# Collaborative Search on the Plane without Communication

[Extended Abstract]

Ofer Feinerman*
The Louis and Ida Rich Career Development Chair, The Weizmann Institute of Science
Rehovot, Israel
feinermanofer@gmail.com

Amos Korman†
CNRS (LIAFA, University Diderot
Paris, France
amos.korman@liafa.jussieu.fr

Zvi Lotker
Ben-Gurion University of the Negev
Beer-Sheva, Israel
zvilo@bgu.ac.il

Jean-Sébastien Sereni‡
CNRS (LIAFA, University Diderot)
Paris, France
sereni@kam.mff.cuni.cz

## ABSTRACT

We use distributed computing tools to provide a new perspective on the behavior of cooperative biological ensembles. We introduce the *Ants Nearby Treasure Search (ANTS)* problem, a generalization of the classical cow-path problem [10, 20, 41, 42], which is relevant for collective foraging in animal groups. In the ANTS problem, $k$ identical (probabilistic) agents, initially placed at some central location, collectively search for a treasure in the two-dimensional plane. The treasure is placed at a target location by an adversary and the goal is to find it as fast as possible as a function of both $k$ and $D$, where $D$ is the distance between the central location and the target. This is biologically motivated by cooperative, central place foraging, such as performed by ants around their nest. In this type of search there is a strong preference to locate nearby food sources before those that are further away. We focus on trying to find what can be achieved if communication is limited or altogether absent. Indeed, to avoid overlaps agents must be highly dispersed making communication difficult. Furthermore, if the agents do not commence the search in synchrony, then even initial communication is problematic. This holds, in particular, with respect to the question of whether the agents can communicate and conclude their total number, $k$. It turns out that the knowledge of $k$ by the individual agents is crucial for performance. Indeed, it is a straightforward observation that the time required for finding the treasure is $\Omega(D + D^2/k)$, and we show in this paper that this bound can be matched if the agents have knowledge of $k$ up to some constant approximation.

We present a tight bound for the competitive penalty that must be paid, in the running time, if the agents have no information about $k$. Specifically, this bound is slightly more than logarithmic in the number of agents. In addition, we give a lower bound for the setting in which the agents are given some estimation of $k$. Informally, our results imply that the agents can potentially perform well without any knowledge of their total number $k$, however, to further improve, they must use some information regarding $k$. Finally, we propose a uniform algorithm that is both efficient and extremely simple, suggesting its relevance for actual biological scenarios.

## Categories and Subject Descriptors

F.2.m [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity—*Miscellaneous*; G.2.1 [**Discrete Mathematics**]: Combinatorics—*Combinatorial algorithms*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Graph algorithms*

## General Terms

Algorithms, Theory

## Keywords

search algorithms, mobile robots, speed-up, cow-path problem, online algorithms, uniform algorithms, social insects, collective foraging, ants

*Supported by the Israel Science Foundation (grant 1694/10).

†Supported by the ANR projects DISPLEXITY and PROSE, and by the INRIA project GANG.

‡This author's work was partially supported by the French *Agence Nationale de la Recherche* under reference ANR 10 JCJC 0204 01.


*PODC'12,* July 16–18, 2012, Madeira, Portugal.

# 1. INTRODUCTION

## 1.1 Background and Motivation

The universality of search behaviors is reflected in multitudes of studies in different fields including control systems, distributed computing and biology. We use tools from distributed computing to study a biologically inspired scenario in which a group of agents, initially located at one central location, cooperatively search for treasures in the plane. The goal of the search is to locate nearby treasures as fast as possible and at a rate that scales well with the number of participating agents.

A variety of animal species search for food around a central location that serves the search's initial point, final destination or both [47]. This central location could be a food storage area, a nest where offspring are reared or simply a sheltered or familiar environment. Central place foraging holds a strong preference to locating nearby food sources before those that are further away. Possible reasons for that are, for example: (1) decreasing predation risk [44], (2) increasing the rate of food collection once a large quantity of food is found [47], (3) holding a territory without the need to reclaim it [35, 44, 46], and (4) the ease of navigating back after collecting the food using familiar landmarks [16].

Searching in groups can increase foraging efficiency [38, p. 732]. In some extreme cases, food is so scarce that group searching is believed to be required for survival [18, 39]. Proximity of the food source to the central location is again important in this case. For example, in the case of recruitment, a nearby food source would be beneficial not only to the individual that located the source but also increase the subsequent retrieval rate for many other collaborators [60]. Foraging in groups can also facilitate the defense of larger territories [55]. Eusocial insects (e.g., bees and ants) engage in highly cooperative foraging, which can be expected as these insects reduce competition between individuals to a minimum and share any food that is found. Social insects often live in a single nest or hive, which naturally makes their foraging patterns central.

Little is known about the communication between foragers, but it is believed that in some scenarios communication may become impractical [36]. This holds, for example, if the foragers start the search at different times and remain far apart (which may be necessary to avoid unnecessary overlaps). Hence, the question of how efficient the search can be if the communication is limited, or altogether absent, is of great importance.

In this paper, we theoretically address general questions of collective searches in the particular natural setting described above. More precisely, we introduce the *Ants Nearby Treasure Search (ANTS)* problem, in which $k$ identical (probabilistic) agents, initially placed at some central location, collectively search for a treasure in the two-dimensional plane. The treasure is placed by an adversary at some target location at distance $D$ from the central location, where $D$ is unknown to the agents. The goal of the agents is to find the treasure as fast as possible, where the time complexity is evaluated as a function of both $k$ and $D$.

In the context of search algorithms, evaluating the time as a function of $D$ was first introduced in a classical paper by Baeza-Yates *et al.* [10], who studied the *cow-path* problem (studied also in [20, 41, 42]). The ANTS problem generalizes the cow-path problem, as it considers multiple identical agents instead of a single agent (a cow in their terminology). Indeed, in this distributed setting, we are concerned with the *speed-up* measure (see also, [8, 9, 27, 29]), which aims to capture the impact of using $k$ searchers in comparison to using a single one. Note that the objectives of quickly finding nearby treasures and having significant speed-up may be at conflict. That is, in order to ensure that nearby treasures are quickly found, a large enough fraction of the search force must be deployed near the central location. In turn, this crowding can potentially lead to overlapping searches that decrease individual efficiency.

It is a rather straightforward observation that the time required for finding the treasure is $\Omega(D + D^2/k)$. Our focus is on the question of how agents can approach this bound if their communication is limited or even completely absent. In particular, as information of the size of the foraging group may not be available to the individual searchers, we concentrate our attention on the question of how important it is for agents to know (or estimate) their total number. As we later show, the lack of such knowledge may have a non-negligible impact on the performance.

## 1.2 Our Results

We introduce and investigate the ANTS problem, a generalization of the cow-path problem. First, we show that if the agents have a constant approximation of their total number $k$, then there exists a rather simple search algorithm whose expected running time is $O(D + D^2/k)$, making it $O(1)$-competitive. We then turn our attention to *uniform* searching algorithms, in which the agents are not assumed to have any information regarding $k$. We completely characterize the speed-up penalty that must be paid when using uniform algorithms. Specifically, we show that for a function $f$ such that $\sum_{j=1}^{\infty} 1/f(j)$ converges, there exists a uniform search algorithm that is $O(f(\log k))$-competitive. On the other hand, we show that if $\sum_{j=1}^{\infty} 1/f(j)$ diverges, then there is no uniform search algorithm that is $O(f(\log k))$-competitive. In particular, this implies that, for every constant $\varepsilon > 0$, there exists a uniform search algorithm that is $O(\log^{1+\varepsilon} k)$-competitive, but there is no uniform search algorithm that is $O(\log k)$-competitive. Hence, the penalty for using uniform algorithms is slightly more than logarithmic in the number of agents.

In addition, we give a lower bound for the intermediate setting in which the agents are given some estimation of $k$. As a special case, this lower bound implies that for any constant $\varepsilon > 0$, if each agent is given a (one-sided) $k^{\varepsilon}$-approximation of $k$, then the competitiveness is $\Omega(\log k)$. Informally, our results imply that the agents can potentially perform well without any knowledge of their total number $k$, however, to further improve they must use some information regarding $k$. Finally, we propose a uniform search algorithm that is concurrently efficient and extremely simple and, as such, it may imply some relevance for actual biological scenarios.

## 1.3 Related Work

Our work falls within the scope of natural algorithms, a recent attempt to investigate biological phenomena from an algorithmic perspective [1, 13, 15].

Collective search is a classical problem that has been extensively studied in different fields of science. Group living and food sources that have to be actively sought after make collective foraging a widespread biological phenomenon. So-

cial foraging theory [33] makes use of economic and game theory to optimize food exploitation as a function of the group size and the degree of cooperativity between agents in different environmental settings. Social foraging theory has been extensively compared to experimental data (see, e.g., [6, 34]) but does not typically account for the spatial characteristics of resource abundance. Central place foraging theory [47] assumes a situation in which food is collected from a patchy resource and is returned to a particular location, such as a nest. This theory is used to calculate optimal durations for exploiting food patches at different distances from the central location and has also been tested against experimental observations [35, 37]. Collective foraging around a central location is particularly interesting in the case of social insects where large groups forage cooperatively with, practically, no competition between individuals. Harkness and Mardouras [36] have conducted a joint experimental and modeling research into the collective search behavior of non-communicating desert ants. Modeling the ants' trajectories using biased random walks, they reproduce some of the experimental findings and demonstrate significant speed-up with group size. In bold contrast to these random walks, Reynolds [54] argues that Lévy flights with a power law that approaches unity is the optimal search strategy for cooperative foragers as traveling in straight lines tends to decrease overlaps between searchers.

From an engineering perspective, the distributed cooperation of a team of autonomous agents (often referred to as robots or UAVs: Unmanned Aerial Vehicles) is a problem that has been extensively studied. These models extend single agent searches in which an agent with limited sensing abilities attempts to locate one or several mobile or immobile targets [49]. The memory and computational capacities of the agent are typically large and many algorithms rely on the construction of cognitive maps of the search area that includes current estimates that the target resides in each point [64]. The agent then plans an optimal path within this map with the intent, for example, of optimizing the rate of uncertainty decrease [40]. Cooperative searches typically include communication between the agents that can be transmitted up to a given distance, or even without any restriction. Models have been suggested where agents can communicate by altering the environment to which other agent then react [61]. Cooperation without communication has also been explored to some extent [7] but the analysis puts no emphasis on the speed-up of the search process. In addition, to the best of our knowledge, no works exist in this context that put emphasis on finding nearby targets faster than faraway one. Similar problems studied in this context are pattern formation [19, 58, 59], rendezvous [2, 28], and flocking [32]. It is important to stress that in all those engineering works, the issue of whether the robots know their total number is typically not addressed, as obtaining such information does not seem to be problematic. Furthermore, in many works, robots are not identical and have unique identities.

In the theory of computer science, the exploration of graphs using mobile agents is a central question. Most of the research for graph exploration is concerned with the case of a single deterministic agent exploring a finite graph (typically, with some restrictions on the resources of the agent and/or on the graph structure). For example, in [3, 12, 21, 26] the agent explores strongly connected directed finite graphs, and in [5, 22, 23, 24, 31, 48, 50] the agent explores undirected finite graphs. When it comes to probabilistic searching, the random walk is a natural candidate, as it is extremely simple, uses no memory, and trivially self-stabilizes. Unfortunately, however, the random walk turns out to be inefficient in a two-dimensional infinite grid. Specifically, in this case, the expected hitting time is infinite, even if the treasure is nearby.

Evaluating the time to find the treasure as a function of $D$, the initial distance to the treasure, was studied in the context of the cow-path problem. One of the first papers that studied the cow-path problem is a paper by Baeza-Yates *et al.* [10], in which the competitive ratio for deterministically finding a point on the real line is shown to be nine. Considering the two-dimensional case, Baeza-Yates *et al.* proved that the spiral search algorithm is optimal up to lower order terms. Randomized algorithms for the problem were studied by Kao *et al.* [41], for the infinite star topology. Karp *et al.* [42] studied an early variant of the cow-path problem on a binary tree. Recently, Demaine *et al.* [20] considered the cow-path problem with a double component price: the first is distance and the second is turn cost. López-Ortiz and Sweet [45] extended the cow-path problem by considering $k$ agents. However, in contrast to the ANTS problem, the agents they consider are not identical, and the goal is achieved by (centrally) designing a different specific path for each of the $k$ agents.

In general, the more complex setting of using multiple identical agents has received much less attention. Exploration by deterministic multiple agents was studied in, e.g., [8, 9, 27, 29]. To obtain better results when using several identical deterministic agents, one must assume that the agents are either centrally coordinated or that they have some means of communication (either explicitly, or implicitly, by being able to detect the presence of nearby agents). When it comes to probabilistic agents, analyzing the speed-up measure for $k$-random walkers has recently gained attention. In a series of papers, initiated by Alon *et al.* [4], a speed-up of $\Omega(k)$ is established for various finite graph families, including, in particular, expanders and random graphs [4, 17, 25]. While some graph families enjoy linear speed-up, for many graph classes, to obtain linear speed-up, $k$ has to be quite small. In particular, this is true for the two-dimensional $n$-node grid, where a linear speed up is obtained when $k < O(\log^{1-\varepsilon} n)$. On the other hand, the cover time of 2-dimensional $n$-node grid is always $\Omega(n/\log k)$, regardless of $k$. Hence, when $k$ is polynomial in $n$, the speed up is only logarithmic in $k$. The situation with infinite grids is even worse. Specifically, though the $k$-random walkers would find the treasure with probability one, the expected time to find the treasure becomes infinite.

The question of how important it is for individual processors to know their total number has recently been addressed in the context of locality. Generally speaking, it has been observed that for several classical local computation tasks, knowing the number of processors is not essential [43]. On the other hand, in the context of local decision, some evidence exist that such knowledge may be crucial for non-deterministic distributed decision [30].

## 2. PRELIMINARIES

We consider the ANTS problem, where $k$ mobile *agents* (robots) are searching for a *treasure* on the two-dimensional plane. Each agent has a field of view bounded by a small positive constant, hence, for simplicity, we can assume that the

agents are actually walking on the integer two-dimensional infinite grid $G := \mathbf{Z}^2$. All $k$ agents starts the search from a central node $s$ of $G$, called the *source*. An adversary locates the treasure at some node $\tau$ of $G$, referred to as the *target* node; the agents have no a priori information about the location of $\tau$. The goal of the agents it to *find* the treasure: this task is accomplished once at least one of the agents visits the node $\tau$.

The agents are probabilistic machines that can move on the grid, but cannot communicate between themselves. All $k$ agents are identical (execute the same protocol). An agent can traverse an edge of the grid in both directions. We do not restrict the internal storage and computational power of agents, nevertheless, we note that all our upper bounds use simple procedures that can be implemented using relatively short memory. For example, with respect to navigation, our constructions only assume the ability to perform four basic procedures, namely: (1) choose a direction uniformly at random, (2) walk in a "straight" line to a prescribed distance, (3) perform a *spiral search* around a node[1], and (4) return to the source node. On the other hand, for our lower bounds to hold, we do not require any restriction on the navigation capabilities.

Regarding the time complexity, we assume that traversing an edge takes one unit of time. Furthermore, for the ease of presentation, we assume that the agents are synchronous, that is, each edge traversal costs precisely one unit of time (and all internal computations are performed in zero time). Indeed, this assumption can easily be removed if we measure the time according to the slowest edge-traversal. We also assume that all agents start the search simultaneously at the same time, denoted by $t_0$. This assumption can also be easily removed by starting to count the time after the last agent initiates the search. We measure the cost of an algorithm by its *expected running time*, that is, the expected time (from time $t_0$) until at least one of the agents finds the treasure. We denote the expected running time of algorithm $\mathcal{A}$ by $\mathcal{T}_{\mathcal{A}}(D, k)$.

The *distance* $\mathrm{d}(u, v)$ between two nodes $u$ and $v$ of $G$ is simply the hop distance between them, i.e., the number of edges on a shortest path connecting $u$ and $v$ in $G$. The distance between a node $u$ and the source node $s$ is simply written $\mathrm{d}(u)$, i.e. $\mathrm{d}(u) := \mathrm{d}(s, u)$. Let $D$ be the distance between the source node $s$ and the target node $\tau$, i.e., $D := \mathrm{d}(\tau)$. For a node $u$ of $G$ and an integer $r$, let $B(u, r)$ be the ball of radius $r$ centered at the node $u$, formally, $B(u, r) := \{v \in G : \mathrm{d}(u, v) \leq r\}$. For short, we denote the ball of radius $r$ around the source node $s$ by $B(r)$, that is, $B(r) := B(s, r)$.

Note that if an agent knows $D$, then it can potentially find the treasure in time $O(D)$, by going to any node at distance $D$, and then performing a circle around the source of radius $D$ (assuming, of course, that its navigation abilities enable it to perform such a circle). On the other hand, without knowledge about $D$, an agent can find the treasure in time $O(D^2)$ by performing a spiral search around the source. When considering $k$ agents, it is easy to see[2] that the expected running time is $\Omega(D + D^2/k)$, even if the total number of agents $k$ is known to all agents, and even if we relax the model and allow agents to freely communicate between each other. It follows from Theorem 3.1 that if $k$ is known to agents then there exists a search algorithm whose expected running time is asymptotically optimal, namely, $O(D + D^2/k)$. We evaluate the performance of an algorithm that does not assume the precise knowledge of $k$ with respect to this aforementioned optimal time. Formally, let $\phi(k)$ be a function of $k$. An algorithm $\mathcal{A}$ is $\phi(k)$-*competitive* if

$$\mathcal{T}_{\mathcal{A}}(D, k) \leq \phi(k) \cdot (D + D^2/k),$$

for every integers $k$ and $D$. We shall be particularly interested in the performances of *uniform* algorithms: these are algorithms in which no information regarding $k$ is available to agents. (The term "uniform" is chosen to stress that agents execute the same algorithm regardless of their number, see, e.g., [43].)

# 3. UPPER BOUNDS

## 3.1 Optimal Running Time with Knowledge on $k$

Our first theorem asserts that agents can achieve an asymptotically optimal running time if they know the precise value of $k$. As a corollary, it follows that, in fact, to obtain such a bound, it is sufficient to assume that agents only know a constant approximation of $k$.

THEOREM 3.1. *Assume that the agents know their total number, $k$. Then, there exists a (non-uniform) search algorithm running in expected time $O(D + D^2/k)$.*

PROOF. For $i \in \mathbf{N}$, set $B_i := \left\{u : \mathrm{d}(u) \leq 2^i\right\}$. Consider the following algorithm (see also Figure 1 for an illustration).

Fix a positive integer $\ell$ and consider the time $T_\ell$ until each agent completed $\ell$ phases $i$ with $i \geq \log D$. Each time an agent performs phase $i$, the agent finds the treasure if the chosen node $u$ belongs to the ball $B(\tau, \sqrt{t_i}/2)$ around $\tau$, the node holding the treasure. Note that at least some constant fraction of the ball $B(\tau, \sqrt{t_i}/2)$ is contained in $B_i$, since $i \geq \log D$. The probability of choosing a node $u$ in that fraction is thus $\Omega(|B(\tau, \sqrt{t_i}/2)|/|B_i|)$, which is at least $\beta/k$ for some positive constant $\beta$. Thus, the probability that by time $T_\ell$ none of the $k$ agents finds the treasure (while executing their respective $\ell$ phases $i$) is at most $(1 - \beta/k)^{k\ell}$, which is at most $\gamma^{-\ell}$ for some constant $\gamma$ greater than 1.

[1]The spiral search around a node $v$ is a particular deterministic local search algorithm (see, e.g., [10]) that starts at $v$ and enables the agent to visit all nodes at distance $\Omega(\sqrt{x})$ from $v$ by traversing $x$ edges, for every integer $x$. For our purposes, since we are concerned only with asymptotic results, we can replace this atomic navigation procedure with any procedure that guarantees such a property. For simplicity, in the remainder of the paper, we assume that for every integer $x$, the spiral search of length $x$ starting at a node $v$ visits all nodes at distance at most $\sqrt{x}/2$ from $v$

[2]To see why, consider a search algorithm $\mathcal{A}$ whose expected running time is $T$. Clearly, $T \geq D$, because it takes time $D$ to merely reach the treasure. Assume, towards a contradiction, that $T < D^2/4k$. In any execution of $\mathcal{A}$, by time $2T$, the $k$ agents can visit a total of at most $2Tk < D^2/2$ nodes. Hence, by time $2T$, more than half of the nodes in $B_D := \{u : 1 \leq \mathrm{d}(u) \leq D\}$ were not visited. Therefore, there must exist a node $u \in B_D$ such that the probability that $u$ is visited by time $2T$ (by at least one of the agents) is less than $1/2$. If the adversary locates the treasure at $u$, then the expected time to find the treasure is strictly greater than $T$, which contradicts the assumption.

**begin**
Each agent performs the following double loop;
**for** $j$ *from* 1 *to* $\infty$ **do** the *stage* $j$ defined as follows
**for** $i$ *from* 1 *to* $j$ **do** the *phase* $i$ defined as follows
- Go to a node $u$ chosen uniformly at random among the nodes in $B_i$;
- Perform a spiral search for time $t_i := 2^{2i+2}/k$;
- Return to the source $s$;

**end**
**end**
**end**

**Algorithm 1**: The non-uniform algorithm $\mathcal{A}_{\mathrm{k}}$.

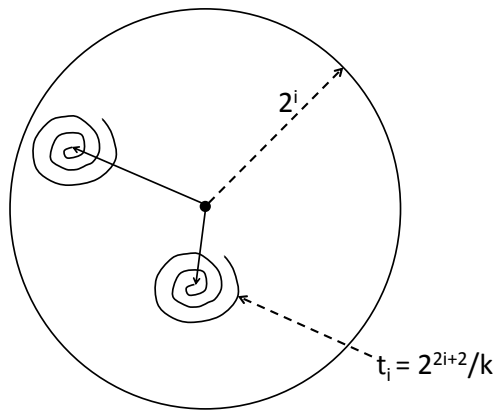


**Figure 1: Illustration of two agents performing phase $i$.**

For $i \in \mathbf{N}$, let $\psi(i)$ be the time required to execute a phase $i$. Note that $\psi(i) = O(2^i + 2^{2i}/k)$. Hence, the time until all agents complete stage $j$ for the first time is

$$\sum_{i=1}^{j} \psi(i) = O(2^j + \sum_{i=1}^{j} 2^{2i}/k) = O(2^j + 2^{2j}/k).$$

Now fix $s := \lceil \log D \rceil$. It follows that for any integer $\ell$, all agents complete their respective stages $s + \ell$ by time $\hat{T}(\ell) := O(2^{s+\ell} + 2^{2(s+\ell)}/k)$. Observe that by this time, all agents have completed at least $\ell^2/2$ phases $i$ with $i \geq s$. Consequently, the probability that none of the $k$ agents finds the treasure by time $\hat{T}(\ell)$ is at most $\gamma^{-\ell^2/2}$. Hence, the expected running time is at most

$$\mathcal{T}_{\mathcal{A}_{\mathrm{k}}}(D,k) = O\left(\sum_{\ell=1}^{\infty} \frac{2^{s+\ell}}{\gamma^{\ell^2/2}} + \frac{2^{2(s+\ell)}}{k\gamma^{\ell^2/2}}\right)$$
$$= O\left(2^s + 2^{2s}/k\right) = O(D + D^2/k).$$

This establishes the theorem. □

Fix a constant $\rho \geq 1$. We say that the agents have a *$\rho$-approximation* of $k$, if, initially, each agent $a$ receives as input a value $k_a$ satisfying $k/\rho \leq k_a \leq k\rho$.

Corollary 3.2. *Fix a constant $\rho \geq 1$. Assume that the agents have a $\rho$-approximation of their total number. Then, there exists a (non-uniform) search algorithm that is $O(1)$-competitive.*

Proof. Each agent $a$ executes Algorithm $\mathcal{A}_{\mathrm{k}}$ (see the proof of Theorem 3.1) with the parameter $k$ equal to $k_a/\rho$. By the definition of a $\rho$-approximation, the only difference between this case and the case where the agents know their total number, is that for each agent, the time required to perform each spiral search is multiplied by a constant factor of at most $\rho^2$. Therefore, the analysis in the proof of Theorem 3.1 remains essentially the same and the running time is increased by a multiplicative factor of at most $\rho^2$. □

## 3.2 Unknown Number of Agents

We now turn our attention to the case of uniform algorithms.

Theorem 3.3. *Consider a non-decreasing function $f\colon \mathbf{N} \to \mathbf{N}$ such that $\sum_{j=1}^{\infty} 1/f(j) < \infty$. There exists a uniform search algorithm that is $O(f(\log k))$-competitive.*

Proof. Consider the uniform search algorithm $\mathcal{A}_{\mathrm{uniform}}$ described below. Let us analyze the performances of the algorithm and show that its expected running time is $T(D,k) := \phi(k) \cdot (D + D^2/k)$, where $\phi(k) := O(f(\log k))$. We first note that it suffices to prove the statement when $k \leq D$. Indeed, if $k > D$, then we may consider only $D$ agents among the $k$ agents and obtain an upper bound on the running time of $T(D,D)$, which is less than $T(D,k)$.

**begin**
Each agent performs the following;
**for** $\ell$ *from* 0 *to* $\infty$ **do** the big-stage $\ell$
**for** $i$ *from* 0 *to* $\ell$ **do** the stage $i$
**for** $j$ *from* 0 *to* $i$ **do** the phase $j$
- $k_j \longleftarrow 2^j$;
- $D_{i,j} \longleftarrow \sqrt{2^{(i+j)}/f(j)}$
- Go to the node $u$ chosen uniformly at random among the nodes in $B(D_{i,j})$;
- Perform a spiral search starting at $u$ for time $t_{i,j} := 2^{i+2}/f(j)$;
- Return to the source;

**end**
**end**
**end**
**end**

**Algorithm 2**: The uniform algorithm $\mathcal{A}_{\mathrm{uniform}}$.

Assertion 1. *For every integer $\ell$, the time until all agents complete big-stage $\ell$ is $O(2^\ell)$.*

For the assertion to hold, it is sufficient to prove that stage $i$ in big-stage $\ell$ takes time $O(2^i)$. To this end, notice that phase $j$ takes time $O(D_{i,j} + 2^i/f(j))$ which is at most $O(2^{(i+j)/2} +$

$2^i/f(j)))$. Therefore, stage $i$ takes time

$$O\left(\sum_{j=0}^{i}(2^{(i+j)/2}+2^i/f(j))\right)=O(2^i).$$

This establishes Assertion 1.

Set $s \coloneqq \lceil \log((D^2 \cdot f(\log k))/k)\rceil + 1$ and fix an integer $i \geq s$. Then, there exists $j \in \{0,\ldots,i\}$ such that $2^j \leq k < 2^{j+1}$.

Assertion 2. *There exists a constant $c \in (0,1)$ such that the probability that none of the agents finds the treasure while executing phase $j$ of stage $i$ is at most $c$.*

To see this, first note that the treasure is inside the ball $B(D_{i,j})$. Indeed, since $i \geq s$, we have

$$D_{i,j}=\sqrt{\frac{2^{i+j}}{f(j)}} \geq \sqrt{\frac{2^{s+j}}{f(j)}} > D.$$

The total number of nodes in the ball $B(D_{i,j})$ is $O(D_{i,j}^2) = O(2^{i+j}/f(j))$, and at least a third of the ball of radius $\sqrt{t_{i,j}}$ around the treasure is contained in $B(D_{i,j})$. Consequently, the probability for an agent $a$ to choose a node $u$ in a ball of radius $\sqrt{t_{i,j}}$ around the treasure in phase $j$ of stage $i$ is

$$\Omega\left(t_{i,j}/|B(D_{i,j})|\right)=\Omega\left(\frac{2^i/f(j)}{2^{i+j}/f(j)}\right)=\Omega\left(2^{-j}\right).$$

If this event happens, then the treasure is found during the corresponding spiral search of agent $a$. As a result, there exists a positive constant $c'$ such that the probability that none of the $k$ agents finds the treasure during phase $j$ of stage $i$ is at most $(1-c'\cdot 2^{-j})^k \leq (1-c'\cdot 2^{-j})^{2^j} \leq e^{-c'}$. This establishes Assertion 2.

By the time that all agents have completed their respective big-stage $s+\ell$, all agents have performed $\Omega(\ell^2)$ stages $i$ with $i \geq s$. By Assertion 2, for each such $i$, the probability that the treasure is not found during stage $i$ is at most $c$ for some constant $c<1$. Hence, the probability that the treasure is not found during any of those $\Omega(\ell^2)$ stages is at most $1/d^{\ell^2}$ for some constant $d>1$. Assertion 1 ensures that all agents complete big-stage $s+\ell$ by time $O(2^{s+\ell})$, so the expected running time is $O(\sum_{\ell=0}^{\infty} 2^{s+\ell}/d^{\ell^2}) = O(2^s) = O(D^2 \cdot f(\log k)/k)$, as desired. □

Setting $f(x) \coloneqq \lceil x^{1+\varepsilon}\rceil$ yields the following corollary.

Corollary 3.4. *For every positive constant $\varepsilon$, there exists a uniform search algorithm that is $O(\log^{1+\varepsilon} k)$-competitive.*

# 4. LOWER BOUNDS

## 4.1 A Tight Lower Bound for Uniform Algorithms

We say that a function $f\colon \mathbf{N} \to \mathbf{N}$ is *sub-exponential* if $f$ is non-decreasing, and there exists a constant $c<2$ such that $f(x+1) < c \cdot f(x)$, for every $x \in \mathbf{N}$.

Theorem 4.1. *Consider a sub-exponential function $f$ such that $\sum_{j=1}^{\infty} 1/f(j) = \infty$. There is no uniform search algorithm that is $O(f(\log k))$-competitive.*

Proof. Suppose that there exists a uniform search algorithm with running time less than $\tau(D,k) \coloneqq (D+D^2/k)\phi'(k)$. Hence, if $k \leq D$, then $\tau(D,k) \leq \frac{D^2\phi(k)}{k}$, where $\phi(k) = 2\phi'(k)$. Suppose, towards a contradiction, that $\phi(k) = O(f(\log k))$.

For an integer $i$, set

$$k_i = 2^i.$$

Let $i_0$ be the first integer such that for every integer $i \geq i_0$, we have $k_i > \phi(k_i)$. (The existence of $i_0$ is guaranteed by the fact that $f$ is sub-exponential). Let $T \geq 2^{2i_0}$ be a (sufficiently large) integer and set $D \coloneqq 2T+1$. That is, for the purpose of the proof, we assume that the treasure is actually placed at some faraway distance $D$ greater than $2T$. This means, in particular, that by time $2T$ the treasure has not been found yet. For every integer $i_0 \leq i \leq \log T/2$, set

$$D_i \coloneqq \sqrt{T \cdot k_i/\phi(k_i)}.$$

Fix an integer $i \in [i_0, \log T/2]$, and consider $B(D_i)$, the ball of radius $D_i$ around the source node. We now deal with the case where the algorithm is executed by $k_i$ agents. For every subset $S$ of nodes in $B(D_i)$, let $\chi(S)$ be the random variable indicating the number of nodes in $S$ that were visited by at least one of the $k_i$ agents by time $2T$. (We write $\chi(u)$ for $\chi(\{u\})$.)

Note that $k_i \leq D_i$ and, therefore, for each node $u \in B(D_i)$ the expected time to visit $u$ is at most $\tau(D_i,k_i) \leq D_i^2\phi(k_i)/k_i = T$. Thus, by Markov's inequality, the probability that $u$ is visited by time $2T$ is at least $1/2$, i.e., $\mathbf{Pr}(\chi(u)=1) \geq 1/2$. Hence, $\mathbf{E}(\chi(u)) \geq 1/2$.

Now consider an integer $i$ in $[i_0, \log T/2]$ and set $S_i \coloneqq B(D_i) \setminus B(D_{i-1})$. By linearity of expectation, $\mathbf{E}(\chi(S_i)) = \sum_{u \in S_i} \mathbf{E}(\chi(u)) \geq |S_i|/2$. Consequently, by time $2T$, the expected number of nodes in $S_i$ that an agent visits is

$$\begin{aligned}\Omega(|S_i|/k_i) &= \Omega\left(\frac{D_{i-1}(D_i-D_{i-1})}{k_i}\right)\\ &= \Omega\left(\frac{T}{\phi(k_{i-1})}\cdot\left(\frac{\sqrt{2\phi(k_{i-1})}}{\sqrt{\phi(k_i)}}-1\right)\right)\\ &= \Omega\left(\frac{T}{\phi(k_i)}\right),\end{aligned}$$

where the second equality follows from the fact that $D_i = D_{i-1}\cdot\sqrt{\frac{2\phi(k_{i-1})}{\phi(k_i)}}$, and the third equality follows from the facts that $\phi(k_i) = O(f(i))$ and $f$ is sub-exponential.

In other words, for every integer $i$ in $[i_0, \log T/2]$, the expected number of nodes in $S_i$ that each agent visits by time $2T$ is $\Omega\left(\frac{T}{\phi(k_i)}\right)$. Since the sets $S_i$ are pairwise disjoint, the linearity of expectation implies that the expected number of nodes that an agent visits by time $2T$ is

$$\Omega\left(\sum_{i=i_0}^{\log T/2}\frac{T}{\phi(k_i)}\right) = T\cdot\Omega\left(\sum_{i=i_0}^{\log T/2}\frac{1}{\phi(2^i)}\right).$$

Consequently, $\sum_{i=i_0}^{\log T/2}\frac{1}{\phi(2^i)}$ must converge as $T$ goes to infinity, and hence so does $\sum_{i=i_0}^{\log T/2}\frac{1}{f(i)}$. This contradicts the assumption on $f$. □

Setting $f(x) \coloneqq x$ yields the following statement.

Corollary 4.2. *There is no uniform search algorithm that is $O(\log k)$-competitive.*

## 4.2 A Lower Bound for Algorithms using an Approximate Knowledge of $k$

We now present a lower bound for the competitiveness of search algorithms assuming that agents are given approximations of their total number $k$. As a special case, our lower bound implies that for any constant $\varepsilon > 0$, if agents are given an estimation $\tilde{k}$ such that $\tilde{k}^{1-\varepsilon} \leq k \leq \tilde{k}$, then the competitiveness is $\Omega(\log k)$. That is, the competitiveness remains logarithmic even for relatively good approximations of $k$.

Formally, let $\varepsilon\colon \mathbf{N} \to (0,1]$. We say that the agents have a *$k^{\varepsilon}$-approximation of $k$* if each agent $a$ receives as input an estimation $\tilde{k}_a$ for $k$ that satisfies:

$$\tilde{k}_a^{1-\varepsilon(\tilde{k}_a)} \leq k \leq \tilde{k}_a.$$

(For example, if the agents have a $k^{\varepsilon}$-approximation of $k$, where $\varepsilon$ is the constant function equal to $1/2$, then, in particular, this means that if all agents are given the same value $\tilde{k}$, then the real number $k$ of agents satisfies $\sqrt{\tilde{k}} \leq k \leq \tilde{k}$.)

Theorem 4.3. *Let $\varepsilon\colon \mathbf{N} \to (0,1]$ and assume that the agents have a $k^{\varepsilon}$-approximation of $k$. If there exists a $\phi(k)$-competitive algorithm, where $\phi$ is a non-decreasing function, then $\phi(k) = \Omega(\varepsilon(k)\log k)$.*

Proof. Assume that there is a search algorithm for this case running in time $(D + D^2/k)\phi(k)$, where $\phi$ is non-decreasing. Suppose that all agents receive the same value $\tilde{k}$, which should serve as an estimate for $k$. Consider an integer $W$ greater than $4\tilde{k}$. Set

$$T := 2W \cdot \phi(\tilde{k}) \quad \text{and} \quad j_0 := \frac{\log W}{2}.$$

For the purposes of the proof, we assume that the treasure is located at distance $D = 2T + 1$, so that by time $2T$ it is guaranteed that no agent finds the treasure.

For $i \in \mathbf{N}$, define

$$S_i := \left\{u \,:\, 2^{j_0+i-1} < \mathrm{d}(u,s) \leq 2^{j_0+i}\right\}.$$

Fix an integer $i$ in $\{\lceil \frac{1-\varepsilon(\tilde{k})}{2}\log\tilde{k}\rceil, \cdots, \lfloor \frac{1}{2}\log\tilde{k}\rfloor\}$. Assume for the time being, that the number of agents is $k_i := 2^{2i}$. Note that

$$\tilde{k}^{1-\varepsilon(\tilde{k})} \leq k_i \leq \tilde{k},$$

hence, $k_i$ is a possible candidate for being the real number of agents. Observe that all nodes in $S_i$ are at distance at most $2^{j_0+i}$ from the source, and that $|S_i| = \Theta(2^{2j_0+2i}) = \Theta(W \cdot k_i)$. By the definition, $j_0 \geq i+1$. Hence, $k_i \leq 2^{j_0+i-1} < \mathrm{d}(u,s)$, and therefore using the expected running time of the algorithm, it follows that for each node $u \in S_i$, the expected time until at least one of the $k_i$ agents covers $u$ is at most

$$\frac{2\,\mathrm{d}(u,s)^2}{k_i} \cdot \phi(k_i) \leq 2W \cdot \phi(\tilde{k}) = T.$$

Recall that we now consider the case where the algorithm is executed with $k_i$ agents. For every subset $S$ of nodes of $G$, let $\chi(S)$ be the random variable indicating the number of nodes in $S$ that were visited by at least one of the $k_i$ agents by time $2T$. (As before, we write $\chi(u)$ for $\chi(\{u\})$.) By Markov's inequality, the probability that $u$ is visited by at least one of the $k_i$ agents by time $2T$ is at least $1/2$, i.e., $\mathbf{Pr}(\chi(u) = 1) \geq 1/2$. Hence, $\mathbf{E}(\chi(u)) \geq 1/2$. By linearity of expectation, $\mathbf{E}(\chi(S_i)) = \sum_{u\in S_i}\mathbf{E}(\chi(u)) \geq |S_i|/2$. Consequently, by time $2T$, the expected number of nodes in $S_i$ that a single agent visits is $\Omega(|S_i|/k_i) = \Omega(W)$.

Since this holds for any $i$ in $\{\lceil \frac{1-\varepsilon(\tilde{k})}{2}\log\tilde{k}\rceil, \cdots, \lfloor \frac{1}{2}\log\tilde{k}\rfloor\}$, and since the sets $S_i$ are pairwise disjoint, the linearity of expectation implies that the expected number of nodes that a single agent visits by time $2T$ is $\Omega(W \cdot \varepsilon(\tilde{k})\log\tilde{k})$. Since $T = 2W \cdot \phi(\tilde{k})$, this implies that $\phi(\tilde{k}) = \Omega(\varepsilon(\tilde{k})\log\tilde{k})$, as desired. This concludes the proof of the theorem. □

# 5. HARMONIC SEARCH

The algorithms described in the Section 3 are relatively simple but still require the use of non trivial iterations, which may be complex for simple and tiny agents, such as ants. If we relax the requirement of bounding the expected running time and demand only that the treasure be found with some low constant probability, then it is possible to avoid one of the loops of the algorithms. However, a sequence of iterations still needs to be performed.

In this section, we propose an extremely simple algorithm, coined the *harmonic search algorithm*[3], which does not perform in iterations and is essentially composed of three components: (1) choose a random direction and walk in this direction for a distance of $d$, chosen randomly according to a distribution in which the probability of choosing $d$ is roughly inverse proportional to $d$, (2) perform a local search (e.g., a spiral search) for time roughly $d^2$, and (3) return to the source. It turns out that this extremely simple algorithm has a good probability of quickly finding the treasure, if the number of agents is sufficiently large.

More specifically, the algorithm depends on a positive constant parameter $\delta$ that is fixed in advance and governs the performance of the algorithm. For a node $u$, let $p(u) := \frac{c}{\mathrm{d}(u)^{2+\delta}}$, where $c$ is the normalizing factor, defined so that $\sum_{u\in V(G)} p(u) = 1$. (Note that $c$ depends on $\delta$.)

**begin**
Each agent performs the following three actions;
1. Go to a node $u \in V(G)$ with probability $p(u)$;
2. Perform a spiral search for time $t(u) := \mathrm{d}(u)^{2+\delta}$;
3. Return to the source;

**end**

**Algorithm 3**: The harmonic search algorithm.

Using arguments similar to those introduced earlier, e.g., in the proofs of Theorems 3.1 and 3.3, one can prove the following result.

Theorem 5.1. *Let $\delta \in (0, 0.8]$. For every $\varepsilon > 0$, there exists a positive real number $\alpha$ such that if the number $k$ of agents is greater than $\alpha D^{\delta}$, then with probability at least $1-\varepsilon$, the running time of the harmonic algorithm is $O(D + \frac{D^{2+\delta}}{k})$.*

[3]The name harmonic was chosen because of structure resemblances to the celebrated harmonic algorithm for the $k$-server problem — see, e.g., [11].

## 6. CONCLUSION AND DISCUSSION

We first presented an algorithm that assumes that agents have a constant approximation of $k$ and runs in optimal expected time, that is, in time $O(D + D^2/k)$. We then showed that there exists a uniform search algorithm whose competitiveness is slightly more than logarithmic. We also presented a relatively efficient uniform algorithm, namely, the harmonic algorithm, that has and extremely simple structure. Our constructions imply that, in the absence of any communication, multiple searchers can still potentially perform rather well. On the other hand, our lower bounds imply that to achieve a better running time, the searchers must either communicate or utilize some information regarding $k$. In particular, even providing each agent with a $k^\varepsilon$-approximation to $k$ (for constant $\varepsilon > 0$) does not suffice to bring the competitiveness strictly below $O(\log k)$.

Although the issue of memory is beyond the scope of this paper, our constructions are simple and can be implemented using relatively low memory. For example, going in a straight line for a distance of $d = 2^\ell$ can be implemented using $O(\log \log d)$ memory bits, by employing a randomized counting technique. In addition, our lower bounds provide evidence that in order to achieve a near-optimal running time, agents must use non-trivial memory size, required merely to store the necessary approximation of $k$. This may be useful to obtain a tradeoff between the running time and the memory size of agents.

From another perspective, it is of course interesting to experimentally verify whether social insects engage in search patterns in the plane which resemble the simple uniform algorithms specified above, and, in particular, the harmonic algorithm. Two natural candidates are desert ants *Cataglyphys* and honeybees *Apis mellifera*. First, these species seem to face settings that are similar to the one we use. Indeed, they cannot rely on communication during the search due to the dispersedness of individuals [36] and their inability to leave chemical trails (this is due to increased pheromone evaporation in the case of the desert ant). Additionally, the task of finding the treasure is relevant, as food sources in many cases are indeed relatively rare or patchy. Moreover, due to the reasons mentioned in Section 1, finding nearby sources of food is of great importance. Second, insects of these species have the behavioral and computational capacity to maintain a compass-directed vector flight [14, 36], measure distance using an internal odometer [56, 57], travel to distances taken from a random power law distribution [53], and engage in spiral or quasi-spiral movement patterns [51, 52, 63]. These are the main ingredients that are needed to perform the algorithms described in this paper. Finally, the search trajectories of desert ants have been shown to include two distinguishable sections: a long straight path in a given direction emanating from the nest and a second more tortuous path within a small confined area [36, 62].

## 7. ACKNOWLEDGMENTS.

The authors are grateful to the anonymous referees for their useful suggestions.

## 8. ADDITIONAL AUTHORS

## 9. REFERENCES


[1] Y. Afek, N. Alon, O. Barad, E. Hornstein, N. Barkai, and Z. Bar-Joseph. A biological solution to a fundamental distributed computing problem. *Science*, 331(6014):183–185, 2011.

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

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

[4] N. Alon, C. Avin, M. Koucký, G. Kozma, Z. Lotker, and M. R. Tuttle. Many random walks are faster than one. *Combin. Probab. Comput.*, 20(4):481–502, 2011.

[5] C. Ambühl, L. Gąsieniec, A. Pelc, T. Radzik, and X. Zhang. Tree exploration with logarithmic memory. *ACM Trans. Algorithms*, 7(2):Art. 17, 21, 2011.

[6] M. Arbilly, U. Motro, M. W. Feldman, and A. Lotem. Co-evolution of learning complexity and social foraging strategies. *J. Theor. Biol.*, 267(4):573–581, 2010.

[7] R. C. Arkin. Cooperation without communication: Multiagent schema-based robot navigation. *J. Rob. Syst.*, 9(3):351–364, 1992.

[8] I. Averbakh and O. Berman. A heuristic with worst-case analysis for minimax routing of two travelling salesmen on a tree. *Discrete Appl. Math.*, 68(1-2):17–32, 1996.

[9] I. Averbakh and O. Berman. $(p-1)/(p+1)$-approximate algorithms for $p$-traveling salesmen problems on a tree with minmax objective. *Discrete Appl. Math.*, 75(3):201–216, 1997.

[10] R. A. Baeza-Yates, J. C. Culberson, and G. J. E. Rawlins. Searching in the plane. *Inform. and Comput.*, 106(2):234–252, 1993.

[11] Y. Bartal and E. Grove. The harmonic $k$-server algorithm is competitive. *J. ACM*, 47(1):1–15, 2000.

[12] M. A. Bender, A. Fernández, D. Ron, A. Sahai, and S. Vadhan. The power of a pebble: exploring and mapping directed graphs. *Inform. and Comput.*, 176(1):1–21, 2002.

[13] V. Bonifaci, K. Mehlhorn, and G. Varma. Physarum can compute shortest paths. In *Proc. Symp. Discrete Algorithms*, (SODA'12), pages 233–240. SIAM, 2012.

[14] E. A. Capaldi, A. D. Smith, J. L. Osborne, S. E. Fahrbach, S. M. Farris, D. R. Reynolds, A. S. Edwards, A. Martin, G. E. Robinson, G. M. Poppy, and J. R. Riley. Ontogeny of orientation flight in the honeybee revealed by harmonic radar. *Nature*, 403:537–540, 2000.

[15] B. Chazelle. Natural algorithms. In *Proc. Symp. Discrete Algorithms*, (SODA'09), pages 422–431. SIAM, 2009.

[16] T. S. Collett, E. Dillmann, A. Giger, and R. Wehner. Visual landmarks and route following in desert ants. *J. Comp. Physiol. A: Mol. Integr. Physiol.*, 170(4):435–442, 1992.

[17] C. Cooper, A. Frieze, and T. Radzik. Multiple random walks in random regular graphs. *SIAM J. Discrete Math.*, 23(4):1738–1761, 2009/10.

[18] F. Courchamp, T. Clutton-Brock, and B. Grenfell. Inverse density dependence and the allee effect. *Trends Ecol. Evol.*, 14(10):405–410, 1999.

[19] S. Das, P. Flocchini, N. Santoro, and M. Yamashita. On the computational power of oblivious robots: forming a series of geometric patterns. In *Proc. Symp. Principles Dist. Comput.*, (PODC'10), pages 267–276. ACM, 2010.

[20] E. D. Demaine, S. P. Fekete, and S. Gal. Online searching with turn cost. *Theoret. Comput. Sci.*, 361(2-3):342–355, 2006.

[21] X. Deng and C. H. Papadimitriou. Exploring an unknown graph. *J. Graph Theory*, 32(3):265–297, 1999.

[22] A. Dessmark and A. Pelc. Optimal graph exploration without good maps. *Theoret. Comput. Sci.*, 326(1-3):343–362, 2004.

[23] K. Diks, P. Fraigniaud, E. Kranakis, and A. Pelc. Tree exploration with little memory. *J. Algorithms*, 51:38–63, 2004.

[24] C. A. Duncan, S. G. Kobourov, and V. S. A. Kumar. Optimal constrained graph exploration. *ACM Trans. Algorithms*, 2(3):380–402, 2006.

[25] R. Elsässer and T. Sauerwald. Tight bounds for the cover time of multiple random walks. *Theoret. Comput. Sci.*, 412(24):2623–2641, 2011.

[26] R. Fleischer and G. Trippen. Exploring an unknown graph efficiently. In *Proc. Eur. Conf. Algorithms*, (ESA'05), pages 11–22. Springer-Verlag, 2005.

[27] P. Flocchini, D. Ilcinkas, A. Pelc, and N. Santoro. How many oblivious robots can explore a line. *Inform. Process. Lett.*, 111(20):1027–1031, 2011.

[28] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous robots with limited visibility. *Theor. Comput. Sci.*, 337:147–168, June 2005.

[29] P. Fraigniaud, L. Gąsieniec, D. R. Kowalski, and A. Pelc. Collective tree exploration. *Networks*, 48(3):166–177, 2006.

[30] P. Fraigniaud, A. Korman, and D. Peleg. Local distributed decision. In *Found. Comput. Sci.*, (FOCS'11), pages 708–717. IEEE, 2011.

[31] L. Gąsieniec, A. Pelc, T. Radzik, and X. Zhang. Tree exploration with logarithmic memory. In *Proc. Symp. Discrete Algorithms*, (SODA'07), pages 585–594. ACM, 2007.

[32] V. Gervasi and G. Prencipe. Coordination without communication: the case of the flocking problem. *Discrete Appl. Math.*, 144(3):324–344, 2004.

[33] L.-A. Giraldeau and T. Caraco. *Social Foraging Theory*. Princeton University Press, 2000.

[34] L.-A. Giraldeau and D. Gillis. Do lions hunt in group sizes that maximize hunters' daily food returns? *Anim. Behav.*, 36(2):611–613, 1988.

[35] L.-A. Giraldeau, D. L. Kramer, I. Deslandes, and H. Lair. The effect of competitors and distance on central place foraging eastern chipmunks, tamias striatus. *Anim. Behav.*, 47(3):621–632, 1994.

[36] R.D. Harkness and N.G. Maroudas. Central place foraging by an ant (*Cataglyphis bicolor* fab.): a model of searching. *Anim. Behav.*, 33(3):916–928, 1985.

[37] K. Holder and G. A. Polis. Optimal and central-place foraging theory applied to a desert harvester ant, *Pogonomyrmex californicus*. *Oecologia*, 72(3):440–448, 1987.

[38] B. Holldobler and E. O. Wilson. *The Ants*. Harvard University Press, 1990.

[39] J. U. M. Jarvis, N. C. Bennett, and A. C. Spinks. Food availability and foraging by wild colonies of damaraland mole-rats (*Cryptomys damarensis*): implications for sociality. *Oecologia*, 113(2):290–298, 1998.

[40] E. Kagan and I. Ben-Gal. An informational search for a moving target. In *Conv. Electr. Electron. Eng. Isr.*, pages 153–155, 2006.

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

[42] R. M. Karp, M. Saks, and A. Wigderson. On a search problem related to branch-and-bound procedures. In *Proc. Symp. Found. Comput. Sci.*, (FOCS'86), pages 19–28. IEEE, 1986.

[43] A. Korman, J.-S. Sereni, and L. Viennot. Toward more localized local algorithms: removing assumptions concerning global knowledge. In *Proc. Symp. Principles Dist. Comput.*, (PODC'11), pages 49–58. ACM, 2011.

[44] J. R. Krebs. Optimal foraging, predation risk and territory defense. *Ardea*, 68:83–90, 1980.

[45] A. López-Ortiz and G. Sweet. Parallel searching on a lattice. In *Proc. Can. Conf. Comput. Geom.*, (CCCG'01), pages 125–128, 2001.

[46] J. C. Mitani and P. S. Rodman. Territoriality: The relation of ranging pattern and home range size to defendability, with an analysis of territoriality among primate species. *Behav. Ecol. Sociobiol.*, 5(3):241–251, 1979.

[47] G. F. Orians and N. E. Pearson. On the theory of central place foraging. *Anal. Ecol. Syst.*, pages 155–177, 1979.

[48] P. Panaite and A. Pelc. Exploring unknown undirected graphs. *J. Algorithms*, 33(2):281–295, 1999.

[49] M. M. Polycarpou, Y. Yang, and K.M. Passino. A cooperative search framework for distributed agents. In *Proc. Int. Symp. Intell. Control*, (ISIC'01), pages 1–6. IEEE, 2001.

[50] O. Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, 2008.

[51] A. M. Reynolds. Optimal random Lévy-loop searching: New insights into the searching behaviours of central-place foragers. *Europhys. Lett.*, 82(2):20001, 2008.

[52] A. M. Reynolds, A. D. Smith, R. Menzel, U. Greggers, Donald R. Reynolds, and J. R. Riley. Displaced honey bees perform optimal scale-free search flights. *Ecology*, 88(8):1955–1961, 2012/04/29 2007.

[53] A. M. Reynolds, A. D. Smith, D. R. Reynolds, N. L. Carreck, and J. L. Osborne. Honeybees perform optimal scale-free searching flights when attempting to locate a food source. *J. Exp. Biol.*, 210:3763–3770, 2007.

[54] A.M. Reynolds. Cooperative random Lévy flight searches and the flight patterns of honeybees. *Phys. Lett. A*, 354(5-6):384–388, 2006.

[55] T. W. Schoener. Theory of feeding strategies. *Annu. Rev. Ecol. Syst.*, 2:369–404, 1971.

[56] S. Sommer and R. Wehner. The ant's estimation of

distance travelled: experiments with desert ants, cataglyphis fortis. *J. Comp. Physiol. A: Mol. Integr. Physiol.*, 190(1):1–6, 2004.

[57] M. V. Srinivasan, S. Zhang, M. Altwein, and J. Tautz. Honeybee navigation: Nature and calibration of the "odometer". *Science*, 287(851–853), 2000.

[58] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363 (electronic), 1999.

[59] I. Suzuki and M. Yamashita. Erratum: "Distributed anonymous mobile robots: formation of geometric patterns" [SIAM J. Comput. **28** (1999), no. 4, 1347–1363 (electronic); mr1681010]. *SIAM J. Comput.*, 36(1):279–280 (electronic), 2006.

[60] J. F. A. Traniello. Recruitment behavior, orientation, and the organization of foraging in the carpenter ant *Camponotus pennsylvanicus* DeGeer (Hymenoptera: Formicidae). *Behav. Ecol. Sociobiol.*, 2(1):61–79, 1977.

[61] I. A. Wagner, Y. Altshuler, V. Yanovski, and A. M. Bruckstein. Cooperative cleaners: A study in ant robotics. *Int. J. Rob. Res.*, 27(1):127–151, 2008.

[62] R. Wehner, C. Meier, and C. Zollikofer. The ontogeny of foragwehaviour in desert ants, cataglyphis bicolor. *Ecol. Entomol.*, 29(2):240–250, 2004.

[63] R. Wehner and M. V. Srinivasan. Searching behaviour of desert ants, genus *Cataglyphis (Formicidae, Hymenoptera). J. Comp. Physiol. A*, 142(3):315–338, 1981.

[64] Y. Yang, A.A. Minai, and M.M. Polycarpou. Decentralized cooperative search by networked uavs in an uncertain environment. In *Proc. Am. Control Conf., (ACC'04)*, volume 6, pages 5558–5563. IEEE, 2004.