Search and Rescue on the Line

Jared Coleman[1][0000−0003−1227−2962], Lorand Cheng[1], and Bhaskar Krishnamachari[1]

University of Southern California, Los Angeles, CA, USA
{jaredcol,lfcheng,bkrishna}@usc.edu

Abstract. We propose and study a problem inspired by a common task in disaster, military, and other emergency scenarios: search and rescue. Suppose an object (victim, message, target, etc.) is at some unknown location on a path. Given one or more mobile agents, also at initially arbitrary locations on the path, the goal is to find and deliver the object to a predefined destination in as little time as possible. We study the problem for the one- and two-agent cases and consider scenarios where the object and agents are arbitrarily (adversarially, even) placed along a path of either known (and finite) or unknown (and potentially infinite) length. We also consider scenarios where the destination is either at the endpoint or in the middle of the path. We provide both deterministic and randomized online algorithms for each of these scenarios and prove bounds on their (expected) competitive ratios.

Keywords: mobile, delivery, search, online, competitive ratio, search and rescue

1 Introduction

In this paper, we study a search and rescue problem where a set of autonomous agents on a one-dimensional path must cooperate to find and deliver an object to its destination (another location on the path) in as little time as possible. Formally, we consider a line with origin 0 onto which n agents with different speeds and an object which must be delivered to 0 are initially located arbitrarily (adversarially, even). We propose algorithms for scenarios where the object and agents are placed on the finite intervals $[0, 1]$ and $[-1, 1]$ but also discuss how slightly modified versions of the algorithms are equally competitive for the infinite intervals $[0, \infty)$ and $(-\infty, \infty)$. Agents can pick up, carry, and give the object to other agents (via physical handover) but can only communicate face-to-face. We assume agents can always move at their maximum speeds and that direction changes and handovers are instantaneous. In the offline setting, where the locations of all other agents and the object are known, this problem is equivalent to the Pony Express Communication Problem, for which an optimal offline algorithm is known [11]. In the online setting, this problem is related to search problems like the cow-path problem but differs in that we must consider the time required to deliver the object after it has been found. A strategy that considers the search and delivery components of the problem separately may not be optimal. For example, a search algorithm that minimizes the time to find the object might force agents into worst-case positions for the subsequent delivery.

Our goal is to find online algorithms with minimal *competitive ratio*. The competitive ratio of an online algorithm A is the maximum over all problem instances I of the ratio between the delivery time by A and the delivery time by an optimal offline algorithm for the same instance. Formally, the competitive ratio of A is

$$\sup_I \frac{T_{A,I}}{T_I^*}$$

where $T_{A,I}$ is the delivery time of algorithm A on instance I and T_I^* is the optimal offline delivery time for instance I. We say an algorithm with a competitive ratio c is c-competitive. For the problem studied in this paper, a c-competitive algorithm guarantees the object is delivered to the origin in at most $c \cdot T^*$ time, where T^* is the optimal (offline) delivery time had the location of the object been known to all agents from the start.

The results of the paper are summarized in the following table.

Table 1: Table of results: lower and upper bounds on the competitive ratios proven for each model studied where $W(x)$ is the product logarithm (Lambert W function [14]) of x and, for the two-agent scenarios, v is the relative speed of the slower agent with respect to the faster agent (i.e. if agents have speeds v_1 and v_2 such that $v_2 \leq v_1$, then $v = v_2/v_1$).

Agents	**Destination**	**Randomized**	**Lower Bound**	**Upper Bound**	**Section**
1	endpoint	no	$1+\sqrt{2}$	$1+\sqrt{2}$	4.1
		yes	5/3	2	4.1
	middle	no	5	5	4.2
		yes	5/3	$1+\frac{1}{2W(1/e)} \approx 2.79556$	4.2
2	endpoint	no	$1+\sqrt{2}$	$\min\left(1+\sqrt{2}, \frac{3-v}{1+v}\right)$	5
2 with radios	endpoint	no	$1+\sqrt{2}$	$\min\left(1+\sqrt{2}, \frac{3}{1+2v}\right)$	5.1

The layout of the paper is as follows. We survey related work in Section 2 and then present some preliminaries on the model and notation in Section 3. We begin our study in Section 4 by focusing on the problem with a single agent, considering scenarios with the destination at the endpoint (Section 4.1) and in the middle (Section 4.2), presenting deterministic and randomized algorithms for both scenarios. We present preliminary results for the multi-agent case by studying the problem for agents with no communication ability in Section 5 and then consider the case where agents can communicate (i.e. via radio) in Section 5.1. Finally, we conclude the paper with a summary of results and a discussion of areas for future work in Section 6.

2 Related Work

Cooperative mobile agents with communication constraints have been used to study search, exploration, rendezvous, message delivery, and other problems related to the search and rescue problem studied in this paper. Cow-path problems, first introduced in 1964 [2], are especially related to the search component of the problem we study. In its simplest form, the cow-path problem involves finding a target on a line with a single agent in as little time as possible. A simple 9-competitive algorithm has been shown to be optimal [2,21]. As a fundamental problem in search theory, many variants of the original cow-path problem have been proposed and solved for different models and using a variety of techniques. For multi-agent systems, it is sometimes framed as an evacuation problem, where the goal is to minimize the time for *every* agent to find and travel to an exit whose location is unknown [3,17]. The Group Search problem, on the other hand, requires any *one* agent to find the target [16]. These problems have been studied for many different topologies including the bounded line [4], the ring [24,29], the disk [15], simple polygons [26], for multiple paths (the original problem is a two-path system - left and right from the starting location of the agent) [25], the plane [20], in graphs [3], and in trees [19]. Competitive algorithms for multi-agent systems have been proposed [3,8,15,17,18,20,24,29], sometimes allowing some of the agents to be faulty [18,24]. A randomized algorithm has also been shown to dramatically improve the competitive ratio by a factor of almost 2 for the original problem (and to a lesser extent for the multi-path variant) [25]. Search for mobile targets has also been studied [6,13,21].

While cow-path problems relate directly with the search component of the problem studied in this paper, they do not consider the rescue component. The subsequent delivery that must occur after the object has been found fundamentally changes the problem. Recently, there has been work in data delivery by systems of mobile agents on the line [7,11], in the plane [10,12], and in graphs by energy-constrained agents [1,5]. The recently proposed Pony Express Communication Problem [11], where agents must cooperate to transmit an object from one endpoint of a line segment to the other, is most similar to the problem we study in this paper. In the offline setting, where the locations of the object and all agents are known to every agent ahead of time, our problem is equivalent to the Pony Express Communication Problem, for which an optimal offline algorithm exists. Essentially, the search and rescue problem we study here can be described as the Pony Express Communication Problem where the initial location of the object is unknown.

We are not aware of any existing work on this problem for the line, though it has been studied on the disk for the one- [23] and two-agent [22] cases. The problem considers agents which start at the center of a unit disk and the object and destination at unknown points on the perimeter of the disk. Algorithms for different communication models (wireless and face-to-face) have been presented and their worst-case delivery times proven. Both algorithms have a constant competitive ratio of $1 + \pi$, but rely on the assumption that the positive arc-distance between the exit and target is known ahead of time. In this paper, we make no assumptions about the distance of the object and also provide algorithms for both wireless and face-to-face communication models for the two-agent case.

Much of the existing work on search and rescue, exploration, and other cooperative tasks for multi-agent systems consider rather complex models and/or environments (obstacles, complex communication networks, communication dropouts, object recognition, urban or disaster environments, etc.). Techniques like queuing theory [9], machine learning [27], and heuristic-based [28] algorithms have been used to great effect. In this paper, we study the problem under a much simpler model in order to provide foundational theoretical guarantees with the hope that they can be used as a basis for future work.

3 Model and Notation

We consider agents that have a constant maximum speed and can start, stop, change directions, and pick up/hand over the object instantaneously. We assume agents can only move finite distances (they cannot move an infinitesimal distance in some direction). Agents may hand over the object to another agent only when they are collocated (face-to-face). In the offline setting, agents know the position of the object and the positions/speeds of all other agents at all times. In the online setting, however, agents do not know the position of the object or the positions/speeds of other agents. Except in Section 5.1, agents are assumed to have no ability to wireless communicate with each other. In all other sections, agents can only communicate with each other through face-to-face encounter. In both the face-to-face and wireless communication models, agents may share their entire state with each other instantaneously. We denote the initial position of the object by s and the (unknown) position of the object by y. For the single agent case we assume, without loss of generality, that the agent's speed is 1. For the two agent case, we use v_1 and v_2 to denote the speeds of the two agents such that $v_1 \geq v_2 > 0$. We use $v = v_2/v_1$ to denote the relative speed of the slower agent with respect to the faster agent.

4 A Single Agent

First, we study the problem for the case of a single agent. In this case, we can without any loss of generality assume the agent's speed to be 1. In other words, we simply define a unit of time to be the amount of time it takes for the agent to traverse the unit interval.

4.1 Destination at the Endpoint

In this section, we consider the interval $[0, 1]$ where the destination is at 0 and the agent's initial position is $s \geq 0$. Then, we discuss how the results extend to the unbounded interval $[0, \infty)$.

Deterministic Algorithms We start by showing a lower bound for any deterministic online single-agent algorithm.

Theorem 4.1. *Any online algorithm for the single-agent case has competitive ratio at least* $1+\sqrt{2}$.

Proof. Suppose the agent starts at position $\frac{1}{2}$. It is clear that any valid algorithm must eventually reach both endpoints 0 and 1 (otherwise there would exist instances of the problem, with the object at either 0 or 1, where the agent never finds the object). First, consider an algorithm which reaches 1 before 0. By adversarially placing the object at 0, the agent cannot deliver it before time $2\left(\frac{1}{2}\right)+\frac{1}{2}=\frac{3}{2}$ while an optimal algorithm would have delivered the object in time $\frac{1}{2}$, resulting in a competitive ratio of 3.

Now let's consider algorithms that reach endpoint 0 first. Let x be the largest visited point on the interval $\left[\frac{1}{2},1\right)$. In other words, the agent travels first to x and then to 0. As an adversary, we can choose to place the object either at $y\in(x,1]$ or at 0. If we choose the former, the delivery time of any algorithm is at least $\left(2x-\frac{1}{2}\right)+2y$ while the optimal delivery time is $\left(y-\frac{1}{2}\right)+y$. If we choose the latter, however, the delivery time of any algorithm is at least $\left(x-\frac{1}{2}\right)+x$ while the optimal delivery time is $\frac{1}{2}$. Since we have the power to choose whichever is worse for the algorithm, the competitive ratio can be written:

$$\max\left(\sup_{y>x}\left[\frac{2x-\frac{1}{2}+2y}{2y-\frac{1}{2}}\right],\frac{2x-1/2}{\frac{1}{2}}\right)=\max\left(\sup_{y>x}\left[1+\frac{2x}{2y-\frac{1}{2}}\right],4x-1\right) \quad (1)$$

$$=\max\left(\frac{8x-1}{4x-1},4x-1\right)\geq 1+\sqrt{2} \quad (2)$$

Observe $\sup_{y>x}\left[1+\frac{2x}{2y-\frac{1}{2}}\right]=\frac{8x-1}{4x-1}$ since $\frac{2x}{2y-\frac{1}{2}}$ is decreasing with respect to y. Then, the inequality above follows since $\frac{8x-1}{4x-1}$ is decreasing on $\left(\frac{1}{2},1\right)$, $4x-1$ is increasing on $\left(\frac{1}{2},1\right)$, and they intersect at $x=\frac{1}{2}+\frac{1}{2\sqrt{2}}$. Thus, for any algorithm (which determines a value for x), the competitive ratio is at least $1+\sqrt{2}$. $\square$

Now, we present Algorithm 1 and prove it to be optimal.

Algorithm 1 Online algorithm for agent starting at $s\in[0,1]$

```
1: $x \leftarrow \min\left(1, s\left(1+\frac{1}{\sqrt{2}}\right)\right)$
2: move along path $s \to x \to 0 \to 1$, returning to 0 with the object once it is found
```

Theorem 4.2. *Algorithm 1 has a competitive ratio of at most* $1+\sqrt{2}$.

Proof. Essentially, the algorithm involves traveling right toward 1 until reaching a point $x=\min\left(1,s\left(1+\frac{1}{\sqrt{2}}\right)\right)$, then traveling all the way to 0 (delivering the

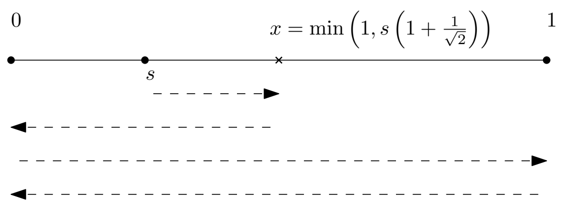
0
$x = \min\left(1, s\left(1+\frac{1}{\sqrt{2}}\right)\right)$
1
s

Fig. 1: The dashed line represents movement of agent executing Algorithm 1. The agent travels from its starting position at s to the point x, then to 0, then to 1 and back to 0 again. Once the agent encounters the object along this path it returns to 0 (not drawn).

object if it found it along the way). If the agent still does not have the object, it traverses the entire interval to the object (all the way to 1 if necessary) and back (Figure 1). Using a similar method as was used in proving Theorem 4.1, there are three interesting cases:

Case 1: $s = 0$. In this case, $x = 0$ and the algorithm clearly performs optimally, since it simply moves right until finding the object and then moves back to 0.

Case 2: $s \geq 2 - \sqrt{2}$. In this case, $x = 1$ and so placing the object at $y < s$ maximizes the competitive ratio (since any $y \geq s$ would result in an optimal delivery time):

$$\frac{(x-s)+x}{s} = \frac{2-s}{s} \leq 1+\sqrt{2}$$

Case 3: $0 < s < 2 - \sqrt{2}$. In this case, $s < x < 1$ and the maximum competitive ratio is achieved either by placing the object at some position $y < s$ or at some position $y > x$ (since any $s \leq y \leq x$ would result in an optimal delivery time):

$$\begin{aligned} \max\left(\sup_{y>x}\left[\frac{(2x-s)+2y}{(y-s)+y}\right], \frac{(x-s)+x}{s}\right) &= \max\left(\frac{4x-s}{2x-s}, \frac{2x-s}{s}\right) \\ &= \max\left(1+\sqrt{2}, 1+\sqrt{2}\right) \qquad (3) \\ &= 1+\sqrt{2} \end{aligned}$$

where Equation (3) follows since $x = s\left(1+\frac{1}{\sqrt{2}}\right)$ in this case. Thus, Algorithm 1 has a competitive ratio of at most $1+\sqrt{2}$. □

This result is particularly interesting when compared to the search and delivery problems separately. First, observe that for the search problem with no lower bound on the distance of the target to the agent's starting location, there is no competitive online algorithm! The agent must move some distance either left or right to begin with - whatever that distance is, we can place the object an arbitrarily small fraction of the distance in the other direction, making the competitive ratio of the algorithm arbitrarily large. The delivery problem on the line segment, on the other hand, is trivial with one agent - just go to the object and then the destination. A competitive online algorithm for the search and rescue problem, however, *does* exist and is *not* trivial!

It's important to understand that Algorithm 1 does *not* minimize the worst-case delivery time of the object. In fact, a simple algorithm of just going to 1 and then back to 0 terminates in at most time 2 (when the object is at 1) while Algorithm 1 can take up to $2+\sqrt{2}$ time (when $s = 2-\sqrt{2}-\epsilon$ for some arbitrarily small $\epsilon > 0$ and the object is at 1). Rather, Algorithm 1 minimizes the delivery time compared to the optimal delivery time if the location of the object were known. The aforementioned simple algorithm, on the other hand, might take time 2 to deliver an object that could have been delivered almost instantaneously! Algorithm 1 guarantees this never happens — an object that can be delivered in time t optimally will be delivered in at most $(1+\sqrt{2})t$ time. So Algorithm 1 (and any other which minimizes competitive ratio in general) might be described as an algorithm that minimizes the regret that an agent has after discovering the location of the object.

There is another even more important scenario where an algorithm's competitive ratio is more useful than its worst-case runtime: when the worst-case runtime is unbounded. Consider the situation where an agent no longer knows the length of the path ahead of it — only its initial distance to 0. In the extreme case, the object could be anywhere on the interval $[0,\infty)$. In this case, there is no simple exhaustive search algorithm that terminates in bounded time because the path could go on forever (or in a more realistic scenario, for a really, really long time)! Even for this extreme case, Algorithm 2 (a slight modification of Algorithm 1 which simply removes the upper bound on the agent's search to the right of s), will still deliver the object to its destination in $1+\sqrt{2}$ times the optimal time.

Algorithm 2 Online algorithm for agent starting at $s \in [0,\infty)$

```
1: $x \leftarrow s\left(1+\frac{1}{\sqrt{2}}\right)$
2: move along path $s \to x \to 0 \to \infty$, returning to 0 with the object once it is found
```

Corollary 1. *Algorithm 2 has a competitive ratio of* $1+\sqrt{2}$.

Proof. Follows directly from the proof for Algorithm 1. □

Using Randomization The analysis for Algorithm 1 involved reasoning about how an adversary might place the object at worst-case positions along the line-segment. By using randomization, we can mitigate the damage an adversary can do by not committing to a predictable, deterministic algorithm. First, we prove a lower bound on how well a randomized algorithm can do:

Theorem 4.3. *Every randomized online algorithm for the single-agent case has an expected competitive ratio of at least* 5/3.

Proof. Consider the scenario where the agent is at some position $s < \frac{1}{2}$ and the object is placed on the interval $(s, 2s]$ uniformly at random with probability 2/3 and at 0 with probability 1/3. Observe in the former case, the expected position of the object is $(2s + s)/2 = 3s/2$. Since the object cannot be in the interval $(0, s]$, any optimal online algorithm must involve the agent *either* moving along the path $s \to 0 \to 1$ *or* along the path $s \to x \to 0 \to 1$ (returning to 0 as soon as the object is found, of course) for some $x \in (s, 1]$. If the agent moves to 0 first, then the expected competitive ratio is

$$\frac{1}{3} \cdot 1 + \frac{2}{3} \cdot \left(\frac{2(3s)/2 + s}{2(3s)/2 - s}\right) = \frac{1}{3} + \frac{2}{3}\left(\frac{3s + s}{3s - s}\right) = 5/3$$

If instead, the agent moves to some position $x \in (s, 2s]$, then the probability that the agent finds the object (in optimal time) on $(s, x]$ is $\frac{2}{3} \cdot \frac{x-s}{s}$. On the other hand, the probability that the object is in the interval $(x, 2s]$ is $\frac{2}{3} \cdot \frac{2s-x}{s}$. Given this situation, observe that the expected position of the object in this case is $(2s + x)/2$. Thus the competitive ratio can be written:

$$\frac{1}{3} \cdot \frac{2x - s}{s} + \frac{2}{3}\left(\frac{x - s}{s} \cdot 1 + \frac{2s - x}{s} \cdot \frac{2x - s + 2(x + 2s)/2}{2(x + 2s)/2 - s}\right) = \frac{s^2 + 11sx - 2x^2}{3s^2 + 3sx}$$

which has a maximum value of 5/3 (at both $x = s$ and $x = 2s$). Thus, any deterministic algorithm for this distribution of inputs has an expected competitive ratio of at least 5/3. Finally, by Yao's Minimax Principle [30], every randomized algorithm must have an expected competitive ratio of at least 5/3. □

Now, we present a simple randomized algorithm: with probability $\frac{1}{2}$, the agent will simply execute an algorithm very similar to Algorithm 1, otherwise the agent will move directly to 0 and then, if it still hasn't found the object, move towards 1 until it does and then return to 0.

Algorithm 3 Online randomized algorithm for agent starting at $s \in [0, 1]$

```
1: Let $p$ be a random bit
2: if $p = 0$ then
3:     move along path $s \to \min(2s, 1) \to 0 \to 1$, returning to 0 with the object once
       it is found
4: else
5:     move along path $s \to 0 \to 1$, returning to 0 with the object once it is found
```

Theorem 4.4. *Algorithm 3 has an expected competitive ratio of* 2.

Proof. First, we consider the case where $s < 1/2$. Observe the object is either in the interval $(0, s)$, $(s, 2s]$, or $(2s, 1]$. The goal of the adversary now is to place the object in the interval which maximizes the expected competitive ratio. For example, if the object is in the first interval, then the algorithm is optimal with probability 1/2 (the agent goes towards 0). Otherwise, it has a competitive ratio of $\frac{2(2s)-s}{s} = 3$. The adversary is not required to commit to a deterministic strategy, however. Consider a mixed strategy where the adversary places the object in $(0, s)$ with a probability of q, in $(s, 2s]$ with a probability of r, and in $(2s, 1]$ with a probability of $1-q-r$. Let CR denote the competitive ratio of Algorithm 3. Then the expected competitive ratio can be written:

$$\begin{aligned}
\mathbb{E}[CR] = \; & \frac{1}{2}\left(q \cdot 1 + r \cdot \frac{2y_2+s}{2y_2-s} + (1-q-r)\frac{2y_3+s}{2y_3-s}\right) + \\
& \frac{1}{2}\left(q \cdot \frac{2(2s-s)+s}{s} + r \cdot 1 + (1-q-r)\frac{2y_3+3s}{2y_3-s}\right) \\
\mathbb{E}[CR] \le \; & \frac{1}{2}\left(q \cdot 1 + r \cdot 3 + (1-q-r)\frac{5}{3}\right) + \frac{1}{2}\left(q \cdot 3 + r \cdot 1 + (1-q-r)\frac{7}{3}\right) = 2
\end{aligned}$$

where y_2 and y_3 are the expected positions of the object in cases 2 and 3, respectively. The above inequality follows from worst-case values (those which maximize the competitive ratio) y_2 approaches s (from above) and y_3 approaches $2s$ (from above).

Now we must consider the case where $s \ge 1/2$. In this case, there are only two intervals in which the adversary may place the object. Let q' be the probability the object is in $[0, s)$ (case 1) and $1-q'$ the probability it is in $(s, 1]$ (case 2). Then the competitive ratio can be written:

$$\begin{aligned}
\mathbb{E}[CR] = \; & \frac{1}{2}\left(q' \cdot 1 + (1-q') \cdot \frac{2y_2'+s}{2y_2'-s}\right) + \frac{1}{2}\left(q' \cdot \frac{2-s}{s} + (1-q') \cdot 1\right) \\
\mathbb{E}[CR] \le \; & \frac{1}{2}\left(q' \cdot 1 + (1-q') \cdot 3\right) + \frac{1}{2}\left(q' \cdot 3 + (1-q') \cdot 1\right) = 2
\end{aligned}$$

where y_2' is the expected position of object in case 2. The above inequality follows from the worst-case value (that which maximizes the competitive ratio) where y_2' approaches s (from above). □

The expected competitive ratio of Algorithm 3 is significantly lower than the competitive ratio of the deterministic Algorithm 1, especially with respect to the lower bound on the competitive ratio of any randomized algorithm.

Remark 1. A slight modification of Algorithm 3 for the unbounded interval $[0, \infty)$ (simply replace 1 in the paths with ∞) has the same expected competitive ratio since the second scenario discussed in the proof for Theorem 4.4 is essentially eliminated and the analysis for the first scenario is equivalent.

4.2 Destination in the Middle

Up until this point, we've only considered scenarios where the destination is at an endpoint. In this section, we consider situations where the object may be on *either* side of the destination.

Deterministic Algorithms We start by showing the necessity of an additional assumption for this variant of the problem: the agent's initial position cannot be at 0 (the destination).

Lemma 4.1. *For any initial configuration where the agent starts at* 0 *and the object is on one of two paths emanating from* 0*, there is no competitive algorithm.*

Proof. Any algorithm must involve the agent doing one of two things at time $t = 0$:

1. wait for some time t'
2. move some distance $d' > 0$ down one of the paths

In the first case, an adversary may simply place the object along any of the paths an arbitrarily small distance $d/2$ away from 0 (so the optimal delivery time is d). The competitive ratio in this case is *at least* t'/d, which is arbitrarily large as $d \to 0$. In the second case, an adversary may place the object an arbitrarily small distance $d/2$ away from 0 along any path *except* the one the agent traveled down. Again, the competitive ratio is at least d'/d, which is arbitrarily large as $d \to 0$ (since $d' > 0$). □

Thus, we assume, without loss of generality that the agent starts at some position $s > 0$ (all proofs follow for $s < 0$ via a symmetrical argument). We present an online "zig-zag" algorithm (Algorithm 4) which involves the agent searching a distance $2s$ to the left (crossing 0) and returning to s, then a distance $4s$ to the right and returning to s, then a distance $8s$ to left and returning to s, and so on, doubling its search distance in each round (Figure 2). In other words, the agent follows the trajectory $s \to -s \to 5s \to -7s \to 15s \to \ldots$.

Algorithm 4 Online Algorithm for agent starting at $s \neq 0$

```
1: $i \leftarrow 1$
2: $x \leftarrow -s$
3: while object not found do
4:     move toward $x$
5:     if arrived at $x \neq s$ or an endpoint then
6:         $x \leftarrow s$
7:     else if arrived at $s$ then
8:         $i \leftarrow i + 1$
9:         $x \leftarrow s + (-1)^i \cdot 2^i s$
10: Return to 0 with object
```

Now, we will show an upper bound of 5 on its competitive ratio, then prove it is optimal.

Theorem 4.5. *Algorithm 4 has a competitive ratio of* 5*.*

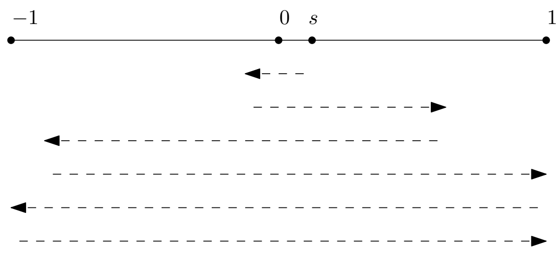

Fig. 2: The dashed line represents movement of agent executing Algorithm 4. The agent travels from its starting position at s to the point $-s$, then to $5s$, and so on. Upon finding the object, the agent returns to 0 (not drawn).

Proof. In Algorithm 4, the agent starts at s and moves left and right in alternating rounds, doubling the distance it travels each round. In round $i = 1, 2, \ldots,$ the agent moves a distance $2^i s$ (left in odd rounds and right in even rounds) out and back to s (for a total of 2^{i+1}). In the case an endpoint is reached, the agent would turn around rather than finish travelling the full distance of the round. However, to simplify the analysis it is easier to consider the alternate algorithm A' such that the agent does not turn around early and travels a distance 2^i (out and back) no matter what, traveling beyond the endpoint if necessary. It is clear that our original algorithm cannot perform worse than A' and in cases where an endpoint is never reached before finding the object, the two algorithms are identical. Thus, any upper bound on the competitive ratio of A' is also an upper bound on Algorithm 4. For the following analysis, assume the algorithm we are referring to is A'.

Without loss of generality, suppose $s > 0$ (a symmetric argument follows when $s < 0$). Let y be the position of the object. Then there are three interesting cases: when $-s \leq y \leq s$, when $y > s$, and when $y < -s$. The first case is the simplest to analyze. Since $-s \leq y < s$, the object is found in round 1 and is clearly optimal. For the second case, if the object is found in round 2, then $s < y < 5s$ and the competitive ratio is

$$\frac{3s + 2y}{2y - s} \leq 5 \tag{4}$$

since the left-hand side of Inequality (4) is decreasing with respect to y and $y > s$. Otherwise, $y > 5s$ and so the object must be found in some even round $k > 2$. Also, observe $2^{k-2}s < y - s$ (otherwise the object would have been found in an earlier round), implying $2^k < \frac{4(y-s)}{s}$. The delivery time of A', then, is

$$T_{A'} = \sum_{i=1}^{k-1} 2^{i+1}s + 2y - s = 2s\left(2^k - 2\right) + 2y - s$$

$$T_{A'} < 2s\left(\frac{4(y-s)}{s} - 2\right) + 2y - s = 10y - 13s,$$

while the optimal delivery time is $2y - s$, thus the competitive ratio is at most $(10y - 13s)/(2y - s) \leq 5$. Finally, when $y < -s$, the object must be found in some odd round $k > 2$. Then, we have $2^{k-2}s < s + |y|$ (otherwise the object would have been found in an earlier round just as in the first case), implying $2^k < \frac{4(s+|y|)}{s}$. The delivery time of A' in this case is

$$\begin{aligned}\sum_{i=1}^{k-1} 2^i s + 2|y| + s &= 2s(2^k - 2) + 2|y| + s \\ &< 2s\left(\frac{4(s+|y|)}{s} - 2\right) + 2|y| + s = 10|y| + 5s,\end{aligned}$$

while the optimal delivery time is $2|y| + s$. Thus the competitive ratio is at most $(10|y| + 5s)/(2|y| + s) \leq 5$. □

Remark 2. Note that algorithm A' is exactly the algorithm for the unbounded case, so the competitive ratio of at most 5 applies to both the bounded and unbounded cases.

Theorem 4.6. *Every online algorithm for the single-agent, line model must have a competitive ratio of at least* 5.

Proof. Consider a scenario where the agent is placed at some arbitrarily small distance $s > 0$ away from 0 and the object is at least a distance $2s$ from s. Any algorithm must involve a sequence of positive distances $x_1, x_2, x_3, \ldots$ such that the agent moves left (or right) a distance x_1 and back to s, then right (or left) a distance x_2 and back to s, and so on. Clearly any optimal online algorithm of this form must satisfy $x_{i+2} > x_i$ and any algorithm with a competitive ratio of 5 or better must satisfy $x_i \leq 3x_{i-1}$ where $x_1 \geq 2s$ (since the object at least a distance $2s$ from s). Thus, $x_i < 2s \cdot 3^{i-1}$ for all $i \geq 2$ and so the number of required turning points for an optimal online algorithm can be made arbitrarily large (by setting s to be arbitrarily small).

For some sequence of turning points, suppose the object is found on the k^{th} round (k can be arbitrarily large by the argument above) and observe the competitive ratio can then be written

$$\begin{aligned}\sup_{k,y,s} \frac{\sum_{i=1}^{k-1} 2x_i + 2y \pm s}{2y \pm s} &= \sup_k \frac{\sum_{i=1}^{k-1} x_i + x_{k-2}}{x_{k-2}} \\ = 1 + \sup_k \frac{\sum_{i=1}^{k-1} x_i}{x_{k-2}} &\geq 1 + \sup_k \frac{\sum_{i=1}^{k-1} r^i}{r^{k-2}} = 1 + \sup_k \frac{r^k - r}{(r-1)r^{k-2}}\end{aligned}$$

where $r > 1$ (the expansion factor). The above inequality follows from Corollary 7.11 of Section 7.2 of [21] (following the method for proving the 9-competitive search on an infinite line in Section 8.2.1 of [21]). Then, since $\frac{r^k - r}{(r-1)r^{k-2}}$ is increasing with respect to k (its derivative $\frac{r^{3-k}\ln r}{r-1}$ is greater than 0 for any $r > 1$), we can simplify the competitive ratio to

$$1 + \sup_k \frac{r^k - r}{(r-1)r^{k-2}} = 1 + \lim_{k\to\infty} \frac{r^k - r}{(r-1)r^{k-2}} = 1 + \frac{r^2}{r-1}$$

which has a minimum value of 5 for $r = 2$. □

Using Randomization Again, the analysis of Algorithm 4 involved reasoning about worst-case positions of the object. The following algorithm and subsequent upper bound are very similar to the well-known optimal randomized algorithm for the cow-path search problem [25]. The algorithm is essentially a standard zig-zag algorithm (like Algorithm 4) except that the starting search direction and initial search distance are randomized. In the following Algorithm 5, the random bit p determines the initial search direction and the random number $\epsilon \in (0,1)$ determines the initial search distance (which is $r^\epsilon s$ for some constant $r > 1$).

Algorithm 5 Online randomized algorithm for agent starting at $s \neq 0$ with expansion rate $r > 1$

```
1: let $p$ be a random bit
2: sample $\epsilon$ uniformly at random from the interval $(0,1)$
3: $i \leftarrow 1$
4: $x \leftarrow s + (-1)^{i-p} \cdot r^{i+\epsilon} s$
5: **while** object not found **do**
6:     move toward $x$
7:     **if** arrived at $x \neq s$ or an endpoint **then**
8:         $x \leftarrow s$
9:     **else if** arrived at $s$ **then**
10:        $i \leftarrow i + 1$
11:        $x \leftarrow s + (-1)^{i-p} \cdot r^{i+\epsilon} s$
12: Return to 0 with object
```

Theorem 4.7. *The expansion rate* $r = \frac{1}{W(1/e)} \approx 3.59112$ *yields an expected competitive ratio of* $1 + \frac{1}{2W(1/e)} \approx 2.79556$ *for Algorithm 5 where* $W(x)$ *is the product logarithm (Lambert W function [14]) of* x.

Proof. Just as in the proof for Theorem 4.5, we consider the alternate algorithm A' such that the agent does not turn around early upon reaching an endpoint. It is clear that our original algorithm cannot perform worse than A' and in cases where an endpoint is never reached, the two algorithms are identical.

Let $d = s \cdot r^{k+\delta}$ denote the position of the object where $0 \leq \delta < 1$. By executing Algorithm 5, the agent moves a distance $r^{i+\epsilon}$ to the left and right in alternating rounds $i = 1, 2, \ldots$. Consider the round when the agent moves a distance $s \cdot r^k$ for the first time. If the agent moves r^k distance for the first time in the opposite direction of the object, then it will definitely find the object in round $k+1$. The expected competitive ratio in this case can be written:

$$\begin{aligned}
\mathbb{E}\left[\frac{\sum_{i=1}^{k} s \cdot 2r^{i+\epsilon} + 2d \pm s}{2d \pm s}\right] &= \mathbb{E}\left[\frac{\sum_{i=1}^{k} 2r^{i+\epsilon} + 2r^{k+\delta} \pm 1}{2r^{k+\delta} \pm 1}\right] \\
&= \mathbb{E}\left[1 + \frac{2r^{\epsilon}\left(r^{k+1} - r\right)}{\left(2r^{k+\delta} \pm 1\right)(r-1)}\right] \\
&= 1 + \frac{2\left(r^{k+1} - r\right)}{\left(2r^{k+\delta} \pm 1\right)(r-1)} \cdot \mathbb{E}\left[r^{\epsilon}\right] \\
&= 1 + \frac{2\left(r^{k+1} - r\right)}{\left(2r^{k+\delta} \pm 1\right)\ln r}
\end{aligned}$$

since $\mathbb{E}[r^{\epsilon}] = \int_1^r x \frac{1}{x \ln r} dx = \frac{r-1}{\ln r}$.

On the other hand, if the agent moves a distance $s \cdot r^k$ for the first time in the direction of the object, it will find it on round k if $\epsilon \geq \delta$ and on round $k+2$ otherwise. Let B be the event that $\epsilon \geq \delta$, then the expected competitive ratio can be written:

$$\begin{aligned}
&\mathbb{E}\left[Pr[B]\left[\frac{\sum_{i=1}^{k-1} s \cdot 2r^{i+\epsilon} + 2d \pm s}{2d \pm s}\right] + (1 - Pr[B])\left[\frac{\sum_{i=1}^{k+1} s \cdot 2r^{i+\epsilon} + 2d \pm s}{2d \pm s}\right]\right] \\
&= \mathbb{E}\left[Pr[B]\left[1 + \frac{2r^{\epsilon}(r^k - r)}{\left(2r^{k+\delta} \pm 1\right)(r-1)}\right] + (1 - Pr[B])\left[1 + \frac{2r^{\epsilon}(r^{k+2} - r)}{\left(2r^{k+\delta} \pm 1\right)(r-1)}\right]\right] \\
&= Pr[B]\left[1 + \frac{2(r^k - r)}{\left(2r^{k+\delta} \pm 1\right)(r-1)} \cdot \mathbb{E}\left[r^{\epsilon}|B\right]\right] \\
&\quad + (1 - Pr[B])\left[1 + \frac{2(r^{k+2} - r)}{\left(2r^{k+\delta} \pm 1\right)(r-1)} \cdot \mathbb{E}\left[r^{\epsilon}|\overline{B}\right]\right] \\
&= Pr[B]\left[1 + \frac{2(r^k - r)(r - r^{\delta})}{\left(2r^{k+\delta} \pm 1\right)(r-1)\ln r Pr[B]}\right] \\
&\quad + (1 - Pr[B])\left[1 + \frac{2(r^{k+2} - r)(r^{\delta} - 1)}{\left(2r^{k+\delta} \pm 1\right)(r-1)\ln r Pr[\overline{B}]}\right]
\end{aligned}$$

since $\mathbb{E}[r^\epsilon|B] = \int_{r^\delta}^{r} x\frac{1}{Pr[B]\cdot x\ln r}dx = \frac{r-r^\delta}{\ln r Pr[B]}$ and $\mathbb{E}[r^\epsilon|\overline{B}] = \int_{1}^{r^\delta} x\frac{1}{Pr[\overline{B}]\cdot x\ln r}dx = \frac{r^\delta-1}{\ln r Pr[\overline{B}]}$. Then the expression can be further simplified:

$$\begin{aligned}
&= 1 + \frac{2(r^k-r)(r-r^\delta)}{(2r^{k+\delta}\pm 1)\,(r-1)\ln r} + \frac{2(r^{k+2}-r)(r^\delta-1)}{(2r^{k+\delta}\pm 1)\,(r-1)\ln r}\\
&= 1 + \frac{2}{(2r^{k+\delta}\pm 1)\,(r-1)\ln r}\left((r^k-r)(r-r^\delta)+(r^{k+2}-r)(r^\delta-1)\right)\\
&= 1 + \frac{2}{(2r^{k+\delta}\pm 1)\,(r-1)\ln r}(r-1)(r^{\delta+k}+r^{d+k+1}-r^{k+1}-r)\\
&= 1 + \frac{2(r^{\delta+k}+r^{d+k+1}-r^{k+1}-r)}{(2r^{k+\delta}\pm 1)\ln r}
\end{aligned}$$

Observe that, since the initial search direction is chosen uniformly randomly, the total expected competitive ratio is

$$\begin{aligned}
&\frac{1}{2}\left[1+\frac{2\left(r^{k+1}-r\right)}{(2r^{k+\delta}\pm 1)\ln r}\right]+\frac{1}{2}\left[1+\frac{2(r^{\delta+k}+r^{d+k+1}-r^{k+1}-r)}{(2r^{k+\delta}\pm 1)\ln r}\right]\\
&= 1+\frac{\left(r^{k+1}-r\right)}{(2r^{k+\delta}\pm 1)\ln r}+\frac{(r^{\delta+k}+r^{d+k+1}-r^{k+1}-r)}{(2r^{k+\delta}\pm 1)\ln r}\\
&= 1+\frac{r^{k+\delta}(1+r)-2r}{(2r^{k+\delta}\pm 1)\ln r}\le 1+\frac{r^{k+\delta}(1+r)-(1+r)}{(2r^{k+\delta}-1)\ln r-\ln r}=1+\frac{(1+r)(r^{k+\delta}-1)}{\ln r\,(2r^{k+\delta}-2)}\\
&= 1+\frac{1+r}{2\ln r}.
\end{aligned}$$

Finally, with an expansion rate of $r = \frac{1}{W(1/e)} \approx 3.59112$, the above upper bound becomes $1+\frac{1}{2W(1/e)} \approx 2.79556$. $\square$

Remark 3. Note that, again, algorithm A' is exactly the modified version of A that would be used in the unbounded case, so the competitive ratio of at most $1+\frac{1}{2W(1/e)}$ applies to both the bounded and unbounded cases.

5 Two Agents

In this section, we present results for multi-agent search and rescue, considering the case of two agents initially located at the same point s on the line segment but with different speeds v_1 and v_2. Without loss of generality, we assume $v_1 \geq v_2$. We refer to the agent with speed v_1 as the "first" or "fast" agent and the agent with speed v_2 as the "second" or "slow" agent. The goal is the same, except agents may hand over the object to each other via face-to-face encounter. We denote $v = v_2/v_1$ as the speed of the slower agent relative to the faster agent (observe $0 \leq v \leq 1$). The delivery time of the optimal algorithm, then, is clearly $(2y-s)/v_1$, where the fast agent delivers the object entirely by itself.

Remark 4. By Theorem 4.1, whenever $v_2 = 0$, the lower bound of $1+\sqrt{2}$ applies to the two-agent case directly.

Now we present Algorithm 6, an online algorithm which involves the slow agent moving toward 1 and the fast agent moving toward 0 only if doing so is better than the fast agent simply executing Algorithm 1 by itself.

Algorithm 6 Online two-agent algorithm for agents starting at $s \in [0,1]$

```
1: if other agent is faster then
2:     $x \leftarrow 1$
3: else if $\frac{2-\sqrt{2}}{2+\sqrt{2}} < v < 1$ then
4:     $x \leftarrow 0$
5: else
6:     $x \leftarrow s(1+1/\sqrt{2})$
7: move along path $s \to x \to 0 \to 1$, returning to 0 with the object once it is found
   and handing it over to any faster agent encountered
```

Theorem 5.1. *For any system with two agents starting at position* $s \in [0,1]$*, Algorithm 6 has a competitive ratio of* $\min\left(1+\sqrt{2}, \frac{3-v}{1+v}\right)$ *where* $v = v_2/v_1$.

Proof. First, observe that since both agents start at s, an optimal offline algorithm involves only the first agent moving directly toward the object and then to 0 for delivery. Thus, in the case where $v \le \frac{2-\sqrt{2}}{2+\sqrt{2}}$ or $v = 1$, the first agent exactly performs Algorithm 1, and so a competitive ratio of $1+\sqrt{2}$ is achieved. The interesting case, then is when $\frac{2-\sqrt{2}}{2+\sqrt{2}} < v < 1$. Let y be the position of the object along the line segment. Clearly if $y \le s$, then the algorithm is optimal. If $y > s$, then there are two possible scenarios. If $s + y \le (y-s)/v$, then the fast agent reaches the object at the same time or before the slow agent, so the competitive ratio is

$$\frac{2y+s}{2y-s} \le \frac{2y + \frac{1-v}{1+v}y}{2y - \frac{1-v}{1+v}y} = \frac{3+v}{3v+1}.$$

If $s + y > (y-s)/v$, then the slow agent reaches the object first, picks it up, and moves toward 0 until encountering the fast agent for a handover. Then the fast agent delivers the object. In this case, the delivery time can be written as the sum of the time it took to meet ($t = \frac{2y}{1+v}$) and the time for the fast agent to carry the object the remaining distance ($y - (tv - (y-s)) = 2y - tv - s$) for a total time of $\frac{4y}{1+v} - s$. Thus, the competitive ratio is

$$\frac{\frac{4y}{1+v} - s}{2y - s} \le \frac{3-v}{1+v}$$

since the function is decreasing with respect to y in the $y > s$ region and thus obtains its maximum value at $y = s$. Finally, observe the second case dominates the competitive ratio:

$$\frac{3-v}{1+v} \geq \frac{3+v}{1+3v} \;\Rightarrow\; 3+8v-3v^2 \geq 3+4v+v^2 \;\Rightarrow\; v(1-v) \geq 0$$

Of course this condition is always satisfied since $0 \leq v \leq 1$. □

Observe that since agents start at the same position, Algorithm 1 still has a competitive ratio of $1+\sqrt{2}$. Algorithm 6 is only better when $v_2 > \frac{2-\sqrt{2}}{2+\sqrt{2}} v_1 \approx 0.1716 v_1$. In other words, as long as one agent is not *too* much faster than the other, Algorithm 6 is more competitive.

5.1 Agents with Radios

In Algorithm 6, the fast agent moves toward 0 and only turns around to help the slower agent if it reaches 0 without finding the object. If the slow agent finds the object before the fast agent reaches 0 and the agents can communicate, though, then clearly the fast agent should turn around immediately to acquire the object as quickly as possible.

Algorithm 7 Online two-agent algorithm for agent starting at $s \in [0,1]$

```
1: if other agent is faster then
2:     $x \leftarrow 1$
3: else if $\frac{2-\sqrt{2}}{2+2\sqrt{2}} < v < 1$ then
4:     $x \leftarrow 0$
5: else
6:     $x \leftarrow s(1+s/\sqrt{2})$
7: move along path $s \to x \to 0 \to 1$, returning to 0 with the object once it is found
   and handing it over to any fast agent encountered
```

Theorem 5.2. *Algorithm 7 has a competitive ratio of* $\min\left(1+\sqrt{2}, \frac{3}{1+2v}\right)$.

Proof. The proof is very similar to that of Theorem 5.1. Without loss of generality, suppose the first agent has a speed of 1 and the second agent a speed of $v \leq 1$. For the case where $v \leq \frac{2-\sqrt{2}}{2+2\sqrt{2}}$ or $v = 1$, the first agent exactly performs Algorithm 1, and so a competitive ratio of $1+\sqrt{2}$ is achieved. The interesting case, then is when $\frac{2-\sqrt{2}}{2+2\sqrt{2}} < v < 1$. First, if the fast agent still arrives at the object first, the competitive ratio is $\frac{3+v}{1+3v}$ (using the same analysis as used in the proof for Theorem 5.1). Otherwise, if the slow agent arrives at the object before the fast agent reaches 0 (i.e. $s > \frac{y-s}{v} \Rightarrow y < s(v+1)$), then the competitive ratio is

$$\frac{t + (y - (tv - (y - s)))}{2y - s} = \frac{s(2-v) - 2y}{v(s - 2y)} \leq \frac{3}{1 + 2v}$$

where $t = \frac{2\left(y - \left(s - \frac{y-s}{v}\right)\right)}{1+v}$ is the time the agents meet for a handover. The first equality follows from substituting this value for t and the final inequality follows since $\frac{s(2-v)-2y}{v(s-2y)}$ is increasing with respect to y (its derivative with respect to y, $\frac{2s(1-v)}{v(s-2y)^2}$, is positive for all $v < 1$) and $y < s(v+1)$. On the other hand, if the slow agent finds the object after the fast agent reaches 0 but still before it catches up ($s < \frac{y-s}{v} < y + s$), then the competitive ratio is

$$\frac{t + (y - (tv - (y - s)))}{2y - s} = \frac{s(1+v) - 4y}{(1+v)(s-2y)} = \frac{s(1+v) - 4y}{s(1+v) - 2y(1+v)} \leq \frac{3}{1+2v}$$

where $t = \frac{2y}{1+v}$ is the time the agents meet for a handover. The first equality follows from substituting this value for t and the final inequality follows since $\frac{s(1+v)-4y}{s(1+v)-2y(1+v)}$ is decreasing with respect to y (its derivative with respect to y, $\frac{2s(v-1)}{(1+v)(s-2y)^2}$, is negative for all $v < 1$) and $y > s(1+v)$. Finally, observe the second and third cases dominate the competitive ratio:

$$\frac{3}{1+2v} \geq \frac{3+v}{1+3v} \;\Rightarrow\; 3 + 9v \geq 3 + 7v + 2v^2 \;\Rightarrow\; v(1-v) \geq 0$$

Clearly this condition is always satisfied since $0 \leq v \leq 1$. $\square$

Observe that Algorithm 7 has a better competitive ratio than Algorithm 1 whenever the slow agent has speed greater than $\frac{2-\sqrt{2}}{2+2\sqrt{2}} \approx 0.1213$. Recall for the case where agents cannot communicate, the slow agent only helps when its speed is greater than $\frac{2+\sqrt{2}}{2-\sqrt{2}} \approx 0.1716$.

6 Conclusion

In this paper, we propose a problem inspired by search and rescue, a task that cooperative robotic systems have long showed promise in assisting with. We provide both deterministic and randomized algorithms for the single-agent case and provide lower and upper bounds on their competitive ratios. We showed the search and rescue problem is fundamentally different than the search and delivery problems considered separately, which are trivial. For the case where the destination is in the middle, however, the optimal (deterministic and randomized) algorithms are essentially the same as the well-known optimal algorithms for search problem, though the resulting competitive ratios are different. For the two-agent case, we essentially provide one algorithm and demonstrate how extra communication ability between the two agents affects its competitive ratio.

While the deterministic single-agent algorithms are optimal, it's not clear if the randomized and multi-agent algorithms can be improved. This is an interesting area for future work on this problem. Other areas that deserve more attention are the study of search and rescue for other topologies (i.e. the ring, the plane, and trees/graphs) and for more general multi-agent scenarios (with different starting locations, communication abilities, etc.).

References

1. Bärtschi, A., Tschager, T.: Energy-efficient fast delivery by mobile agents. In: Klasing, R., Zeitoun, M. (eds.) Fundamentals of Computation Theory - 21st International Symposium, FCT 2017, Bordeaux, France, September 11-13, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10472, pp. 82–95. Springer (2017). https://doi.org/10.1007/978-3-662-55751-8_8, https://doi.org/10.1007/978-3-662-55751-8_8
2. Beck, A.: On the linear search problem. Israel Journal of Mathematics **2**(4), 221–228 (1964)
3. Borowiecki, P., Das, S., Dereniowski, D., Kuszner, L.: Distributed evacuation in graphs with multiple exits. In: Suomela, J. (ed.) Structural Information and Communication Complexity - 23rd International Colloquium, SIROCCO 2016, Helsinki, Finland, July 19-21, 2016, Revised Selected Papers. Lecture Notes in Computer Science, vol. 9988, pp. 228–241 (2016). https://doi.org/10.1007/978-3-319-48314-6_15, https://doi.org/10.1007/978-3-319-48314-6_15
4. Bose, P., Carufel, J.D., Durocher, S.: Revisiting the problem of searching on a line. In: Bodlaender, H.L., Italiano, G.F. (eds.) Algorithms - ESA 2013 - 21st Annual European Symposium, Sophia Antipolis, France, September 2-4, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8125, pp. 205–216. Springer (2013). https://doi.org/10.1007/978-3-642-40450-4_18, https://doi.org/10.1007/978-3-642-40450-4_18
5. Carvalho, I.A., Erlebach, T., Papadopoulos, K.: On the fast delivery problem with one or two packages. J. Comput. Syst. Sci. **115**, 246–263 (2021). https://doi.org/10.1016/j.jcss.2020.09.002, https://doi.org/10.1016/j.jcss.2020.09.002
6. Cenek, E.: Chases and escapes by paul j. nahin. SIGACT News **40**(3), 48–50 (2009). https://doi.org/10.1145/1620491.1620500, https://doi.org/10.1145/1620491.1620500
7. Chalopin, J., Jacob, R., Mihalák, M., Widmayer, P.: Data delivery by energy-constrained mobile agents on a line. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II. Lecture Notes in Computer Science, vol. 8573, pp. 423–434. Springer (2014). https://doi.org/10.1007/978-3-662-43951-7_36, https://doi.org/10.1007/978-3-662-43951-7_36
8. Chrobak, M., Gasieniec, L., Gorry, T., Martin, R.: Group search on the line. In: Italiano, G.F., Margaria-Steffen, T., Pokorný, J., Quisquater, J., Wattenhofer, R. (eds.) SOFSEM 2015: Theory and Practice of Computer Science - 41st International Conference on Current Trends in Theory and Practice of Computer Science, Pec pod Sněžkou, Czech Republic, January 24-29,

2015. Proceedings. Lecture Notes in Computer Science, vol. 8939, pp. 164–176. Springer (2015). https://doi.org/10.1007/978-3-662-46078-8_14, https://doi.org/10.1007/978-3-662-46078-8_14
9. Clark, L., Galante, J., Krishnamachari, B., Psounis, K.: A queue-stabilizing framework for networked multi-robot exploration. IEEE Robotics Autom. Lett. **6**(2), 2091–2098 (2021). https://doi.org/10.1109/LRA.2021.3061304, https://doi.org/10.1109/LRA.2021.3061304
10. Coleman, J., Kranakis, E., Krizanc, D., Morales-Ponce, O.: Message delivery in the plane by robots with different speeds. In: Johnen, C., Schiller, E.M., Schmid, S. (eds.) Stabilization, Safety, and Security of Distributed Systems - 23rd International Symposium, SSS 2021, Virtual Event, November 17-20, 2021, Proceedings. Lecture Notes in Computer Science, vol. 13046, pp. 305–319. Springer (2021). https://doi.org/10.1007/978-3-030-91081-5_20, https://doi.org/10.1007/978-3-030-91081-5_20
11. Coleman, J., Kranakis, E., Krizanc, D., Morales-Ponce, O.: The pony express communication problem. In: Flocchini, P., Moura, L. (eds.) Combinatorial Algorithms - 32nd International Workshop, IWOCA 2021, Ottawa, ON, Canada, July 5-7, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12757, pp. 208–222. Springer (2021). https://doi.org/10.1007/978-3-030-79987-8_15, https://doi.org/10.1007/978-3-030-79987-8_15
12. Coleman, J., Kranakis, E., Krizanc, D., Morales-Ponce, O.: Delivery to safety with two cooperating robots. CoRR **abs/2210.04080** (2022). https://doi.org/10.48550/arXiv.2210.04080, https://doi.org/10.48550/arXiv.2210.04080
13. Coleman, J., Kranakis, E., Krizanc, D., Morales-Ponce, O.: Line search for an oblivious moving target. CoRR **abs/2211.03686** (2022). https://doi.org/10.48550/arXiv.2211.03686, https://doi.org/10.48550/arXiv.2211.03686
14. Corless, R.M., Gonnet, G.H., Hare, D.E.G., Jeffrey, D.J., Knuth, D.E.: On the lambert W function. Adv. Comput. Math. **5**(1), 329–359 (1996). https://doi.org/10.1007/BF02124750, https://doi.org/10.1007/BF02124750
15. Czyzowicz, J., Georgiou, K., Kranakis, E., Narayanan, L., Opatrny, J., Vogtenhuber, B.: Evacuating robots from a disk using face-to-face communication. Discret. Math. Theor. Comput. Sci. **22**(4) (2020), http://dmtcs.episciences.org/6732
16. Czyzowicz, J., Georgiou, K., Kranakis, E.: Group search and evacuation. In: Flocchini, P., Prencipe, G., Santoro, N. (eds.) Distributed Computing by Mobile Entities, Current Research in Moving and Computing, Lecture Notes in Computer Science, vol. 11340, pp. 335–370. Springer (2019). https://doi.org/10.1007/978-3-030-11072-7_14, https://doi.org/10.1007/978-3-030-11072-7_14
17. Czyzowicz, J., Killick, R., Kranakis, E., Krizanc, D., Narayanan, L., Opatrny, J., Pankratov, D., Shende, S.M.: Group evacuation on a line by agents with different communication abilities **212**, 57:1–57:24 (2021). https://doi.org/10.4230/LIPIcs.ISAAC.2021.57, https://doi.org/10.4230/LIPIcs.ISAAC.2021.57
18. Czyzowicz, J., Kranakis, E., Krizanc, D., Narayanan, L., Opatrny, J.: Search on a line with faulty robots. Distributed Comput. **32**(6), 493–504 (2019). https://doi.org/10.1007/s00446-017-0296-0, https://doi.org/10.1007/s00446-017-0296-0
19. Devillez, H., Egressy, B., Fritsch, R., Wattenhofer, R.: Two-agent tree evacuation. In: Jurdzinski, T., Schmid, S. (eds.) Structural Information and Communication Complexity - 28th International Colloquium, SIROCCO 2021, Wrocław,

Poland, June 28 - July 1, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12810, pp. 204–221. Springer (2021). https://doi.org/10.1007/978-3-030-79527-6_12, `https://doi.org/10.1007/978-3-030-79527-6_12`

20. Feinerman, O., Korman, A., Lotker, Z., Sereni, J.: Collaborative search on the plane without communication. In: Kowalski, D., Panconesi, A. (eds.) ACM Symposium on Principles of Distributed Computing, PODC '12, Funchal, Madeira, Portugal, July 16-18, 2012. pp. 77–86. ACM (2012). https://doi.org/10.1145/2332432.2332444, `https://doi.org/10.1145/2332432.2332444`
21. Gal, S.: Search games. Wiley encyclopedia of operations research and management science (2010)
22. Georgiou, K., Karakostas, G., Kranakis, E.: Search-and-fetch with 2 robots on a disk: Wireless and face-to-face communication models. arXiv preprint arXiv:1611.10208 (2016)
23. Georgiou, K., Karakostas, G., Kranakis, E.: Search-and-fetch with 2 robots on a disk - wireless and face-to-face communication models. In: Liberatore, F., Parlier, G.H., Demange, M. (eds.) Proceedings of the 6th International Conference on Operations Research and Enterprise Systems, ICORES 2017, Porto, Portugal, February 23-25, 2017. pp. 15–26. SciTePress (2017). https://doi.org/10.5220/0006091600150026, `https://doi.org/10.5220/0006091600150026`
24. Georgiou, K., Kranakis, E., Leonardos, N., Pagourtzis, A., Papaioannou, I.: Optimal circle search despite the presence of faulty robots. In: Dressler, F., Scheideler, C. (eds.) Algorithms for Sensor Systems - 15th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2019, Munich, Germany, September 12-13, 2019, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11931, pp. 192–205. Springer (2019). https://doi.org/10.1007/978-3-030-34405-4_11, `https://doi.org/10.1007/978-3-030-34405-4_11`
25. Kao, M., Reif, J.H., Tate, S.R.: Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem pp. 441–447 (1993), `http://dl.acm.org/citation.cfm?id=313559.313848`
26. Kleinberg, J.M.: On-line search in a simple polygon. In: Sleator, D.D. (ed.) Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms. 23-25 January 1994, Arlington, Virginia, USA. pp. 8–15. ACM/SIAM (1994), `http://dl.acm.org/citation.cfm?id=314464.314473`
27. Queralta, J.P., Taipalmaa, J., Pullinen, B.C., Sarker, V.K., Gia, T.N., Tenhunen, H., Gabbouj, M., Raitoharju, J., Westerlund, T.: Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. IEEE Access **8**, 191617–191643 (2020). https://doi.org/10.1109/ACCESS.2020.3030190, `https://doi.org/10.1109/ACCESS.2020.3030190`
28. Queralta, J.P., Taipalmaa, J., Pullinen, B.C., Sarker, V.K., Gia, T.N., Tenhunen, H., Gabbouj, M., Raitoharju, J., Westerlund, T.: Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. IEEE Access **8**, 191617–191643 (2020). https://doi.org/10.1109/ACCESS.2020.3030190, `https://doi.org/10.1109/ACCESS.2020.3030190`
29. Spieser, K., Frazzoli, E.: The cow-path game: A competitive vehicle routing problem. In: Proceedings of the 51th IEEE Conference on Decision and Control, CDC 2012, December 10-13, 2012, Maui, HI, USA. pp. 6513–6520. IEEE

(2012). https://doi.org/10.1109/CDC.2012.6426279, https://doi.org/10.1109/CDC.2012.6426279

30. Yao, A.C.: Probabilistic computations: Toward a unified measure of complexity (extended abstract). In: 18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977. pp. 222–227. IEEE Computer Society (1977). https://doi.org/10.1109/SFCS.1977.24, https://doi.org/10.1109/SFCS.1977.24