Energy Consumption of Group Search on a Line

Jurek Czyzowicz
Université du Québec en Outaouais, Gatineau, Québec, Canada

Konstantinos Georgiou
Department of Mathematics, Ryerson University, Toronto, Ontario, Canada

Ryan Killick
School of Computer Science, Carleton University, Ottawa, Ontario, Canada

Evangelos Kranakis
School of Computer Science, Carleton University, Ottawa, Ontario, Canada

Danny Krizanc
Department of Mathematics & Comp. Sci., Wesleyan University, Middletown, CT, USA

Manuel Lafond
Department of Computer Science, Université de Sherbrooke, Sherbrooke, Québec, Canada

Lata Narayanan
Department of Comp. Sci. and Software Eng., Concordia University, Montreal, Québec, Canada

Jaroslav Opatrny
Department of Comp. Sci. and Software Eng., Concordia University, Montreal, Québec, Canada

Sunil Shende
Department of Computer Science, Rutgers University, Camden, NJ, USA

Abstract

Consider two robots that start at the origin of the infinite line in search of an *exit* at an unknown location on the line. The robots can collaborate in the search, but can only communicate if they arrive at the same location at exactly the same time, i.e. they use the so-called *face-to-face* communication model. The group search time is defined as the worst-case time as a function of d, the distance of the exit from the origin, when both robots can reach the exit. It has long been known that for a single robot traveling at unit speed, the search time is at least $9d - o(d)$; a simple doubling strategy achieves this time bound. It was shown recently in [15] that $k \geq 2$ robots traveling at unit speed also require at least $9d$ group search time.

We investigate *energy-time trade-offs* in group search by two robots, where the energy loss experienced by a robot traveling a distance x at constant speed s is given by s^2x, as motivated by energy consumption models in physics and engineering. Specifically, we consider the problem of minimizing the total energy used by the robots, under the constraints that the search time is at most a multiple c of the distance d and the speed of the robots is bounded by b. Motivation for this study is that for the case when robots must complete the search in $9d$ time with maximum speed one ($b = 1;\ c = 9$), a single robot requires at least $9d$ energy, while for two robots, all previously proposed algorithms consume at least $28d/3$ energy.

When the robots have bounded memory and can use only a constant number of fixed speeds, we generalize an algorithm described in [3, 15] to obtain a family of algorithms parametrized by pairs of b, c values that can solve the problem for the entire spectrum of these pairs for which the problem is solvable. In particular, for each such pair, we determine optimal (and in some cases nearly optimal) algorithms inducing the lowest possible energy consumption.

We also propose a novel search algorithm that simultaneously achieves search time $9d$ and consumes energy $8.42588d$. Our result shows that two robots can search on the line in optimal time $9d$ while consuming less total energy than a single robot within the same search time. Our algorithm uses robots that have unbounded memory, and a finite number of dynamically computed speeds. It can be generalized for any c, b with $cb = 9$, and consumes energy $8.42588b^2d$.

2012 ACM Subject Classification Computing methodologies → Mobile agents; Theory of computation → Online algorithms

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 137; pp. 137:1–137:15

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Keywords and phrases Evacuation, Exit, Line, Face-to-face Communication, Robots, Search

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.137

Category Track C: Foundations of Networks and Multi-Agent Systems: Models, Algorithms and Information Management

Related Version Full version hosted on arXiv `https://arxiv.org/abs/1904.09714`.

Funding Research supported by NSERC discovery grants, NSERC graduate scholarship, and NSF.

1 Introduction

The problem of searching for a treasure at an unknown location in a specified continuous domain was initiated over fifty years ago [6, 7]. Search domains that have been considered include the infinite line [2, 6, 7, 34], a set of rays [10, 11], the unit circle [12, 24, 37], and polygons [27, 33, 35]. Consider a robot (sometimes called a mobile agent) starting at some known location in the domain and looking for *an exit* that is located at an unknown distance d away from the start. What algorithm should the robot use to find the exit as soon as possible? The most common cost measure used for the search algorithm is the worst-case *search time*, as a function of the distance d of the exit from the starting position. For a fixed-speed robot, the search time is proportional to the length of the trajectory of the robot. Other measures such as turn cost [28] and different costs for revisiting [9] have also been studied.

We consider for the first time the *energy* consumed by the robots while executing the search algorithm. The energy used by a robot to travel a distance x at speed s is computed as s^2x and is motivated from the concept of viscous drag in fluid dynamics; see Section 2 for details on the energy model. For a single robot searching on the line, the classic *Spiral Search* algorithm (also known as the doubling strategy) has search time $9d$ and is known to be *optimal* when the robot moves with unit speed. Since in the worst case, the robot travels distance $9d$ at unit speed, the energy consumption is $9d$ as well. Clearly, as the speed of the robot increases, the time to find the exit decreases but the energy used increases. Likewise, as the speed of the robot decreases, the time to find the exit increases, while the energy consumption decreases. Thus there is a natural trade-off between the time taken by the robot to search for the exit and the energy consumed by the robot. To investigate this trade-off, we consider the problem of minimizing the total energy used by the robots to perform the search when the speed of the robot is bounded by b, and the time for the search is at most a multiple c of the distance d from the starting point to the exit.

Group search by a set of $k \geq 2$ collaborating robots has recently gained a lot of attention. In this case, the search time is the time when *all* k robots reach the exit. The problem has also been called *evacuation*, in view of the application when it is desired that all robots reach and evacuate from the exit. Two models of communication between the robots have been considered. In the wireless communication model, the robots can instantly communicate with each other at any time and over any distance. In the face-to-face communication model (F2F), two robots can communicate only when in the same place at the same time. In many search domains, and for both communication models, group search by $k \geq 2$ agents has been shown to take less time than search by a single agent; see for example [24, 27].

In this paper, we focus on group search on the line, by two robots using the F2F model. Chrobak *et al* [15] showed that group search in this setting cannot be performed in time less than $9d - o(d)$, regardless of the number of robots, assuming all robots use at most

unit speed. They also describe several strategies that achieve search time $9d$. In the first strategy, the two robots independently perform the Spiral Search algorithm, using unit speed during the entire search. Next, they consider a strategy first described in [3], that we call the *Two-Turn* strategy, whereby two robots head off independently in opposite directions at speed $1/3$; when one of them finds the exit, it moves at unit speed to chase and catch the other robot, after which they both return at unit speed to the exit. Finally, they present a new strategy, called the *Fast-Slow* algorithm in which one robot moves at unit speed, while the other robot moves at speed $1/3$, both performing a spiral search. The doubling strategy is very energy-inefficient, it uses energy $18d$ if the two robots always travel together, or $14D$ if the robots start by moving in opposite directions. The other two algorithms both use energy $28d/3 > 9d$. Interestingly, the two strategies that achieve an energy consumption of $28d/3$ with search time $9d$, both use two different and pre-computed speeds, but are quite different in terms of the robot capacities needed. In the Two-Turn strategy, the robots are extremely simple and use constant memory; they use only three states. In Fast-Slow and Spiral Search, the robots need unbounded memory, and perform computations to determine how far to go before turning and moving in the opposite direction.

Memory capability, time- and speed-bounded search, and energy consumption by a two-robot group search algorithm on the line: these considerations motivate the following questions that we address in our paper:

1. Is there a search strategy for constant-memory robots that has energy consumption $< 9d$?
2. Is there *any* search strategy that uses time $9d$ and energy $< 9d$?

1.1 Our results

We generalize the Two-Turn strategy for any values of c, b. We analyze the entire spectrum of values of c, b for which the problem admits a solution, and for each of them we provide optimal (and in some cases nearly optimal) speed choices for our robots (Theorem 5). In particular, and somewhat surprisingly, our proof makes explicit how for any fixed c the optimal speed choices do not simply "scale" with b; rather more delicate speed choices are necessary to comply with the speed and search time bounds. For the special case of $c \cdot b = 9$, our results match with the specific Two-Turn strategy described in [15]. Our results further show that no Two-Turn strategy can achieve energy consumption less than $9d$ while keeping the search time at $9d$. In fact, we conjecture that this trade-off is impossible for *any* group search strategy that uses only constant memory robots.

In the unbounded-memory model, for the special case of $c = 9$ and $b = 1$, we give a novel search algorithm that achieves energy consumption of $8.42588d$, thus answering the second question above in the affirmative. This result shows that though two robots cannot search faster than one robot on the line [15], somewhat surprisingly, two robots can search using less total energy than one robot, in the same optimal time. Our algorithm uses robots that have unbounded memory, and a finite number of dynamically computed speeds. Note that our algorithm can be generalized for any c, b with $cb = 9$, and utilizes energy $8.42588b^2d$ (Theorem 16).

1.2 Related Work

Several authors have investigated various aspects of mobile robot (agent) search, resulting in an extensive literature on the subject in theoretical computer science and mathematics (e.g., see [1, 30] for reviews). Search by constant-memory robots has been done mainly for finite-state automata (FSA) operating in discrete environments like infinite grids, their

finite-size subsets (labyrinths) and other graphs. The main concern of this research was the feasibility of search, rather than time or energy efficiency. For example, [14] showed that no FSA can explore all labyrinths, while [8] proved that one FSA using two pebbles or two FSAs, communicating according to the F2F model can explore all labyrinths. However, no collection of FSAs may explore all finite graphs communicating in the F2F model [39] or wireless model [17]. On the other hand, all graphs of size n may be explored using a robot having $O(\log n)$ memory [38].

Exploration of infinite grids is known as the ANTS problem [29], where it was shown that four collaborating FSAs in the semi-synchronous execution model and communicating according to the F2F scenario can explore an infinite grid. Recently, [13] showed that four FSAs are really needed to explore the grid (while three FSAs can explore an infinite band of the 2-dimensional grid).

Continuous environment cases have been investigated in several papers when the efficiency of the search is often represented by the time of reaching the target (e.g., see [2, 6, 7, 34]). Even in the case of continuous environment as simple as the infinite line, after the seminal papers [6, 7], various scenarios have been studied where the turn cost has been considered [28], the environment was composed of portions permitting different search speeds [26], some knowledge about the target distance was available [10] or where some other parameters are involved in the computation of the cost function [9] (e.g. when the target is moving).

The group search, sometimes interpreted as the evacuation problem has been studied first for the disc environment under the F2F [12, 18, 24, 27, 36] and wireless [18] communication scenarios and then also for other geometric environments (e.g., see [27]). Other variants of search/evacuation problems with a combinatorial flavour have been recently considered in [16, 19, 20, 31, 32]. Some papers investigated the line search problem in the presence of crash faulty [25] and Byzantine faulty agents [23]. The interested reader may also consult the recent survey [22] on selected search and evacuation topics.

The energy used by a mobile robot is usually considered as being spent solely for travelling. As a consequence, in the case of a single, constant speed robot the search time is proportional to the distance travelled and the energy used by a robot. Therefore the problems of minimization of time, distance or energy are usually equivalent for most robots' tasks. For teams of collaborating robots, the searchers often need to synchronize their walks in order to wait for information communicated by other searchers (e.g, see [12, 18, 36]), hence the time of the task and the distance travelled are different. However, the distance travelled by a robot and its energy used are still commensurable quantities.

To the best of our knowledge, energy consumption as a function of mobile robot speed which is based on laws of engineering physics (related to the *drag force*) has never been studied in the search literature before. Our present work is motivated by [15], which proves that the competitive ratio 9 is tight for group search time with two mobile agents in the F2F model when both agents have unit maximal speeds. More exactly, it follows from [15] that having more unit-speed robots cannot improve the group search time obtained by a single robot. Nevertheless, our paper shows that using more robots can improve the energy spending, while keeping the group-search time still the best possible.

Chrobak et al [15] present interesting examples of group search algorithms for two distinct speed robots communicating according to the F2F scenario. An interested reader may consult [4], where optimal group search algorithms for a pair of distinct maximal speed robots were proposed for both communication scenarios (F2F and wireless) and for any pair of robots' maximal speeds. It is interesting to note that, according to [4], for any distinct-speed robots with F2F communication, the optimal group search time is obtained only if one of the robots perform the search step not using its full speed.

Paper Organization. In Section 2 we formally define the evacuation problem EE_c^b, and proper notions of efficiency. Our algorithms and their analysis for constant-memory robots is presented in Section 3, while in Section 4 we introduce and analyze algorithms for unbounded-memory robots. Whenever we omit proofs, due to space limitations, we provide an outline of our arguments. The interested reader may consult the full version of our paper [21] for the missing details.

2 Preliminaries

Two robots start walking from the origin of an infinite (bidirectional) line in search of a hidden exit at an unknown absolute distance d from the origin. The exit is considered found only when one of the robots walks over it. An algorithm for group search by two robots specifies trajectories for both robots and terminates when both robots reach the exit. The time by which the second robot reaches the exit is referred to as the *search time* or the *evacuation time*.

Robot models. The two robots operate under the F2F communication model in which two robots can communicate only when they are in the same place at the same time. Each robot can change its speed at any time. We distinguish between *constant-memory* robots that can only travel at a constant number of hard-wired speeds, and *unbounded-memory* robots that can dynamically compute speeds and distances, and travel at any possible speed.

Energy model. A robot moving at constant speed s traversing an interval of length x is defined to use energy $s^2 \cdot x$. This model is well motivated from first principles in physics and engineering and corresponds to the energy loss experienced by an object moving through a viscous fluid [5]. In particular, an object moving with constant speed s will experience a *drag* force F_D proportional[1] to s^2. In order to maintain the speed s over a distance x the object must do work equal to the product of F_D and x resulting in a continuous energy loss proportional to the product of the object's squared speed and travel distance. For simplicity we have taken the proportionality constant to be one.

The total energy that a robot uses traveling at speeds $s_1, s_2, \ldots, s_t$, traversing intervals $x_1, x_2, \ldots, x_t$, respectively, is defined as $\sum_{i=1}^{t} s_i^2 \cdot x_i$. For group search with two robots, the *energy consumption* is defined as the sum total of the two robots' energies used until the search algorithm terminates.

For each $d > 0$ there are two possible locations for the exit to be at distance d from the origin: we will refer to either of these as input instances d for the group search problem. Our goal is to solve the following *optimized* search problem parametrized by two values, $\mathbf{b}$ and $\mathbf{c}$:

▶ **Problem $\mathrm{EE}_{\mathbf{c}}^{\mathbf{b}}$.** Design a group search algorithm for two robots in the F2F model that minimizes the energy consumption for d-instances under the constraints that the search time is no more than $\mathbf{c} \cdot \mathbf{d}$ and the robots use speeds that are at most $\mathbf{b}$. When there are no speed limits on the robots (i.e. $b = \infty$), we abbreviate EE_c^{∞} by EE_c. Note that b, c are inputs to the algorithm, but d and the exact location of the exit are not known.

[1] The constant of proportionality has (SI) units kg/m and depends, among other things, on the shape of the object and the density of the fluid through which it moves.

As it is standard in the literature on related problems, we assume that the exit is at least a known constant distance away from the origin. In this work, we pick the constant equal to 2, although our arguments can be adjusted to any other constant. It is not difficult to show that EE_c^b is well defined for each $b, c > 0$ with $bc \geq 1$, and the optimal offline solution, for instance d, is for both robots to move at speed $\frac{1}{c}$ to the exit. This offline algorithm has energy consumption $\frac{2d}{c^2}$. Consider an online algorithm for EE_c^b, which on any instance d has energy consumption at most $e(c, b, d)$. The *competitive ratio* of the algorithm is defined as $\sup_{d>0} \frac{c^2}{2d}\ e(c, b, d)$.

Due to [15], and when $b = 1$, no online algorithm (for two robots) can have evacuation time less than $9d - \epsilon$ (for any $\epsilon > 0$ and for large enough d). By scaling, using arbitrary speed limit b, we obtain the following fact.

▶ **Observation 1.** *No online F2F algorithm can solve* EE_c^b *if* $cb < 9$.

3 Solving EE_c^b with Constant-Memory Robots

In this section we propose a family of algorithms for solving EE_c^b (including $b = \infty$). The family uses an algorithm that is parametrized by three discrete speeds: s, r and k. The robots use these speeds depending on finite state control as follows:

▶ **Algorithm $\mathcal{N}_{s,r,k}$.** Robots start moving in opposite directions with speed s until the exit is found by one of them. The finder changes direction and moves at speed $r > s$ until it catches the other robot. Together the two robots return to the exit using speed k.

▶ **Lemma 2.** *Let* b, c *be such that there exist* s, r, k *for which* $\mathcal{N}_{s,r,k}$ *is feasible. Then, for instance* d *of* EE_c^b*, the induced evacuation time of* $\mathcal{N}_{s,r,k}$ *is* $d \cdot T(s, r, k)$ *and the induced energy consumption is* $2d \cdot E(s, r, k)$*, where*

$$T(s, r, k) := \frac{2(k + r)}{k(r - s)} + \frac{1}{s}, \quad E(s, r, k) := \frac{r}{r - s}\left(s^2 + r^2 + 2k^2\right)$$

We propose a systematic way in order to find optimal values for s, r, k of algorithm $\mathcal{N}_{s,r,k}$ for optimization problem EE_c^b (including $b = \infty$), whenever such values exist.

▶ **Theorem 3.** *Algorithm* $\mathcal{N}_{s,r,k}$ *gives rise to a feasible solution to problem* EE_c^b *if and only if* $bc \geq 9$*. For every such* $b, c > 0$*, the optimal choices of* $\mathcal{N}_{s,r,k}$ *can be obtained by solving Non Linear Program:*

$$\begin{aligned} &\min_{s,r,k \in \mathbb{R}} E(s, r, k) && (\mathrm{NLP}_c^b)\\ s.t.\ \ &T(s, r, k) \leq c \\ &r \geq s \\ &0 \leq s, r, k \leq b \end{aligned}$$

where functions $E(\cdot, \cdot, \cdot), T(\cdot, \cdot, \cdot)$ *are as in Lemma 2. Moreover, if* s_0, r_0, k_0 *are the optimizers to* NLP_c^b*, then the competitive ratio of* $\mathcal{N}_{s_0,r_0,k_0}$ *equals* $c^2 \cdot E(s_0, r_0, k_0)$.

The following subsections are devoted to solving NLP_c^b, effectively proving Theorem 5. First in Section 3.1 we solve the case $b = \infty$ and we use our findings to solve the case of bounded speeds b in the follow-up Section 3.2.

3.1 Optimal Choices of $\mathcal{N}_{s,r,k}$ for the Unbounded-Speed Problem

In this section we propose solutions to the unbounded-speed problem EE_c. Since EE_c is the same as $\mathrm{EE}_c^\infty$the problem is well-defined for every fixed $c > 0$. Moreover, by the proof of Theorem 3 algorithm $\mathcal{N}_{s,r,k}$ induces a feasible solution for every $c > 0$ as well, and the optimal speeds can be found by solving NLP_c^∞. Indeed, in the remainder of the section we show how to choose optimal values for s, r, k for solving EE_c with $\mathcal{N}_{s,r,k}$. Let

$$\sigma \approx 2.65976, \rho \approx 11.3414, \kappa \approx 6.63709, \tag{1}$$

whose exact values are the roots of an algebraic system and will be formally defined later. The main theorem of this section reads as follows.

▶ **Theorem 4.** *Let σ, ρ, κ as in (1). For every $c > 0$, the optimal speeds of $\mathcal{N}_{s,r,k}$ for problem EE_c are $s = \frac{\sigma}{c}$, $r = \frac{\rho}{c}$, $k = \frac{\kappa}{c}$. Moreover, the competitive ratio of the corresponding solution is independent of c and equals $\frac{\rho\left(2\kappa^2+\rho^2+\sigma^2\right)}{\rho-\sigma} \approx 292.369$.*

A high level outline of the proof of Theorem 4 is as follows. First we show that any optimal choices of the speeds of $\mathcal{N}_{s,r,k}$ must satisfy the time constraint of NLP_c^∞ tightly. Then, we show that finding optimal speeds s, r, k of $\mathcal{N}_{s,r,k}$ for the general problem EE_c reduces to problem EE_1. Finally, we obtain the optimal solution to NLP_1^∞ by standard tools of nonlinear programming (KKT conditions).

3.2 (Sub)Optimal Choices of $\mathcal{N}_{s,r,k}$ for the Bounded-Speed Problem

In this section, we show how to choose optimal values for s, r, k for solving EE_c^b with $\mathcal{N}_{s,r,k}$, for the entire spectrum of c, b values for which the problem is solvable by online algorithms.

The main result of this section is the following:

▶ **Theorem 5.** *Let $\gamma_1 \approx 9.06609$, $\gamma_2 = \rho \approx 11.3414$, and σ, ρ, κ as in (1). For every $c, b > 0$ with $cb \geq 9$, the following choices of speeds s, r, k are feasible for $\mathcal{N}_{s,r,k}$*

	$9 \leq cb \leq \gamma_1$	$\gamma_1 < cb < \gamma_2$	$cb \geq \gamma_2$
s	$\frac{-\sqrt{(bc)^2-10bc+9}+bc-3}{2c}$	$0.532412b - 0.0262661b^2c$	σ/c
r	b	b	ρ/c
k	b	$\frac{2bs}{bcs-b-cs^2-s}$	κ/c

The induced competitive ratio is given by:

$$f(x) := \begin{cases} \frac{1}{2}x\left(x\left(x-\sqrt{(x-9)(x-1)}\right)+\sqrt{(x-9)(x-1)}+3\right), & 9 \leq x \leq \gamma_1 \\ \frac{x^2\left((0.532412-0.0262661x)^2+\frac{11595.8(20.2699-1.x)^2}{(x(x-2.46798)-398.916)+2221.18)^2}+1\right)}{0.0262661x+0.467588}, & \gamma_1 < x < \gamma_2 \\ 292.369 & x \geq \gamma_2 \end{cases}$$

and the induced energy, for instances d, is $f(cb)\frac{2d}{c^2}$. Moreover, the competitive ratio depends only on the product cb.

In particular, the speeds' choices are optimal when $cb \leq \gamma_1$ and when $cb \geq \gamma_2$. When $\gamma_1 < cb < \gamma_2$, the derived competitive ratio is no more than 0.03 additively off from that induced by optimal choices of s, r, k.

▶ **Corollary 6.** *For $c = 9, b = 1$, the bounded-memory robot algorithm $\mathcal{N}_{s,r,k}$ has energy consumption $28d/3$ and competitive ratio 378.*

Theorem 5 is proven by solving NLP_c^b of Theorem 3. Speed values s, r, k, are chosen optimally when cb is either at most γ_1 or at least γ_2 (i.e. optimizers to NLP_c^b admit analytic description). The optimal speed parameters when $\gamma_1 < cb < \gamma_2$ cannot be determined analytically (they are roots of high degree polynomials). The values that appear in Theorem 5 are heuristically chosen, but interestingly induce nearly optimal competitive ratio.

The proof of Theorem 5 is given by Lemma 7 (the case $cb \leq \gamma_1$), Lemma 8 (the case $cb \geq \gamma_2$), and Lemma 9 (the case $\gamma_1 < cb < \gamma_2$). Next we state these Lemmata, and we sketch their proofs.

▶ **Lemma 7.** *For every $c \in (9/b, \gamma_1/b]$, where $\gamma_1 \approx 9.06609$, the optimizers to NLP_c^b are $k = r = b$, and $s_b = \frac{-\sqrt{(bc)^2-10bc+9}+bc-3}{2c}$. The induced competitive ratio is $f(cb)$, (see definition of $f(x)$ for $x \leq \gamma_1$ in statement of Theorem 5), and the energy consumption, for instances d, is $f(cb)\frac{2d}{c^2}$.*

For proving Lemma 7, first we recall the known optimizer for the special case $cb = 9$, and we identify the tight constraints. Requiring that the exact same inequality constraints to NLP_c^b remain tight, we ask how large can the product cb be so as to have KKT condition hold true. From the corresponding algebraic system, we obtain the answer $cb \leq \gamma_1 \approx 9.06609$.

Similarly, from Theorem 4 we know the optimizers to NLP_c^b for large enough values of cb, and the corresponding tight constraints to the NLP. Again, using KKT conditions, we show that the same constraints remain tight for the optimizers as long as $cb \geq \gamma_2 \approx 11.3414$. This way we obtain the following Lemma.

▶ **Lemma 8.** *For every $c > \rho/b \approx 11.3414/b$, the optimal speeds of $\mathcal{N}_{s,r,k}$ for EE_c^b are $s = \sigma/c,\ r = \rho/c,\ k = \kappa/c$, i.e. they are the same as for EE_c^∞. If the target is placed at distance d from the origin, then the induced energy equals $584.738\frac{d}{c^2}$. Moreover, the induced competitive ratio is 292.369, and is independent of b, c.*

The case $\gamma_1 < cb < \gamma_2$ can be solved optimally only numerically, since the best speed values are obtained by roots to a high degree polynomial. Nevertheless, the following lemma proposes a heuristic choice of speeds (that of Theorem 5) which is surprisingly close to the optimal.

▶ **Lemma 9.** *The choices of s, r, k of Theorem 5 when $\gamma_1 < cb < \gamma_2$ are feasible. Moreover, the induced competitive ratio is at most 0.03 additively off from the competitive ratio induced by the optimal choices of speeds (evaluated numerically).*

The trick in order to find "good enough" optimizers to NLP_c^b is to guess the subset of inequality constraints that remain tight when $\gamma_1 < cb < \gamma_2$. First, we observe that constraint $r \leq b$ is tight for the provable optimizers for all c, b when $cb \in [9, \gamma_1] \cup [\gamma_2, \infty)$. As the only other constraint that switches from being tight to non-tight in the same interval is $k \leq b$, we are motivated to maintain tightness for constraints $r \leq b$ and the time constraint. Still the algebraic system associated with the corresponding KKT conditions cannot be solved analytically. To bypass this difficulty, and assuming we know (optimal) speed s, we use the tight time constraint to find speed k as a function of c, b, s. From numerical calculations, we see that optimal speed s is nearly optimal in c, and so we heuristically set $s = \alpha c + \beta$. We choose α, β so as to have s satisfy optimality conditions for the boundary values $cb = \gamma_1, \gamma_2$. After we identify all parameters to our solution, we compare the value of our solution to the optimal one (obtained numerically), and we verify (using numerical calculations) that our heuristic solution is only by at most 0.03 additively off. The advantage of our analysis is that we obtain closed formulas for the speed parameters for all values of $cb \geq 9$.

4 Solving EE_c^b with Unbounded-Memory Robots

In this section we prove Theorem 16, that is we solve EE_c^b by assuming that the two robots have unbounded memory, and in particular that they can perform time and state dependent calculations and tasks. Note that, by scaling, our results hold for all b, c for which $cb = 9$. For simplicity our exposition is for the natural case $c = 9$ and $b = 1$. Also, as before, d will denote the unknown distance to the exit from the origin. Moreover, the exit is still assumed, for the purposes of performance analysis, to be at least 2 away from the origin.

Throughout the execution of our evacuation algorithm, robots can be in 3 different states (similar to the case of constant-memory robots). First, both robots start with the *Exploration State* and they remain in this until the exit is located. While in the exploration state, robots execute an elaborate exploration that requires synchronous movements in which robots, at a high level, stay in good proximity, still they expand the searched space relatively fast. Then, the exit finder enters the *Chasing State* in which the robot, depending on its distance from the origin, calculates a speed at which to move in order to catch and notify the other robot. Lastly, when the two robots meet, they both enter the *Exit State* in which both robots move toward the exit with the smallest possible speed while meeting the time constraint.

Our algorithm takes as input the values of $c = 9, b = 1$, and use a speed value $s \leq b$, that will be chosen later. When the exit finder switches its state from Exploration to Chasing, it remembers the distance d of the exit to the origin, as well as the value k of a counter that was used while in the Exploration State. When the exit finder catches the other robot, they both switch to the Exit State, and they remember their distance p from the origin, as well as the value of time t that their rendezvous was realized. The speed of their Exit State will be determined as a function of p, d, t (and hence of s, c, b as well).

4.1 A Critical Component: l-Phase Explorations

We adopt the language of [15] in order to discuss a structural property that any feasible evacuation algorithm for EE_9^1 satisfies. As a result, the purpose of this section is to provide high level intuition for our evacuation algorithm that is presented in subsequent sections.

We refer to the two robots (starting exploration from the origin) as L and R, intended to explore to the left and to the right of the origin, respectively. The robot trajectories can be drawn on the Cartesian plane where point-location $(x, -t)$ will correspond to point x on the line being visited by some robot at time t. The following Theorem (due to [15]) was originally phrased for the time-evacuation unit-speed robots' problem. We adopt the language of our problem.

▶ **Theorem 10** ([15]). *For any feasible solution to* EE_9^1*, the point-location of any robot lies within the cone spanned by vectors* $\binom{-1}{-3}, \binom{1}{-3}$.

Next we present some preliminaries toward describing our k-phase exploration algorithms. A *phase* is a pair (s, r) where $s \in [0, 1]$ is a speed and $r \in \mathbb{R}$ is a *distance ratio*, possibly negative. An *l-phase algorithm* is determined by a position p_0 on the line and a sequence $S = (s_1, r_1), \ldots, (s_k, r_l)$ of l phases (movement instructions). Whenever $r_i x < 0$, movement will be to the left, whereas $r_i x > 0$ will correspond to movement to the right.

l-PHASE EXPLORATION: GIVEN p_0 AND $S = (s_1, r_1), \ldots, (s_l, r_l)$
Go to p_0 at speed $1/3$
repeat
  $x \leftarrow$ current position
  for $i = 1, \ldots, l$ **do**
    Travel at speed s_i for a distance of $r_i \cdot x$
  end
end

We will make sure that each time the loop is executed, position x and corresponding time induce point-locations of the robots that lie in the boundary of the cone of Theorem 10. If a loop starts at location x, then it takes additional time $\sum_{i\in[l]} \frac{|r_i||x|}{s_i}$ to complete one iteration. We will be referring to quantity $1 + \sum_{i\in[l]} \frac{|r_i|}{3s_i}$ as the expansion factor of Exploration S.

4.2 Algorithm $\mathcal{A}(s)$: The Exploration, Chasing and Exit States

In this section we give a formal description of our evacuation algorithm. The most elaborate part of it is when robots are in Exploration States, in which they will perform 3-phase exploration. It can be shown that 3-phase exploration based evacuation algorithms that do not violate the constraints of problem EE_9^1 have expansion factor at most 4. Moreover, among those, the ones who minimize the induced energy consumption makes robots move at speed 1 in the first and third phase[2]. Robot's speed in the second phase will be denoted by s.

We now present a specific 3-phase exploration algorithm, that we denote by $\mathcal{A}(s)$, complying with the above conditions, with phases $(-1, 1), (4s/(1-s), s)$ and $(4-4s/(1-s), 1)$, where s is an exploration speed to be determined later. Robot L will execute the 3-phase exploration with starting position -1, while robot R with starting position 2. When subroutine $travel(v, p)$ is invoked, the robot sets its speed to v and, from its current position, goes toward position p on the line until it reaches it. We depict the trajectories of the robots while in the Exploration State in Figure 1.

EXPLORATION STATE OF L
$travel(1/3, -1)$
$k \leftarrow 0$
repeat
  $travel(1, 0)$
  $travel(s, -4^{k+1} \cdot \frac{s}{1-s})$
  $travel(1, -4^{k+1})$
  $k \leftarrow k + 1$
end

EXPLORATION STATE OF R
$travel(1/3, 2)$
$k \leftarrow 0$
repeat
  $travel(1, 0)$
  $travel(s, 2 \cdot 4^{k+1} \cdot \frac{s}{1-s})$
  $travel(1, 2 \cdot 4^{k+1})$
  $k \leftarrow k + 1$
end

[2] The proof of these facts is lengthy and technical, and is not required for the correctness of our algorithm, rather it only justifies some parameter choices

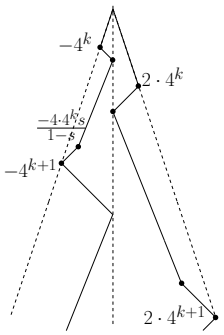

Figure 1 A representation of position (x-axis, vertical dashed line is 0) and time (y-axis), and the trajectory followed by the two robots (solid lines). The two diagonal dashed lines form the "1/3 cone" of Theorem 10.

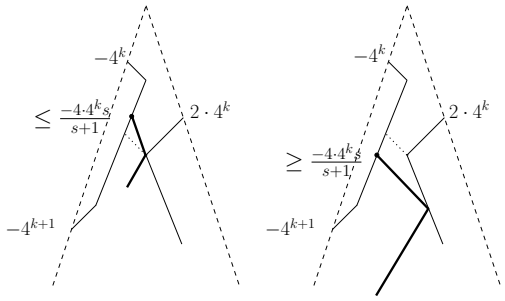

Figure 2 The robots' behavior when the exit is found by L is indicated by the bold line. In the first case (left), the catch-up speed is slower than 1 (and the rendezvous is realized at the turning point of the non-finder), whereas it is 1 in the second case (right).

A complete execution of one repeat loop within the Exploration State will be referred to as a *round*. Variable k counts the number of completed rounds. Each robot stays in the Exploration State till the exit it found. When switching to the Chasing state (which happens only for the exit finder), robot remembers its current value of counter k, as well as the distance d of the exit to the origin. Based on these values (as well as s) it calculates the most efficient trajectory in order to catch the other robot (predicting, when applicable, that the rendezvous can be realized while the other robot is approaching the exit finder). When the rendezvous is realized, robots store their current distance p to the origin, as well as the time t that has already passed. Then, robots need to travel distance $p + d$ to reach the exit. Knowing they have time $9d - t$ remaining, they go to the exit together as slow as possible to reach the exit in time exactly $9d$. Figure 2 provides an illustration of the behavior of the robots after finding the exit.

Chasing State
$K \leftarrow 4^k$
if *I am R* **then**
  $K \leftarrow 2 \cdot 4^k$
end
$s' \leftarrow \min\left\{\frac{d}{4K - d/s}, 1\right\}$
Travel toward the other robot at speed s' until meeting it at distance p from the origin, and at time t.

Exit State
$\bar{s} \leftarrow \frac{p+d}{9d-t}$
Go toward the exit with speed $\bar{s}$.

4.3 Performance Analysis & an Optimal Choice for Parameter s

In this section we are ready to provide the details for proving Theorem 16. Evacuation algorithm $\mathcal{A}(s)$ is not feasible to EE_c^b for all values of speed parameter s (of the Exploration States). We will show later that trajectories induce evacuation time at most $9d$ only if $s \in [1/3, 1/2]$. In what follows, and even though we have not fixed the value of s yet, we

will assume that s has some value between $1/3$ and $1/2$. The purpose of this section is to fix a value for parameter s, show that $\mathcal{A}(s)$ is feasible to EE_9^1, and subsequently compute the induced energy consumption and competitive ratio. As a reminder, each iteration of the repeat loop of the Exploration States is called a round, and k is a counter for these rounds.

▶ **Proposition 11.** *For every $k \geq 0$, and at the start of its k-th round,*
- *robot L is at position -4^k at time $3 \cdot 4^k$, and*
- *robot R, is at position $2 \cdot 4^k$ at time $6 \cdot 4^k$.*

Let $X \in \{L, R\}$ be one of the robots. We define $K(X,k) = 4^k$ if $X = L$, and $K(X,k) = 2 \cdot 4^k$ if $X = R$, i.e. the position of X at the start of round k. We will often analyze 3 cases for the distance d of the exit with respect to $K := K(X,k)$ (as it also appears in the description of the Chasing State), associated with the following closed intervals

$$D_1(K) := [K, 4Ks/(s+1)],\ D_2(K) := [4Ks/(s+1), 4Ks/(1-s)],\ D_3(K) := [4Ks/(1-s), 4K].$$

We may simply write D_1, D_2 and D_3 if K is clear from the context. Note that during the second phase of round K, robot L explores D_1 and D_2, whereas D_3 is explored during the third phase. The same statement holds for R. The following lemma will be useful in analyzing the worst case evacuation time and energy consumption of our algorithm.

▶ **Lemma 12.** *Suppose that robot $X \in \{L, R\}$ finds the exit at distance d when its round counter has value k. Let p and t be, respectively, the position and time at which X first meets with the other robot after having found the exit, and set $K := K(X,k)$. Then the following hold:*
1. *If $d \in D_1$, then $p = 0$ and $t = 8K$.*
2. *If $d \in D_2$, then $|p| = \frac{d+ds-4Ks}{1-s}$ and $t = 8K + \frac{d+d/s-4K}{1-s}$.*
3. *If $d \in D_3$, then $|p| = 2ds/(1-s)$ and $t = 8K + 2d + 2ds/(1-s)$.*

Using the lemma above, we can now prove that $\mathcal{A}(s)$ meets the speed bound and the evacuation time bound.

▶ **Lemma 13.** *For any $s \in [1/3, 1/2]$, evacuation algorithm $\mathcal{A}(s)$ is feasible to EE_9^1.*

Lemma 12 allows us to derive the speed s_{b1}, s_{b2} and s_{b3} at which both robots go toward the exit after meeting for the cases $d \in D_1, d \in D_2$ and $d \in D_3$, respectively. We also know the speed s_{c1} at which the exit-finder catches up to the other robot when $d \in D_1$. We define

$$s_{b1} := \tfrac{d}{9d-8K},\quad s_{c1} = \tfrac{d}{4K-d/s},\quad s_{b2} := \tfrac{2d-4Ks}{d(8-9s-1/s)+4K(2s-1)},\quad s_{b3} := \tfrac{d(1+s)}{d(7-9s)+8K(s-1)}$$

The speed s_{b2} is a simple rearrangement of the speed $\frac{d+qs}{9d-(8K+q)}$, where $q = \frac{d+d/s-4K}{1-s}$, and s_{b3} is obtained by rearranging $\frac{d+2ds/(1-s)}{9d-(8K+2d+2ds/(1-s))}$.

Next we compute the energy consumption. For given K, d and s, denote by $E_L(K,d,s)$ the energy spent by robot L from time 3 to time $9d$ when it exits. Similarly, $E_R(K,d,s)$ is the energy spent by R from time 6 to time $9d$. Then, the energy consumption is $E(K,d,s) := \frac{1}{3} + E_L(K,d,s) + E_R(K,d,s)$. For any K and s, we also define $F(K,s) := (K-1)(5-4s(s+1))$.

▶ **Lemma 14.** *Suppose that robot $X \in \{L, R\}$ finds the exit at distance d when its round counter has value k, and let $K := K(X,k)$. Then*

$$E(K,d,s) = \frac{1}{3} + \begin{cases} F(K,s) + 3K + d(s^2 + s_{c1}^2 + 2s_{b1}^2) & \text{if } d \in D_1 \\ F(K,s) + 3K + \left(\dfrac{2d-4Ks}{1-s}\right)(1 + s^2 + 2s_{b2}^2) & \text{if } d \in D_2 \\ F(K,s) + 3K - 4Ks(s+1) + \dfrac{2d}{1-s}(s^3 + s_{b3}^2(s+1) + 1) & \text{if } d \in D_3. \end{cases}$$

Denote by $E_i(k,d,s)$ the value of $E(K,d,s)$ when $d \in D_i$, $i = 1,2,3$. Our intension now is to fix speed value s that solves the following Nonlinear Program

$$\min_{s\in[1/3,1/2]} \left\{ \max \left\{ \sup_{d\in D_1, k\geq 1, X} \frac{E_1(K,d,s)}{d}, \sup_{d\in D_2, k\geq 1, X} \frac{E_2(K,d,s)}{d}, \sup_{d\in D_3, k\geq 1, X} \frac{E_3(K,d,s)}{d} \right\} \right\}.$$

For every $s \in [1/3, 1/2]$ we show in Lemma 15 that $\frac{E_1(K,d,s)}{d}$ is decreasing in $d \in D_1$, that $\frac{E_2(K,d,s)}{d}$ is increasing in $d \in D_2$, and that $\frac{E_3(K,d,s)}{d}$ is decreasing in $d \in D_3$. Then, the best parameter s can be chosen so as to make all worst case valued $\frac{E_i(K,d,s)}{d}$ equal (if possible) when $i = 1,2,3$. The optimal s can be found by numerically finding the roots of a high degree polynomial, and accordingly, we heuristically set $s = 0.39403$, inducing the best possible energy consumption for algorithm $\mathcal{A}(s)$. All relevant formal arguments are within the proof of the next lemma.

▶ **Lemma 15.** *On instance d of EE_9^1, algorithm $\mathcal{A}(s)$ induces energy consumption at most $8.42588d$, when $s = 0.39403$.*

By Lemma 15, we conclude that for the specific value of s, algorithm $\mathcal{A}(s)$ has competitive ratio $\frac{9^2}{2} 8.42588 \approx 341.24814$, concluding the proof of Theorem 16.

▶ **Theorem 16.** *For every $c, b > 0$ with $cb = 9$, there is an evacuation algorithm for unbounded-memory autonomous robots solving EE_c^b inducing energy consumption $8.42588b^2 d$ for instances d, and competitive ratio 341.24814.*

5 Discussion

The main contribution of our paper was to introduce an energy consumption model appropriate to linear search and investigate how the F2F communication model affects time/energy trade-offs until completion of the search by two robots, considering two different computational capabilities for the robots. Our approach inspired new algorithms that take better account of the impact of the change in the speed of the robots during the course of the search and leads to better understanding through evaluation of trade-offs of the overall performance of the algorithms.

Our paper raises several interesting problems worth investigating. In addition to improving the trade-offs in the algorithms proposed, one may wish to pursue new avenues for research by examining additional search domains, like the unit disk, in the spirit of [18]. It would also be natural to consider more realistic models of linear search with multiple agents some of which may be faulty [23, 25]. Further, it would be interesiting to investigate randomized search algorithms in our setting as well as more general models in which the energy loss experienced by a robot traveling a distance x at constant speed s is given by $s^a x$, for some fixed positive exponent a.

References

1. S. Alpern and S. Gal. *The theory of search games and rendezvous*. Springer, 2003.
2. R. Baeza Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Information and Computation*, 106(2):234–252, 1993.
3. R. Baeza-Yates and R. Schott. Parallel searching in the plane. *Computational Geometry*, 5(3):143–154, 1995.
4. E. Bampas, J. Czyzowicz, L. Gasieniec, D. Ilcinkas, R. Klasing, T. Kociumaka, and D. Pajak. Linear search by a pair of distinct-speed robots. *Algorithmica*, 81(1):317–342, 2019.

5 G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000.

6 A. Beck. On the linear search problem. *Israel J. of Mathematics*, 2(4):221–228, 1964.

7 R. Bellman. An optimal search. *SIAM Review*, 5(3):274–274, 1963.

8 M. Blum and D. Kozen. On the power of the compass (or, why mazes are easier to search than graphs). In *FOCS*, pages 132–142, 1978.

9 P. Bose and J.-L. De Carufel. A General Framework for Searching on a Line. *Theoretical Computer Science*, pages 703:1–17, 2017.

10 P. Bose, J.-L. De Carufel, and S. Durocher. Searching on a line: A complete characterization of the optimal solution. *Theoretical Computer Science*, pages 569:24–42, 2015.

11 S. Brandt, K.-T. Foerster, B. Richner, and R. Wattenhofer. Wireless Evacuation on m Rays with k Searchers. In *SIROCCO*, pages 140–157, 2017.

12 S. Brandt, F. Laufenberg, Y. Lv, D. Stolz, and R. Wattenhofer. Collaboration without Communication: Evacuating Two Robots from a Disk. In *CIAC*, pages 104–115, 2017.

13 S. Brandt, J. Uitto, and R. Wattenhofer. A Tight Lower Bound for Semi-Synchronous Collaborative Grid Exploration. In *DISC*, pages 13:1–13:17, 2018.

14 L. Budach. Automata and labyrinths. *Math. Nachrichten*, 86:195–282, 1978.

15 M. Chrobak, L. Gasieniec, Gorry T., and R. Martin. Group Search on the Line. In *SOFSEM*, pages 164–176. Springer, 2015.

16 H. Chuangpishit, K. Georgiou, and P. Sharma. Average Case - Worst Case Tradeoffs for Evacuating 2 Robots from the Disk in the Face-to-Face Model. In *ALGOSENSORS'18*. Springer, 2018.

17 S. A. Cook and C. Rackoff. Space lower bounds for maze threadability on restricted machines. *SIAM Journal on Computing*, 9(3):636–652, 1980.

18 J. Czyzowicz, L. Gasieniec, T. Gorry, E. Kranakis, R. Martin, and D. Pajak. Evacuating Robots via Unknown Exit in a Disk. In *DISC*, pages 122–136. Springer, 2014.

19 J. Czyzowicz, K. Georgiou, R. Killick, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, and S. Shende. God Save the Queen. In *(FUN)*, pages 16:1–16:20, 2018.

20 J. Czyzowicz, K. Georgiou, R. Killick, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, and S. Shende. Priority Evacuation from a Disk Using Mobile Robots. In *SIROCCO*, pages 209–225, 2018.

21 J. Czyzowicz, K. Georgiou, R. Killick, E. Kranakis, M. Lafond, D. Krizanc, L. Narayanan, J. Opatrny, and S. Shence. Energy Consumption of Group Search on a Line. *CoRR*, abs/1904.09714, 2019. `arXiv:1904.09714`.

22 J. Czyzowicz, K. Georgiou, and E. Kranakis. Group Search and Evacuation. In *Distributed Computing by Mobile Entities, Current Research in Moving and Computing, LNCS*, volume 11340, pages 335–370, 2019.

23 J. Czyzowicz, K. Georgiou, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, and S. Shende. Search on a Line by Byzantine Robots. In *ISAAC*, pages 27:1–27:12, 2016.

24 J. Czyzowicz, K. Georgiou, E. Kranakis, L. Narayanan, J. Opatrny, and B. Vogtenhuber. Evacuating Robots from a Disc Using Face to Face Communication. In *CIAC 2015*, pages 140–152, 2015.

25 J. Czyzowicz, E. Kranakis, D. Krizanc, L. Narayanan, and Opatrny J. Search on a Line with Faulty Robots. In *PODC*, pages 405–414. ACM, 2016.

26 J. Czyzowicz, E. Kranakis, D. Krizanc, L. Narayanan, J. Opatrny, and M. Shende. Linear Search with Terrain-Dependent Speeds. In *CIAC*, pages 430–441, 2017.

27 J. Czyzowicz, E. Kranakis, K. Krizanc, L. Narayanan, J. Opatrny, and S. Shende. Wireless Autonomous Robot Evacuation from Equilateral Triangles and Squares. In *ADHOCNOW*, pages 181–194. Springer, 2015.

28 E. D. Demaine, S. P. Fekete, and S. Gal. Online searching with turn cost. *Theoretical Computer Science*, 361(2):342–355, 2006.

29 Y. Emek, T. Langner, D. Stolz, J. Uitto, and R. Wattenhofer. How Many Ants Does it Take to Find the Food? *Theor. Comput. Sci.*, page 608:255–267, 2015.

30 S. Gal. *Search Games*. Wiley Encyclopedia for Operations Research and Management Science, 2011.

31 K. Georgiou, G. Karakostas, and E. Kranakis. Search-and-Fetch with One Robot on a Disk - (Track: Wireless and Geometry). In *ALGOSENSORS*, pages 80–94, 2016.

32 K. Georgiou, G. Karakostas, and E. Kranakis. Search-and-Fetch with 2 Robots on a Disk - Wireless and Face-to-Face Communication Models. In *ICORES*, pages 15–26. SciTePress, 2017.

33 F. Hoffmann, C. Icking, R. Klein, and K. Kriegel. The polygon exploration problem. *SIAM Journal on Computing*, 31(2):577–600, 2001.

34 M.-Y. Kao, J. H. Reif, and S. R. Tate. Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. *Information and Computation*, 131(1):63–79, 1996.

35 J. Kleinberg. On-line search in a simple polygon. In *SODA*, pages 8–15. SIAM, 1994.

36 I. Lamprou, R. Martin, and S. Schewe. Fast Two-Robot Disk Evacuation with Wireless Communication. In *DISC*, pages 1–15, 2016.

37 D. Pattanayak, H. Ramesh, P.S. Mandal, and S. Schmid. Evacuating Two Robots from Two Unknown Exits on the Perimeter of a Disk with Wireless Communication. In *ICDCN*, pages 20:1–20:4, 2018.

38 O. Reingold. Undirected st-connectivity in log-space. In *STOC*, pages 376–385, 2005.

39 H.-A. Rollik. Automaten in planaren Graphen. *Acta Informatica*, 13(3):287– 298, 1980.