ca

Danny Krizanc†
Department of Mathematics and Computer Science
Wesleyan University, Middletown CT, USA
dkrizanc@wesleyan.edu

Lata Narayanan‡ and Jaroslav Opatrny§
Department of Computer Science and Soft. Engineering
Concordia University, Montreal, QC, Canada
‡lata@cs.concordia.ca
§opatrny@cs.concordia.ca

Sunil Shende
Department of Computer Science
Rutgers University, Camden, NJ, USA
sunil.shende@rutgers.edu

Received 3 August 2018
Accepted 10 December 2020
Published 13 January 2021
Communicated by Marek Chrobak

*This is an extended and revised version of a paper which appeared in the proceedings of ISAAC 2016, December 12–14, 2016, Sydney, Australia.
†Corresponding author.

We consider the problem of fault-tolerant parallel search on an infinite line by n robots. Starting from the origin, the robots are required to find a target at an unknown location. The robots can move with maximum speed 1 and can communicate wirelessly among themselves. However, among the n robots, there are f robots that exhibit *byzantine faults*. A faulty robot can fail to report the target even after reaching it, or it can make malicious claims about having found the target when in fact it has not. Given the presence of such faulty robots, the search for the target can only be concluded when the non-faulty robots have sufficient evidence that the target has been found. We aim to design algorithms that minimize the value of $S_d(n, f)$, the time to find a target at a (unknown) distance d from the origin by n robots among which f are faulty. We give several different algorithms whose running time depends on the ratio f/n, the density of faulty robots, and also prove lower bounds. Our algorithms are optimal for some densities of faulty robots.

Keywords: Cow path problem; parallel search; wireless communication; mobile robots; byzantine faults.

1. Introduction

Searching is central to many areas of computer science, including data structures, computational geometry, and artificial intelligence. The linear search (also known as the *cow-path*) problem is concerned with a robot, starting at the origin, looking for a target placed at an unknown location on the infinite line. The robot moves with uniform (i.e., constant) speed and can change direction (at any time) without any loss in time. The ultimate goal is to find the target in optimal time [5]. Historically speaking, the problem was first posed by Bellman [12] and independently considered in 1964 by Beck [7]. In their formulation, the target was placed according to a known probability distribution on the real line and the goal was to find the target in minimum expected time.

In this paper, we consider the problem of *parallel, cooperative search* for a target on the infinite line by n mobile robots at most f of which are faulty. As before a target is placed at a location unknown to the robots. The robots start at the same time and location and can communicate wirelessly and instantaneously at any distance. During the search the robots may cooperate by broadcasting messages. The search may be impeded by the faulty robots which may exhibit byzantine faults, including failing to report having reached the target or making malicious claims about having found the target when in fact they have not. The ultimate goal is to minimize the time it takes all non-faulty robots to be certain that the correct location of the target has been found.

1.1. *Motion and communication models*

To begin, we describe the robots' locomotive, fault and communication models used in a search algorithm.

Robots and their trajectories. Robots are assumed to start at a common location, considered to be the origin of the line. They can move at maximum speed one either along the positive direction (denoted as moving *right*) or along the negative direction (denoted as moving *left*). A robot can change direction arbitrarily often (by *turning*) without loss in time. An algorithm for parallel search specifies a trajectory for each robot, given by its turning points, and the speed(s) to follow between turning points. Since each robot has a distinct identity, it may also follow a distinct trajectory. Robots are assumed to know each other's trajectories and moreover can communicate with each other wirelessly and instantaneously at any distance. Since robots know about each other's trajectories, the only kind of message broadcast by a robot R is whether or not it has found the target at some location. If R says nothing while visiting a location, the implicit assumption made by the other robots is that R did not detect the target. The robot R follows its predefined trajectory until either it finds the target, in which case it announces that it has found the target, or it hears some other robot R' announce that it has found the target at which point R may change its trajectory to participate in a verification protocol in regard to the announcement.

Faults and communication. All n robots know that f among them are faulty but they do not know in advance which are the f ones that are faulty. In particular, a non-faulty robot does not know which robots are non-faulty; instead they must identify faulty robots from non-faulty ones based on conflict resolution and verification of messages received through the communication exchanges taking place during the execution of the search protocol. To this end, robots are equipped with distinct identities which they cannot alter at any time. In addition to the correct identity, the current location of a robot is automatically included in any broadcast message sent by the robot. Consequently, a faulty robot that does not follow its assigned trajectory can be immediately identified as faulty by the other robots, if it chooses to broadcast at a point not on its assigned trajectory. In all other ways, faulty robots are indistinguishable from non-faulty robots, except that the former can make both positive and negative detection errors as follows. A non-faulty or reliable robot never lies when it has to confirm or deny the existence of the target at some location. In contrast, a faulty robot *may stay silent* even when it visits the target and *may also claim maliciously that it has found the target* when it has not found it. Thus, a reliable robot cannot necessarily trust an announcement that the target has been found (not even its own such announcement) nor can it be certain that a location that has been silently visited by another robot, does not contain the target. In other words, the search for a target can terminate only after at least one robot that is *provably reliable* has visited and announced that it has found it. This requirement is critical to all our algorithms: if at some time, multiple robots make conflicting announcements at a location then the resulting votes can only be resolved if something is known about the number of reliable robots that could participate in the vote. For instance, if three robots vote and it is known that two of them are reliable, then the majority vote would provably be the truth.

1.2. *Preliminaries and notation*

Consider a parallel search algorithm for a target located at distance d from the origin. In what follows, we restrict our attention to algorithms that are guaranteed to eventually find the target no matter where it is placed on the line. In particular, we assume that every point on the line is eventually visited by at least $2f+1$ robots so that a proper identification of the target is certain to occur. First we define the search time of an algorithm of this type and its corresponding competitive ratio.

Definition 1 (Search Time). Let $S_d(n, f)$ denote the minimum over all possible search algorithms, using a collection of n robots of which f are faulty, of the worst case time (over all possible behaviors of the byzantine robots) to find the target placed at a distance d unknown to the robots from their common starting position, namely, the origin of the line.

Definition 2 (Competitive Ratio). The corresponding *competitive ratio* is defined as $S_d(n, f)/d$ which is the ratio of the algorithm's search time and the lower bound d on the time taken by any algorithm for the problem.

For large values of n, f, it will be more convenient to express our results in terms of the density $\frac{f}{n}$ of faulty robots. This leads to the following definition.

Definition 3 (Asymptotic Competitive Search Ratio). Extend the definition of $S_d(n, f)$ above to non-integer values by replacing n with $\lceil n \rceil$ while the parameter f remains integral. Let $\beta = \frac{f}{n}$. Then

$$\hat{S}(\beta) = \inf\{\alpha \mid \exists \text{ constant } c_\beta \text{ such that } \forall f, S_d(f/\beta + c_\beta, f) \leq \alpha d\} \quad (1)$$

denotes the asymptotic competitive search ratio of any algorithm with search time $S_d(n, f)$.

Note that if $n \geq 4f+2$, then in any partition of the robots into two groups each of size at least $2f+1$ we will always have $f+1$ reliable robots per group. Therefore, an algorithm that sends the corresponding robots in the two groups in opposite directions is guaranteed to find the target since when the target is visited by one of the groups, a straightforward majority vote in the group confirms its presence reliably. $S_d(4f+2, f) = d$, which is optimal. On the other hand, if $n \leq 2f$, there is no algorithm to complete the search: the faulty robots may always completely disagree with the reliable ones, making it impossible to be certain of the location of the target. Therefore, in the sequel, we examine the interesting case where $2f+1 \leq n \leq 4f+1$.

1.3. *Our results*

In Sec. 2, we are concerned mostly with upper bounds. Section 2.1 establishes the guiding principles for the design of our algorithms. We begin our study of upper

Table 1. Upper and lower bounds on the search time $S_d(n, f)$ for a given number $n \le 6$ of robots and faults $f = 1, 2$.

n, f	Upper Bound	Lower Bound
3, 1	$9d$	$5.23d$
4, 1	$3d$	$3d$
5, 1	$2d$	$2d$
6, 1	d	d
5, 2	$9d$	$4.43d$
6, 2	$(2+\sqrt{5})d$	$3d$

Table 2. Upper and lower bounds on the asymptotic competitive search ratio $\hat{S}(\beta)$ for various ranges of the density β. Note that for $\beta > \frac{1}{2}$ the search problem is impossible to solve.

Range of β	Upper Bound	Lower Bound
$(0, \frac{1}{4}]$	1	1
$(\frac{1}{4}, \frac{3}{10}]$	2	2
$(\frac{3}{10}, \frac{1}{3}]$	3	2
$(\frac{1}{3}, \frac{5}{14}]$	3	3
$(\frac{5}{14}, \frac{13}{34}]$	4	3
$(\frac{13}{34}, \frac{19}{46}]$	5	3
$(\frac{19}{46}, \frac{47}{110}]$	6	3
$(\frac{47}{110}, \frac{65}{146}]$	7	3
$(\frac{65}{146}, \frac{157}{346}]$	8	3
$(\frac{157}{346}, \frac{1}{2}]$	9	3

bounds in Sec. 2.2 by establishing $S_d(n, f)$ for specific small values of n and f. These results are summarized in Table 1. For comparison, we include in the table lower bounds for these cases. The lower bounds for the cases (4, 1), (5, 1) and (6, 2) follow from results in Sec. 3. The lower bounds for the cases (3, 1) and (5, 2) follow from the lower bounds in the crash fault model (described below) shown in [28]. The lower bound for (6, 1) is the trivial lower bound.

For larger values of n and f we express our results in terms of the density $\beta = \frac{f}{n}$. In Sec. 2.3 we show how to extend our algorithms for small values of n and f to the general setting. Table 2 summarizes our results from Sec. 2.3. In Sec. 3, we discuss lower bounds. In particular, we show that our results in $(0, \frac{3}{10}]$ and $(\frac{1}{3}, \frac{5}{14}]$ are tight.

1.4. Related work

A search problem is usually seen as the problem of locating a hidden target using searchers capable of moving in an environment. It is an optimization question, usually attempting to minimize the time needed to complete the search. The question has been studied in numerous settings involving static or moving targets, one or many searchers, known or unknown environments, synchronous or asynchronous timing, different speeds, agents, as well as many other variations (cf. [23]).

In several studies, when the environment is not known in advance, search implies exploration, often involving mapping and localizing searchers within the environment [2, 3, 21, 24, 26, 31]. However, even for the case of a known, simple environment like a line, there were several interesting studies attempting to optimize the search time. These started with the independent works of Bellman [12] and Beck [7] which the authors attempted to minimize the competitive ratio. They proved that time $9d$ is needed to guarantee finding a target at a (a priori unknown) distance d from the origin. Several other works on linear search followed (e.g., see [4, 7, 12]). The search by a single searcher has been studied for many different models, e.g., when the turn cost was considered [20], when bounds on the distance to the target are known in advance [14], or for more general cost functions [13].

Recent variants of linear search were studied for collections of collaborating searchers (robots). The authors in [16] considered linear group search, when the process is complete when the target is reached by the last robot visiting it. The robots collaborate by attempting to minimize the group search time. However, [16] shows that having many robots does not help and the optimal search time is still bounded from below by $9d$ in the face-to-face communication model. Group search using a pair of robots having distinct maximal speeds was studied in [6], in which techniques producing optimal search time were designed.

Fault tolerance has been studied in distributed computing in various settings in the past (e.g., see [25, 29, 30]). The subject of unreliability was mainly considered for static components of the environment (e.g., network nodes or links), which are sometimes modeled by dynamically evolving environments (cf. [15, 27]). Malfunctions arising in mobile robots were investigated for the problems of gathering or pattern forming [1, 17, 22, 32] or patrolling [18, 19]. In [11], Czyzowicz et al. investigated the problem of performing linear search in the presence of robots subject to crash faults. In this setting, a faulty robot may or may not detect the target while visiting its location but they may not report finding the target at a location not containing the target, i.e., they may only fail by not reporting having found the target. Their upper bounds were shown to be optimal by Kupavskii and Welzl [28] who also consider faulty search on multiple rays. In general, dealing with Byzantine agents can be more involved, requiring algorithms to identify and to refute the most malicious adversarial behaviour (e.g., see [22]).

2. Upper Bounds

As already observed, if $n \geq 4f + 2$, linear search can be performed optimally in time d, and no algorithm exists if $n \leq 2f$. Therefore, we consider below the case when $2f + 1 \leq n \leq 4f + 1$. Clearly, the robots can always stay together as a group, and perform the doubling zig-zag strategy of Bellman and Beck that is optimal for a single robot and then attempt to reach it; see Section 5. Since the reliable robots are always in a majority, we are guaranteed to find the target.

In the remainder of this section, first we give the main guiding principles used in the design of algorithms for n robots of which at most f are faulty, and then describe upper bounds.

2.1. *Algorithm design principles*

The general framework for our algorithms involves five basic principles, namely *Partition into groups*, *Symmetry of algorithms*, *Resolution of conflicts*, *Simultaneous announcements*, and *Computations by the robots*, which we describe below.

Partition into groups. Depending on the ratio of faulty robots, we partition the robots into a certain number of groups. Two of the groups lead the exploration in opposite directions from the origin. Further, when possible, each of these two groups will have at least $f + 1$ robots so that a target would be announced if the target is reached.

Symmetry of algorithms. The algorithms are symmetric as far as left and right part of the line is concerned. We therefore typically discuss the behaviour of the algorithm with respect to one side of the line only.

Resolution of conflicts. If at any time there is an announcement of a target, the robots in the search groups stop until the claim is resolved. In the meantime, robots from some other group(s) move to the claim. Once the claim is resolved, either the target is found and the robots stop, or a certain number of faulty robots is identified. From that point the algorithm disregards any message from these faulty robots, effectively reducing the number of faulty robots to contend with, and the groups continue the search. Thus, each such announcement exposes more faulty robots, until eventually we can be certain that a majority of robots in each search group are reliable and the remaining search can be correctly completed.

Simultaneous announcements. When two announcements are made at the same time the algorithm deals only with one of them at a time, chosen arbitrarily. After the resolution of the first announcement the search is possibly restarted, the robots redo their observation, and then the announcement is repeated if needed, thus taking into consideration the situation after the resolution of the first announcement. We show this does not influence the search time.

Computations by the robots. We assume that the time spent on calculations is negligible in comparison with the time spent in moving. Thus, we count only the time needed in movements of the robots until the target is found.

As indicated above, throughout the execution of the algorithms conflicts will be resolved by voting. More precisely, we define $V(x)$ to be the robots about position x at time t. If y robots have claimed that the target is at x before time t, while z have claimed (by visiting and keeping silent) that it is not at x, then we say $V(x,t) = (y,z)$.

The following two simple observations are used extensively in the proofs below.

Lemma 4. *Let $V(x,t) = (y,z)$, and let f be the number of faulty robots before time t. Then*

(1) *If $y > f$ then the target is at position x and the search is concluded.*
(2) *If $z > f$ then the target is not at position x and y new faulty robots have been identified at time t.*

Lemma 5. *Assume $d > x \geq 0$. Suppose at time t, there are f faulty robots remaining, and there are at least $2f+1$ robots at positions $\geq x$ and $< d$ and at least $2f+1$ robots at positions $\leq -x$ and $> -d$. Then any target that is distance d from the origin can be found by time $t + (d - x)$.*

Lemma 4 suggests that the interesting cases are when y and z are non-zero and less than or equal to f. This motivates the following definition:

Definition 6 (Conflict). We say there is a *conflict* at position x at time t if $V(x,t) = (y,z)$, with $0 < y \leq f$ and $0 < z \leq f$.

To warm up and build intuition, we start with giving algorithms for the cases of one or two faulty robots, and later show how to use these techniques to give algorithms for general values of n and f.

2.2. Algorithms for small f and n

Note that for the case $f = 1$, we only need to consider the case of 3, 4, and 5 robots.

Proposition 7. $S_d(4,1) \leq 3d$.

Proof. Partition all robots into two search groups, L and R, with two robots in each group. Each robot in R (L) moves right (left, resp.) at speed 1 until it finds the target or hears an announcement that the target has been found. Suppose now that there is an announcement at time x from position $x > 0$. If $V(x,x) = (2,0)$, by Lemma 4, the target has been found at x and the algorithm terminates. Suppose that $V(x,x) = (1,1)$. Then one of the robots in L, say A, travels to x to resolve the conflict, taking additional time $2x$, while all other robots remain stationary. At time $3x$, the robot A reaches x. If $V(x,3x) = (2,1)$, by Lemma 4, the target has been found, and the algorithm terminates. If instead $V(x,3x) = (1,2)$, then by Lemma 4, the faulty robot is identified, and all other robots can be inferred to be reliable. Now the search continues with the groups moving in opposite directions with only the reliable robots being considered, until the target is found. Notice that an announcement at $-x$, simultaneous with that at x, would be resolved at time with reliable robots. Therefore, if the target is at distance d, the time taken to find it is $\leq 3x + d - x = 2x + d \leq 3d$ since $d \geq x$. Thus in all cases, $S_d(4,1) \leq 3d$. □

By increasing the total number of robots, we can achieve a better search time as shown below:

Proposition 8. $S_d(5,1) \leq 2d$.

Proof. Partition the robots into three groups L, A and R being the search groups. There are two robots L moving left at speed 1, one robot, designated A for convenience, remains stationary at the origin, and one search group of two robots R moving right at speed 1. Suppose that there is an announcement at time x from position $x > 0$. If $V(x, x) = (2, 0)$, then by Lemma 4, the search terminates. If instead $V(x, x) = (1, 1)$, then robot A joins R at position x, taking additional time x to resolve the conflict, while all other robots remain stationary. If $V(x, 2x) = (2, 1)$ by Lemma 4 the target has been found, and the algorithm terminates. If instead $V(x, 2x) = (1, 2)$, then by Lemma 4, the faulty robot is identified, and all remaining robots can be inferred to be reliable. Now each group L and R with reliable robots continues moving in opposite directions until the target is found. Notice that an announcement at $-x$, simultaneous with that at x, would be resolved at time $2x$ with reliable robots. If the target is at distance d, the time taken to find it is $\leq 2x + d - x = x + d \leq 2d$ since $d \geq x$. Thus in all cases, $S_d(5,1) \leq 2d$. □

For the case of 1 faulty robot among 3 robots, we conjecture that the best worst case search time for a target at distance d is the zig-zag algorithm with running time equal to $9d$ (see [5]). Similarly for the case of 2 faulty robots among 5 robots, the best algorithm we know is the zig-zag algorithm with competitive ratio 9. This leads us to conjecture that in general, for the case of f out of $2f+1$ robots being faulty, the competitive ratio is 9.

For the case of $f = 2$ and $n = 6$ we present a search algorithm in which, unlike our other algorithms, the movement of robots among the groups in order to resolve a conflict is delayed past the announcement of the conflict. Surprisingly, this provides the best known algorithm for this case. (A bound of $5d$ for this case is easily derived using the techniques described below.)

Theorem 9. $S_d(6,2) \leq (2 + \sqrt{5})d$.

Proof. In this search algorithm we partition the robots into two groups of size three and have the two search groups move at speed 1 from the origin in opposite directions. Assume that when the groups reach distance x from the origin an announcement of a target is sent from x. We only need to consider the case when the vote is $V(x, x) = (2, 1)$ or $(1, 2)$. In either case the group at the right side contains at most 1 faulty robot and therefore has a majority of reliable robots. In this algorithm the group of robots at the right continue to explore the line further to the right for a distance to be determined later, in case a target is located between x and $x+s$. At time $3x+2s$ the group on the left sends 2 robots to join the group on the right, one of each type, while we send 2 robots from the group on the right to the left group. Now, at time $3x+2s$ we have sufficient number of robots on the left to determine which one of the two groups was correct, and at least one faulty robot sent to the right group is identified. Thus, at time $3x+2s$ the number of useful robots remaining on the right is 2, at most one of which is faulty. The number of robots remaining on the left is equal to 3, at most 1 of which is faulty. Thus the group on the left has a majority of reliable robots, and a conflict that needs additional resolution can arise only on the right side. The two groups resume their search. Now assume that a conflict occurs on the right at position $x+s$ with a split vote $(1,1)$, for $s>0$. Thus the last remaining faulty robot is in the right group. We send one robot from the left to the right. The additional time needed for this will be $2x+s$, with the current time being $5x+3s$. On the right we have identified the last faulty robot and the search on the right can also go on without a problem from this point on, while the left side contains a majority of reliable robots.

The target will either have been identified at position d at time $3x+2s+d-x$ or at time $5x+3s+d-x-s$. Clearly the worst case on the left occurs when $d=x$ and on the right when $d=x+s+\epsilon$, as the ratio improves as d becomes larger. Similarly, taking $\epsilon=0$ will maximize the ratio on the right. We equate the two corresponding competitive ratios to derive the equation:

$$\frac{3x+2s}{x}=\frac{5x+3s}{x+s}$$

From which we derive the quadratic equation $s^2+sx-x^2=0$, whose solution is $s=\frac{(-1+\sqrt{5})x}{2}$. The resulting competitive ratio is equal to $2+\sqrt{5}$. □

2.3. Algorithms for large n

We now consider the case of large n, with different values of the density of faulty robots f/n. We start by generalizing the results from the previous subsection, then build recursive techniques that allow us to deal with larger densities of faulty robots, while paying a price in terms of the search time.

Our first result generalizes Proposition 8.

Theorem 10. $S_d\left(\frac{10f+4}{3}, f\right) \le 2d$, *provided that* $f \equiv 2 \bmod 3$.

Proof. Partition the robots into three groups L, R, and C of sizes $\frac{4f+1}{3}$, $\frac{4f+1}{3}$ and $\frac{2f+2}{3}$ respectively. Robots in L and R are the search groups that move left and right at speed 1 respectively, and robots in C stay stationary until an announcement is made. Suppose there is an announcement at time x at position $x>0$, and let $V(x,x)=(y,z)$ with $y>0$. There are three possible cases. If $y \ge f$, then by

Lemma 11 at t_4, the search is concluded if $z \geq f$, the target is not at x, and y faulty robots have been identified. In this case, we disregard these robots from now on and the remaining robots continue moving in their original directions until a genuine conflict occurs (i.e., both y and z are less than the number of faulty robots remaining) or the target is found.

It remains to consider the case $y, z \leq f$. Observe that $\max\{y, z\} \geq \frac{f+1}{3}$. In this case, the algorithm moves the robots in C into location x. Consider now $V(x, 2x) = (y', z')$. Since at time $2x$ there are $2f + 1$ robots at x, either the search is concluded or at least $\frac{f+1}{3}$ faulty robots have been identified and may be disregarded.

Consider the remaining robots; there are $f' = f - y'$ faulty robots among them. There are $2f' + 1$ robots (after the faulty have been identified) remaining in the group $\frac{4f+1}{3}$ robots in group L. It is easy to verify that with $\frac{f+1}{3} \leq y' \leq f$, we have at least $2f' + 1$ robots in both L and R. Therefore, the robots in L and R can continue exploration of the line in their original directions and the time required to reach the target is at most $2d$. Observe that all the quantities mentioned are integral when $f \equiv 2 \mod 3$.

Using the fact that $S_d(n', f') \leq S_d(n, f)$ if $n' \geq n$ and $f' \leq f$ we can easily derive the following corollary:

Corollary 11. *If $\beta \leq \frac{3}{10}$ then $\hat{S}(\beta) \leq 2$.*

Proof. Let $\beta = \frac{3}{10} = \frac{f}{n}$ where f is an integer and $f \equiv x \mod 3$. Observe that for $x = 0, 1,$ or 2:

$$
\begin{aligned}
S_d(\lceil n \rceil + 8, f) &\leq S_d(\lceil n \rceil + 8, f + (2 - x)) \\
&= S_d\left(\left\lceil \frac{10f}{3} + 8 \right\rceil, f + (2 - x)\right) \\
&\leq S_d\left(\left\lceil \frac{10f}{3} + \frac{10(2 - x) + 4}{3} \right\rceil, f + (2 - x)\right) \\
&= S_d\left(\left\lceil \frac{10(f + (2 - x)) + 4}{3} \right\rceil, f + (2 - x)\right) \\
&\leq 2d,
\end{aligned}
$$

the last inequality follows from Theorem 10. Note that $f + (2 - x) \equiv 2 \mod 3$. Taking $c_{\frac{3}{10}} = 8$ in the definition of $\hat{S}$ we have $\hat{S}(\frac{3}{10}) \leq 2$ and the result follows from the fact that $\hat{S}(\beta) \leq \hat{S}(\beta')$ if $\beta \leq \beta'$. □

The following result generalizes Proposition 7.

Theorem 12. $S_d\left(\frac{14f+4}{5}, f\right) \leq 3d$ *provided* $f \equiv 4 \mod 5$.

Proof. Partition the robots into two search groups L and R each containing $\frac{7f+2}{5}$ robots. The robots in L move left and R move right at speed 1. Without loss of generality, assume there is an announcement at x. Let $V(x, x) = (y, z)$. If $\max\{y, z\} > f$, the announcement is resolved using Lemma 4. Suppose instead that $\max\{y, z\} \le f$. Then $\min\{y, z\} \ge \frac{2f+2}{5}$ and at least $\frac{2f+2}{5}$ robots at x are faulty. In this case, $\frac{3f+3}{5}$ robots from L move from $-x$ to x, and at the same time $\frac{2f+2}{5}$ robots that voted yes and $\frac{2f+2}{5}$ robots that voted no are sent from x to $-x$. At time $3x$, in total $2f+1$ robots have visited x, and by Lemma 4, either the target is identified, or at least $\frac{2f+2}{5}$ faulty robots are identified and may be disregarded from now on. The worst case occurs when exactly $\frac{2f+2}{5}$ robots have been identified as faulty. Identifying more faulty robots only improves the ratio of non-faulty to faulty robots. In this case, after the exchange of robots and the elimination of the faulty robots there are $\frac{6f+1}{5}$ robots in L and in R with $\frac{3f-2}{5}$ faulty robots remaining, i.e., a majority of reliable robots in both search groups. Therefore by Lemma 5, search for a target at distance d can be finished in time $d - x \le 3d$ as claimed. Note that all quantities are integral if $f \bmod 5 = 4$. □

As above, the following corollary is immediate:

Corollary 13. *If $\beta \le \frac{5}{14}$ then $\hat{S}(\beta) \le 3$.*

As illustrated in the proofs of Theorems 10 and 12, when an announcement of a target is made, either the target can be confirmed, or the number of unidentified faulty robots can be reduced either by an exchange of robots between two search groups or by moving a group of stationary robots to the potential target. For higher densities of faulty robots this technique can be repeated, for which we pay an increase in the search time. This is the motivation for the recurrence formulas below that are used to obtain search algorithms for higher densities of robots.

Definition 14. Assume $d > x > 0$. Let $T_x(l, s, r, f)$ be the minimum over all algorithms of the search time required by the robots to find the target given that initially l robots are located at $-x$, s robots are at the origin 0, r robots are at $+x$, and at most f robots are faulty.

The following observation is used extensively in the proofs below:

Lemma 15. *$T_x(l, s, r, f) \ge T_x(l', s', r', f')$ if $l' \ge l$ and $s' \ge s$ and $r' \ge r$ and $f' \le f$.*

As we see in the algorithms described so far, one way to solve our search problem is to send two equal-sized groups of robots at full speed to positions x and $-x$ where a conflict occurs, from which we get the following upper bound.

Lemma 16. *Let $d \ge x > 0$ and suppose the first conflict occurs at distance x from the origin. Then $S_d(n, f) \le x + T_x(\lfloor n/2 \rfloor, 0, \lfloor n/2 \rfloor, f)$. Furthermore, if $\lfloor n/2 \rfloor \ge 2f + 1$ then $T_x(\lfloor n/2 \rfloor, 0, \lfloor n/2 \rfloor, f) = d - x$.*

If there is an announcement at x, we can identify some of the faulty robots, and by paying a price in terms of additional time, we can reduce it to a new problem with a smaller number of faulty robots. This can be encapsulated in the following lemma:

Lemma 17. *Let k be any even integer such that $f \geq k > 0$. Suppose there is conflict at distance x from the origin. Then*

$$T_x(f+k, 0, f+k, f) \leq 2x + T_x(f+k/2, 0, f+k/2, f-k).$$

Proof. Assume there are $f+k$ robots each at x and $-x$, with at most f faulty robots in all, and that a conflict occurs at $x > 0$. Let $V(x,x) = (y,z)$. If a conflict has occurred it must be that $k \leq \min\{y,z\} \leq \max\{y,z\} \leq f$. Now the robots move as follows:

(1) All $f+k$ robots at position x move to $-x$.
(2) $f+k/2$ of the robots at $-x$ move to x.

Note that these movements take time $2x$, and there are now $f+k/2$ robots at x and $f+k/2+k$ robots at $-x$. Since $3f/2+3k/2$ robots have now visited x, the vote $V(x,3x)$ is enough to resolve the conflict. Observe that at least $\min\{y,z\} \geq k$ robots in $\{y,z\}$ are identified as faulty so that there are at most $f-k$ faulty robots. After eliminating k of the faulty robots that have been identified, $f+k/2$ robots remain at both x and $-x$. □

Lemmas 15, 16 and 17 along with Theorem 12 can be used to obtain slower algorithms for higher densities. We have

Theorem 18. (1) $\hat{S}(\beta) \leq 5$ *for* $\beta \leq 19/4$.
(2) $\hat{S}(\beta) \leq 7$ *for* $\beta \leq 65/14$.

Proof. An examination of the proof of Theorem 12 shows that it may be reformulated using the following observations:

(a) For $x \leq d$,

$$S_d\left(\frac{14f+4}{5}, f\right) \leq x + T_x\left(\frac{7f+2}{5}, 0, \frac{7f+2}{5}, f\right);$$

(b) Suppose that a conflict occurs at $x \leq d$. Then

$$T_x\left(\frac{7f+2}{5}, 0, \frac{7f+2}{5}, f\right) \leq 2x + T_x\left(\frac{6f+1}{5}, 0, \frac{6f+1}{5}, \frac{3f-2}{5}\right);$$

(c) For $x \leq d$,

$$T_x\left(\frac{6f+1}{5}, 0, \frac{6f+1}{5}, \frac{3f-2}{5}\right) = d - x.$$

Points (a) and (c) above follow from Lemma 16 and point (b) follows from Lemma 17 taking $k = \frac{2f+2}{5}$. Since an announcement must occur either before or at time d, together with the lemma $S_d\left(\frac{14f+4}{5}\right) \le x + 2x + d - x$ for some $x \le d$. To extend this to higher densities we apply Lemma 17 multiple times.

Consider f that is a multiple of 19 and let $n = \frac{46f}{19} + 12$. By Lemma 17 we may conclude that if there is a conflict at some time x, then

$$T_x\left(\frac{23f}{19}+6, 0, \frac{23f}{19}+6, f\right) \le 2x + T_x\left(\frac{21f}{19}+3, 0, \frac{21f}{19}+3, \frac{15f}{19}-6\right)$$

taking $k = \frac{4f}{19} + 6$. Observe that $\frac{15f}{19} - 6 \equiv 4 \mod 5$, $\frac{7(\frac{15f}{19}-6)+2}{5} = \frac{21f}{19} - 8$ and by Lemma 15

$$T_x\left(\frac{21f}{19}+3, 0, \frac{21f}{19}+3, \frac{15f}{19}-6\right) \le T_x\left(\frac{21f}{19}-8, 0, \frac{21f}{19}-8, \frac{15f}{19}-6\right)$$

holds. That is after a single movement, we have reduced our problem to one we may solve using the approach of Theorem 12. From the theorem and from this we can conclude for f a multiple of 19 and $n = \frac{46f}{19}+12$, if conflicts occur at $x_0 \le x_1 \le d$ we have

$$S_d\left(\frac{46f}{19}+12, f\right) \le x_0 + 2x_0 + (x_1 - x_0) + 2x_1 + (d - x_1) \le 5d,$$

from which it follows that $\hat{S}(\beta) \le 5$ for $\beta \le 19/46$.

To show $\hat{S}(\beta) \le 7$ for $\beta \le 65/146$ we apply the recurrence one more time to reduce to the case $\hat{S}(\beta) \le 5$ just considered. Assume f is divisible by $19 * 65 = 1235$ and let $n = \frac{146f}{65} + 78$. By Lemma 17 we may conclude that if there is a first conflict at some time $x \le d$, then

$$T_x\left(\frac{73f}{65}+38, 0, \frac{73f}{65}+38, f\right) \le 2x + T_x\left(\frac{69f}{65}+19, 0, \frac{69f}{65}+19, \frac{57f}{65}-38\right)$$

taking $k = \frac{8f}{65} + 38$. Observe that $\frac{57f}{65} - 38$ is divisible by 19 and that $\frac{23}{19}\left(\frac{57f}{65}-38\right)+6 = \frac{69f}{65} - 40 < \frac{69f}{65} + 19$.

Thus if conflicts occur at $x_1 \le x_2 \le d$ we have

$$S_d\left(\frac{146f}{65}+78, f\right) \le 2x_0 + 2x_1 + 2x_2 + d \le 7d$$

which proves the second part of the theorem. □

In a similar manner as Lemma 17, the following lemma establishes a recurrence that can be used to extend Theorem 10, using the bounds from Theorems 12 and 18, to higher densities (at a cost of a higher competitive ratio).

Lemma 19. *Assume f is divisible by 3 and $k \le \frac{4f}{13}$. Suppose there is a conflict at distance x from the origin. Then*

$$T_x(f+k+1, 0, f+k+1, f) \le 2x + T_x\left(3k-1, \frac{3k}{2}, 3k-1, f-k-1\right).$$

Proof. Assume there are $f+k+1$ robots at each of x and d, with at most f faulty robots in all, and that a conflict occurs at time x. Let $V(x,t)=(y,z)$. Then we must have $k+1 \le \min\{y,z\} \le \max\{y,z\} \le f$. Now the robots move as follows:

(1) $f-k+1$ robots move from x.
(2) $2k+2$ robots move from x. These consist of $k+1$ among those that voted the target is at x and $k+1$ among those that voted the target is not at x.
(3) $\frac{3k}{2}$ move from x to 0.

Note that these actions take $2x$ time, and it is clear that there are $f+k+1-(f-k+1)+2k+2-4k-2$, $\frac{3k}{2}$, $2f+k+1+(f-k+1)+(2k+2)-\frac{3k}{2}-2 = 2f-\frac{7k}{2} = \frac{13k}{2}-\frac{7k}{2} = 3k$ robots at x, 0, and d, respectively. Once the robots from x have relocated to d, they vote. Their votes add up to a total of $2f+2$ and therefore the conflict can be resolved.

Furthermore at least $\min\{y,z\} \ge k+1$ faulty robots have been identified, so the number of faulty robots that remain is at most $f-k-1$. By removing $k+1$ of the identified robots at x, we are left with $3k-1$ robots at $d-x$ and $\frac{3k}{2}$ at 0 and the lemma follows by Lemma 15. □

Using an analogous argument to that used in Theorem 18 we can apply Lemmas 17 and 19 and Theorem 10 to get:

Theorem 20. (1) $\hat{S}(\beta) \le 4$ for $\beta \le 13/3$.
(2) $\hat{S}(\beta) \le 6$ for $\beta \le 47/11$.
(3) $\hat{S}(\beta) \le 8$ for $\beta \le 157/34$.

Proof. Examining the proof of Theorem 10 we see that it comes down to showing that for $f \equiv 2 \bmod 3$:

(a) For $x \le d$,

$$S_d\left(\frac{10f+4}{3}, f\right) \le x + T_x\left(\frac{4f+1}{3}, \frac{2f+2}{3}, \frac{4f+1}{3}, f\right);$$

(b) Suppose that a conflict occurs at $x \le d$. Then:

$$T_x\left(\frac{4f+1}{3}, \frac{2f+2}{3}, \frac{4f+1}{3}, f\right) \le x + T_x\left(\frac{4f+1}{3}, 0, \frac{4f+1}{3}, \frac{2f-1}{3}\right);$$

(c) For $x \le d$,

$$T_x\left(\frac{4f+1}{3}, 0, \frac{4f+1}{3}, \frac{2f-1}{3}\right) \le d - x.$$

Points (a) and (c) follow from Lemma 16 and point (b) from the discussion in the proof of Theorem 10.

Consider f divisible by 39 and $d = \frac{34f}{13} + 2$. By Lemma 16 we have for any $x \le d$:

$$S_d\left(\frac{34f}{13} + 2, f\right) \le x + T_x\left(\frac{17f}{13} + 1, 0, \frac{17f}{13} + 1, f\right).$$

By Lemma 19 we may conclude that if there is a first conflict at some $x \le d$, then

$$T_x\left(\frac{17f}{13} + 1, 0, \frac{17f}{13} + 1, f\right) \le 2x + T_x\left(\frac{12f}{13} - 1, \frac{6f}{13}, \frac{12f}{13} - 1, \frac{9f}{13} - 1\right)$$

taking $k = \frac{4f}{13}$.

Observe that $\frac{9f}{13} - 1 \equiv 2 \mod 3$, $\frac{12f}{13} - 1 = \frac{4(\frac{9f}{13} - 1) + 1}{3}$ and $\frac{6f}{13} = \frac{2(\frac{9f}{13} - 1) + 2}{3}$. If a second conflict occurs at $x' \le d$, by part (b) above we have

$$T_{x'}\left(\frac{12f}{13} - 1, \frac{6f}{13}, \frac{12f}{13} - 1, \frac{9f}{13} - 1\right) \le x' + T_{x'}\left(\frac{12f}{13} - 1, 0, \frac{12f}{13} - 1, \frac{6f}{13} - 1\right).$$

From this we may conclude

$$S_d\left(\frac{34f}{13} + 2, f\right) \le x + 2x + (x' - x) + x' + d - x' \le 4d,$$

which implies $\hat{S}(\beta) \le 4$ for $\beta \le 13/34$.

The results in items (2) and (3) follow analogously using Lemma 17 to reduce the density to cases (1) and (2), respectively. □

As was mentioned earlier, at least $n = 2f + 1$ robots are needed in order to solve conflicting claims of byzantine robots while searching. The bound given below is simple, but no better upper bound is yet known. Finding a better upper bound seems to be difficult.

Theorem 21. $S_d(2f + 1, f) \le 9d$.

Proof. All robots can execute the zig-zag algorithm for a single robot [5, 7, 12] as one group, until at least $f + 1$ robots claim to find the target in the same location. Since we have at most f faulty robots, the correct target is identified in that location and the search terminates in $9d$ as for a single non-faulty robot. □

As a corollary we have:

Corollary 22. $\hat{S}(\beta) \le 9$ for $\beta \le 1/2$.

3. Lower Bounds

Clearly if $n \ge 4f + 2$ then $2f + 1$ robots can be sent in each direction and the search is completed in time d as required. We now show that with any fewer robots time $2d$ is required.

Theorem 23. If $n \le 4f + 1$ then $S_d(n, f) \ge 2d$.

Proof. At time $d-\epsilon$ no one has visited d or $-d$. Consider where the robots are at this time. It must be the case that one of the intervals $(-d,0)$ or $(0,d)$ contains at most $2f$ robots. Without loss of generality, it is $(0,d)$. Put the target at d. Sort the robots by distance to d (ties broken arbitrarily) and make the f robots closest to d faulty and silent. Then at least one robot from $(-d,0]$ must also reach d so that $f+1$ non-faulty robots can identify the target. The search time is at least $d-\epsilon+d = 2d-\epsilon$. □

Observe that Theorem 23 shows that for $\beta \geq 1/4$ we have $\hat{S}(\beta) \geq 2$ and along with Proposition 8 we have that $S_d(5,1) = 2d$.

Finally we show that if the density of faulty is $\geq \frac{1}{3}$ the competitive ratio is at least 3.

Theorem 24. *If $n \leq 3f+1$ then $S_d(n,f) \geq 3d$.*

Proof. To prove the lower bound $S_d(3f+1,f) \geq 3d$ as follows. Consider visits to the set of symmetric positions $\{-d,+d\}$ by the robots. In particular, consider the first time that at least $f+1$ robots complete visits to one of the positions in the set. For instance, without loss of generality, assume that position $-d$ is visited first by at least $f+1$ robots and later (or simultaneously) position $+d$ is visited by at least $f+1$ robots. Clearly this time is at least d. The adversary arranges for a conflict at position $+d$. Note that unless the sets of robots visiting the two positions are disjoint, and hence the conflict at position $+d$ involves at most $2f$ robots participating in a vote, to resolve the conflict, at least one of the robots that visited $-d$ must move to $+d$. It follows that the total time required is at least $3d$. □

Observe that Theorem 24 shows that for $\beta \geq 1/3$ we have $\hat{S}(\beta) \geq 3$ and along with Proposition 7 we have that $S_d(4,1) = 3d$.

4. Discussion

In this paper we considered a generalization of the well-known cow-path problem by having the search done in parallel with a group of n robots, with up to f of them being byzantine faulty. We presented optimal search algorithms for several ratios n/f, and gave non-trivial upper and lower bounds in many cases. Several interesting problems in this setting remain open, the most interesting one being to give tight upper and lower bounds in the case $n = 2f+1$.

Acknowledgements

This work was partially supported by NSERC Grants and NSF Grant AF-1813940. The authors would like to thank the anonymous referees for many helpful suggestions.

References

[1] N. Agmon and D. Peleg, Fault-tolerant gathering algorithms for autonomous mobile robots, *SIAM J. Comput.* 36(1) (2006) 56–82.

[2] S. Albers and M. R. Henzinger, Exploring unknown environments, *SIAM J. Comput.* 29(4) (2000) 1164–1188.

[3] S. Albers, K. Kursawe and S. Schuierer, Exploring unknown environments with obstacles, *Algorithmica* 32(1) (2002) 123–143.

[4] S. Alpern and S. Gal, *The Theory of Search Games and Rendezvous*, Vol. 55 (Kluwer Academic Publishers, 2002).

[5] R. Baeza Yates, J. Culberson and G. Rawlins, Searching in the plane, *Inform. Comput.* 106(2) (1993) 234–252.

[6] E. Bampas, J. Czyzowicz, L. Gasieniec, D. Ilcinkas, R. Klasing, T. Kociumaka and D. Pajak, Linear search by a pair of distinct-speed robots, in *SIROCCO*, to appear, 2016.

[7] A. Beck, On the linear search problem, *Israel Journal of Mathematics* 2(4) (1964) 221–228.

[8] A. Beck, More on the linear search problem, *Israel Journal of Mathematics* 3(2) (1965) 61–70.

[9] A. Beck and M. Beck, Son of the linear search problem, *Israel Journal of Mathematics* 48(2-3) (1984) 109–122.

[10] A. Beck and D. Newman, Yet more on the linear search problem, *Israel Journal of Mathematics* 8(4) (1970) 419–429.

[11] A. Beck and P. Warren, The return of the linear search problem, *Israel Journal of Mathematics* 14(2) (1973) 169–183.

[12] R. Bellman, An optimal search, *SIAM Review* 5(3) (1963) 274–274.

[13] P. Bose and J.-L. De Carufel, A general framework for searching on a line, in *WALCOM: Algorithms and Computation — 10th International Workshop, WALCOM 2016, Kathmandu, Nepal, 29–31 March, Proceedings* (2016), pp. 143–153.

[14] P. Bose, J.-L. De Carufel and S. Durocher, Revisiting the problem of searching on a line, in *Algorithms — ESA 2013 — 21st Annual European Symposium, Sophia Antipolis, France, September 2–4, 2013. Proceedings* (2013), pp. 205–216.

[15] A. Casteigts, P. Flocchini, W. Quattrociocchi and N. Santoro, Time-varying graphs and dynamic networks, in *Ad-hoc, Mobile, and Wireless Networks*, LNCS, Vol. 6811 (Springer, 2011), pp. 346–359.

[16] Marek Chrobak, Leszek Gasieniec, Thomas Gorry and Russell Martin, Group search on the line, in *International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM)* (Springer, 2015), pp. 164–176.

[17] R. Cohen and D. Peleg, Convergence properties of the gravitational algorithm in asynchronous robot systems, *SIAM J. Comput.* 34(1) (2005) 1516–1528.

[18] J. Czyzowicz, L. Gasieniec, A. Kosowski, E. Kranakis, D. Krizanc and N. Taleb, When patrolmen become corrupted: Monitoring a graph using faulty mobile robots, in *Algorithms and Computation — Proceedings of 26th International Symposium, ISAAC 2015* (2015), pp. 343–354.

[19] Jurek Czyzowicz, Evangelos Kranakis, Danny Krizanc, Lata Narayanan and Jaroslav Opatrny, Search on a line with faulty robots, in *Proceedings of the ACM Symposium on Principles of Distributed Computing* (ACM, 2016), pp. 405–414.

[20] E. D. Demaine, S. P. Fekete and S. Gal, Online searching with turn cost, *Theor. Comput. Sci.* 361(2) (2006) 342–355.

[21] X. Deng, T. Kameda and C. Papadimitriou, How to learn an unknown environment, in *FOCS* (IEEE, 1991), pp. 298–303.

[22] Y. Dieudonné, A. Pelc and D. Peleg, Gathering despite mischief, *ACM Transactions on Algorithms (TALG)* 11(1) (2014) 1.
[23] F. V. Fomin and D. M. Thilikos, An annotated bibliography on guaranteed graph searching, *Theoretical Computer Science* 399(3) (2008) 236–245.
[24] F. Hoffmann, C. Icking, R. Klein and K. Kriegel, The polygon exploration problem, *SIAM J. Comput.* 31(2) (2001) 577–600.
[25] J. Hromkovič, R. Klasing, B. Monien and R. Peine, Dissemination of information in interconnection networks (broadcasting & gossiping), In *Combinatorial Network Theory* (Springer, 1996), pp. 125–212.
[26] J. Kleinberg, On-line search in a simple polygon, In *SODA* (SIAM, 1994), p. 8.
[27] F. Kuhn, N. Lynch and R. Oshman, Distributed computation in dynamic networks, In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing* (ACM, 2010), pp. 513–522.
[28] Andrey Kupavskii and Emo Welzl, Lower bounds for searching robots, some faulty, In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing* (ACM, 2018), pp. 447–453.
[29] L. Lamport, R. Shostak and M. Pease, The byzantine generals problem, *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4(3) (1982) 382–401.
[30] N. A. Lynch, *Distributed Algorithms* (Morgan Kaufmann, 1996).
[31] C. H. Papadimitriou and M. Yannakakis, Shortest paths without a map, In *ICALP* (Springer, 1989), pp. 610–620.
[32] S. Souissi, X. Défago and M. Yamashita, Gathering asynchronous mobile robots with inaccurate compasses, *Principles of Distributed Systems* (2006) 333–349.